

MassMotion

# MassMotion

Help Guide

# Oasys

YOUR IDEAS BROUGHT TO LIFE

13 Fitzroy Street  
London  
W1T 4BQ  
Telephone: +44 (0) 20 7755 3302  
Facsimile: +44 (0) 20 7755 3720

Central Square  
Forth Street  
Newcastle Upon Tyne  
NE1 3PL  
Telephone: +44 (0) 191 238 7559  
Facsimile: +44 (0) 191 238 7555

e-mail: [oasys@arup.com](mailto:oasys@arup.com)  
Website: <http://www.oasys-software.com/>

# MassMotion

© 2019 Oasys Software Limited

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Printed: April 2019

# Table of Contents

<b>Part I Welcome to MassMotion</b>	<b>7</b>
<b>Part II What's New in 10</b>	<b>9</b>
<b>1 List of Changes</b> .....	<b>9</b>
<b>2 Unicode Support</b> .....	<b>11</b>
<b>Part III User Guide</b>	<b>13</b>
<b>1 Installation</b> .....	<b>13</b>
System Requirements .....	13
Licensing .....	13
MassMotion & Flow .....	14
<b>2 How MassMotion Works</b> .....	<b>15</b>
The Scene .....	15
People as Agents .....	19
Using MassMotion .....	20
<b>3 Project Workflow</b> .....	<b>21</b>
Authoring .....	21
Simulation .....	51
Analysis .....	52
<b>4 Troubleshooting</b> .....	<b>54</b>
Auditing .....	54
Validation .....	54
Observing Agents or Objects .....	54
Finding Object References .....	55
Debugging a Simulation .....	55
Using Analysis to Diagnose Issues .....	56
Auto Save .....	56
Graphics .....	56
<b>Part IV Reference</b>	<b>58</b>
<b>1 Project</b> .....	<b>58</b>
Opening Projects .....	58
Importing Data .....	60
Exporting Data .....	64
Project Settings .....	69
Files .....	70
<b>2 User Interface</b> .....	<b>71</b>
Main Window .....	71
Selection .....	82
Manipulator .....	83
Generating Objects from Geometry .....	97
Working with Drawings .....	107
Editing Object Properties .....	114
Process Chains .....	117
Choosing Objects .....	120



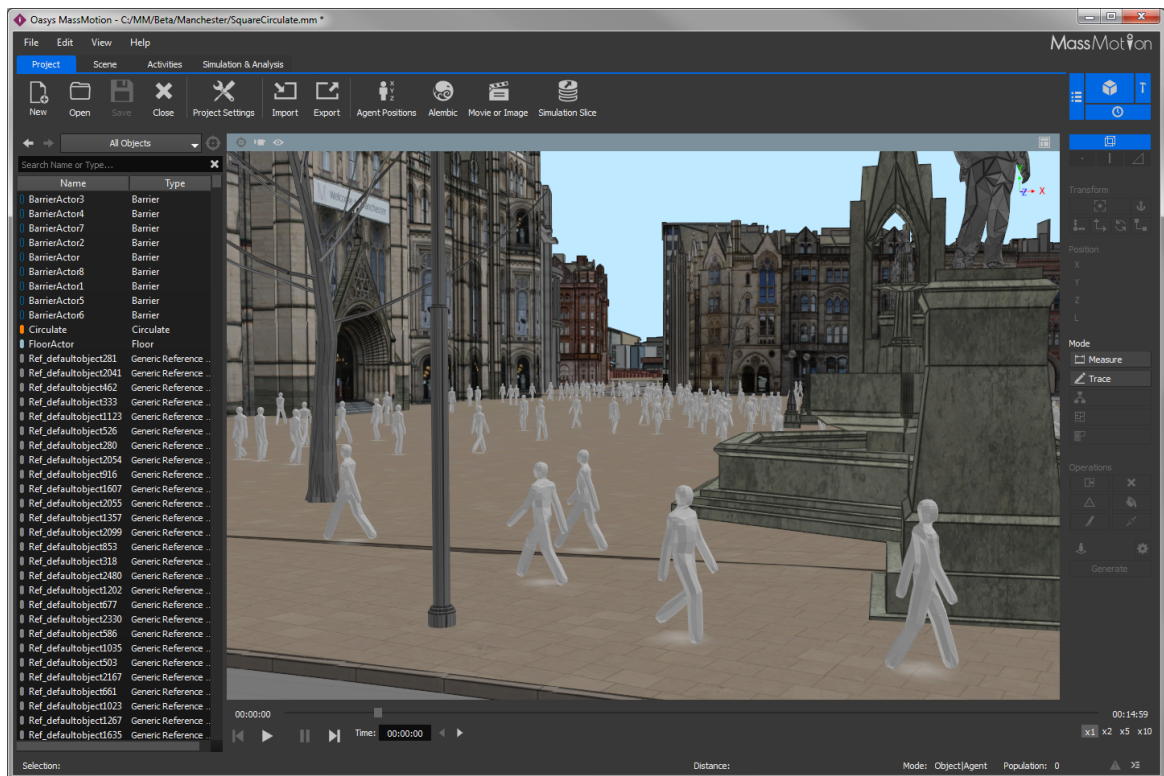
---

Working with Colours .....	123
Working with Time .....	125
Triggering Events .....	127
Issue Window .....	128
Keyboard and Mouse Controls .....	128
Application Preferences .....	131
Fixing Face Normals .....	133
<b>3 Objects .....</b>	<b>133</b>
Scene Objects .....	134
Collections .....	175
Activities .....	182
Analysis .....	236
Bookmark .....	287
<b>4 Simulation .....</b>	<b>288</b>
Agent Behaviour .....	288
Surface Maps .....	292
Areas .....	294
Wait Style .....	295
Simulation Execution .....	296
Generated Simulation Files .....	307
Randomness .....	308
<b>5 Analysis &amp; Reporting .....</b>	<b>310</b>
Observers .....	311
Transition .....	313
LOS Colour Mapping .....	314
Decorations .....	315
<b>6 Working with the Viewer .....</b>	<b>319</b>

# Part I

---

# 1 Welcome to MassMotion



The field of pedestrian planning is rapidly growing as design professionals respond to a world where population density is increasing. This leads to greater emphasis on the efficiency and safety of commercial buildings, performance venues, schools, transit facilities and public areas. The process of understanding how people will move through and occupy a finished project is both challenging and fascinating.

MassMotion is developed to enable design and planning professionals to rapidly test and analyse the movement of people in many kinds of environments. To do this MassMotion provides users with a suite of tools for creating and modifying 3D environments, defining operational scenarios, executing dynamic simulations and developing powerful analyses.

A range of introductory videos are provided on the [Oasys product page](#) to enable users to quickly begin modeling and simulating with MassMotion. To unlock the full potential of the MassMotion toolset users are encouraged to review the [User Guide](#) and to consult the [Reference](#) as necessary.

# Part II

---

## 2 What's New in 10

MassMotion version 10 adds native elevator objects, wait spaces, custom route networks, tally objects, improved performance, support for international characters, and much more.

### Elevators

MassMotion now supports basic elevators or lifts with a new [elevator](#) object. Elevators are quick to add and require little setup. Several [behaviours](#) have been provided, including basic call/answer, a simple 'visit every stop' mode, and shuttle based evacuation. Multiple elevators can be grouped together for coordinated movements using an [elevator bank](#).

### Custom Route Networks

Route choice can be limited to particular objects or object types through the use of new [network](#) objects. Persons with reduced mobility might be given a network that excludes stairs. Or an elevator might become unavailable with the onset of an evacuation event. Networks will allow greater customization of the different populations within a simulation.

### Tally

[Tally](#) objects provide a mechanism for storing and then responding to values in the simulation. A value can be a measured quantity such as an area population or an abstract user defined value. User defined values can be scaled, added, or modified by [actions](#) or the [cache change](#) event. The value of a tally can be used in [tests](#) or [triggers](#) to control other actions or events. A tally might be used to count the number of agents that cross a turnstile while holding a bag token, or it might be used to trigger the closing of a gate whenever the population in one area is greater than the population in another.

### Waiting

Waiting agents can be constrained to a particular area through the use of new [wait space](#) objects. Wait spaces contain a target which can be used to control how agents congregate within the area. Wait spaces are assigned like any other wait style through the property of the target gate, elevator, or as part of the wait task. A new wait style, "Focus beside target" has been added to force agents to wait on either side of the gate or elevator. The "Spread out" wait style has been improved to reduce agent movement in dense environments.

### General

There have been general improvements to authoring performance. International (unicode) characters are now supported for both file names and object names (see [unicode support](#)). It is easier to create generic objects with predefined shapes. And a new [expected demand OD count matrix](#) query will output the expected demand OD for a given journey, evacuate, or circulate event.

For full details on what has changed in version 10 see [List of Changes](#).

## 2.1 List of Changes

### Elevators

- Created new [Elevator](#) object for simulating elevator/lift movement between floors.
- Created new [Elevator Bank](#) for coordinating control of connected elevators.

### Network Accessibility

- Created new [Network](#) object for defining a subset of the route network (e.g. no stairs).
- Added network property to [Profile](#).
- Added 'Assign network' action for dynamically changing the network used by an agent.
- Added support for network objects in [static cost](#) and [static distance](#) maps.

### Wait Spaces

- Created new [Wait Space](#) object for defining areas in which agents can wait (set through 'wait style' property of gates /elevators / tasks).
- Added support for wait space based waiting to gates, wait tasks, and elevators.

### Waiting Behaviour

- Added new 'Focus beside target' for agents standing on either side of a gate or elevator.
- Improved 'Spread out' behaviour such that agents are more willing to stand still in dense areas.
- Improved 'Move aside' behaviour when waiting for priority access to a link.

### Tally

- Created new [Tally](#) object for storing, modifying, combining, or testing measured or abstract values.
- Added the [Count Change](#) event or the "Add to count" [action](#) to modify abstract tally counts.
- Added the [Tally test](#) or [Tally trigger](#) to alter simulation operations based on the value of a tally.
- Upgraded existing count based triggers and tests to use tally objects.

### Internationalization

- Added support for international (unicode) characters in file names.
- Added partial support for international (unicode) characters in object names (see [Unicode Support](#)).

### Performance

- Changed network initialization to only create cost trees for portals/zones/servers that are used as destinations.
- Improved performance when showing/hiding members of a Transform
- Improved performance deleting members of a Transform.
- Improved performance when deleting large numbers of objects.

### General

- Added 'Seek area' [action](#) to send agents to any point in the route network.
- Renamed bank objects to route bank objects.
- Fixed bug where agents might not be aware of a long-way-around route that crosses an 'open' perimeter.
- Added creation of objects with pre-set shapes (L shaped floor, straight run stair, barrier column, etc.).
- Added 'Find -> Used by -> Profiles' context menu entry.
- Fixed bug in 'area population' trigger and test where agents could be double-counted if appearing in multiple volumes.
- Added missing 'Save as...' File entry for some object property windows.
- Fixed bug where large custom avatars would disappear when only partially in view.

### Analysis

- Added '[Agent Transition](#)' table query for recording each time an agent executes a transition.
- Added '[Expected Demand OD Count](#)' query for displaying expected counts from a journey, evacuate, or circulate event.
- Added [Tally Count](#) graph query for displaying tally object values over time.

### Viewer

- Added database slice 'mmv' format that can only be opened in the viewer (not MassMotion or Flow).
- Added new entry in the user guide to explain how to make the most out of presenting results with the free Viewer (see [Working with the Viewer](#)).

## 2.2 Unicode Support

International (unicode) characters are now supported throughout most of MassMotion.

The following areas are still restricted to ASCII characters:

- Logic diagrams generated for actions, tests, and tally objects.
- The names of objects [exported](#) using the obj format.

# Part III

---



## 3 User Guide

The user guide provides a brief introduction to MassMotion. It describes some of the key concepts and outlines basic project workflow. It is intended to be used in combination with the video tutorials available at <http://www.oasys-software.com> and the comprehensive [Reference](#) section.

### 3.1 Installation

The most up to date version of MassMotion may be downloaded at <http://www.oasys-software.com>.

#### 3.1.1 System Requirements

MassMotion is a high performance 64bit multi-threaded application. Advanced hardware is recommended when running simulations with very high concurrent agent counts (25,000+). It is especially important to have a high CPU count with high processor speed and a fast solid state storage device for efficient database transactions. GPU performance is important for highly detailed 3D environments and for 3D playback of high agent count environments.

MassMotion uses Windows Media technology for generating videos and images. On some versions of Windows (N/KN) the Windows Media Feature Pack may need to be installed separately. The Windows Media Feature Pack is available for free and can be downloaded from the Microsoft support website.

**Recommended Minimum Specification:**

- Windows 64bit OS (Windows 7 & later)
- 8-core Intel or AMD workstation/server class CPU
- 16GB of RAM
- OpenGL 4 workstation GPU from NVIDIA or AMD
- 500GB Solid State Hard Drive
- 2 1680x1050 display monitors
- 3-button mouse

**Minimum Specification:**

- Windows 64-bit OS (Windows 7 & later)
- Dual-core Intel or AMD workstation/server class CPU
- 4GB of RAM
- OpenGL 3.0 compatible GPU
- 500GB Hard Drive
- 1280x1024 display monitor
- 3-button mouse

#### 3.1.2 Licensing

MassMotion licensing can be per machine, networked, or across an educational institution. For information on how to purchase MassMotion please visit <http://www.oasys-software.com>.

**Standalone Licenses**

Standalone licenses may only be installed on one computer at a time. Users may switch the license from one machine to another by deactivating it on the current machine and then activating it on the new one. This licensing scheme does not require an active internet connection during use but an internet connection is required for initial activation and transfer. Licenses can be managed through the "Help -> Licensing..." menu.

To activate or move your license see <https://www.oasys-software.com/support/licensing-of-oasys-software/>.

**Shared & Virtual Licenses**

Shared licenses may be utilized by any machine within an organization, provided that the total number of concurrently active users does not exceed the number of shared licenses. This is the required licensing option for virtual machines.

To activate your license see <https://www.oasys-software.com/support/licensing-of-oasys-software/>.

**University Licenses**

Site licenses are typically used by educational institutions and licensing is controlled by designated IP address. Any machine that connects to the internet using the IP address will be granted a 60-day license that will be automatically renewed each time MassMotion is launched while connected to the internet through the designated network.

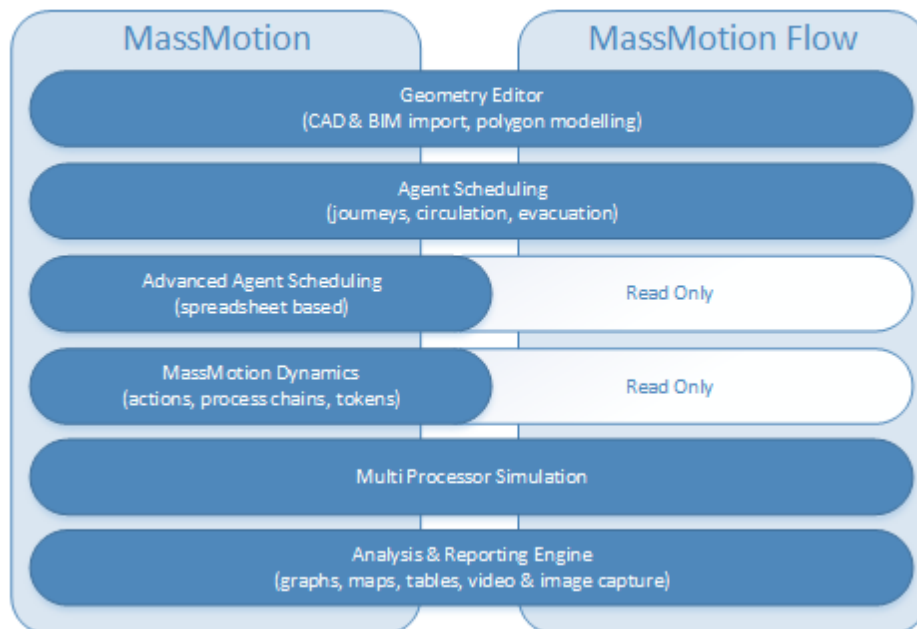
To learn more about site licenses see <https://www.oasys-software.com/education/>.

To check the activation of an existing site license see [https://www.oasys-software.com/unipac\\_check/](https://www.oasys-software.com/unipac_check/).

**3.1.3 MassMotion & Flow**

Oasys offers two crowd simulation products: MassMotion and Flow. The project files (.mm) and results databases (.mddb) from each version may be opened and used in the other. The two versions are differentiated by the extent of the authoring and agent scheduling features available as per the diagram below. While Flow can run simulations or analyse projects with actions, complex events, and process modeling, only MassMotion can create or edit those components.

MassMotion Product Features Map



## 3.2 How MassMotion Works

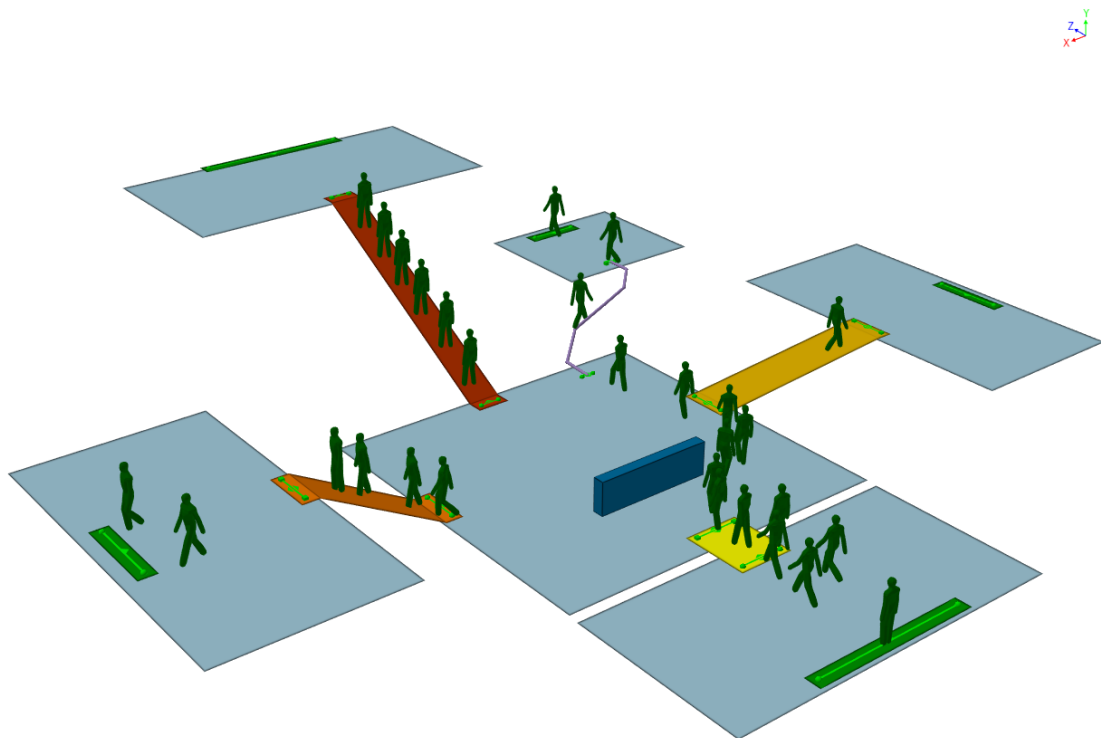
The following sections provides a brief introduction to some of the main concepts in MassMotion. [The Scene](#) covers topics related to the simulation environment and [People as Agents](#) describes the representation of people.

### 3.2.1 The Scene

MassMotion models real world spaces by breaking those spaces into component parts and classifying the parts according to function. People in a MassMotion simulation know to walk around an obstruction because it has been marked as a barrier. Speed of movement is reduced when walking up a surface because that surface has been marked as a stair.

The way in which classified objects are arranged can have a large impact on how people navigate a space, affecting their speed, their movement patterns, and their route choices.

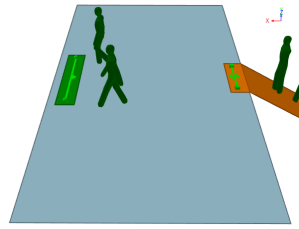
The basic elements of a scene are: [floor](#), [link](#), [stair](#), [ramp](#), [escalator](#), [path](#), [elevator](#), [portal](#), and [barrier](#).



MassMotion Scene

#### Floors

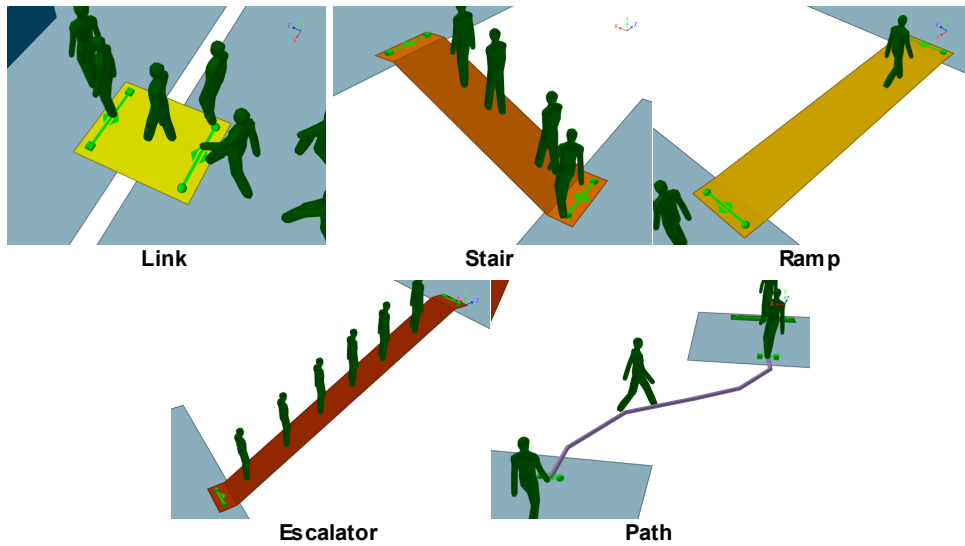
[Floors](#) are the most fundamental scene object. They represent the spaces (rooms, hallways, plazas, sidewalks, train platforms) which define the program areas of a design. Each floor defines a separate and distinct walkable area, with pedestrian movement constrained by the floor boundary.



Floor

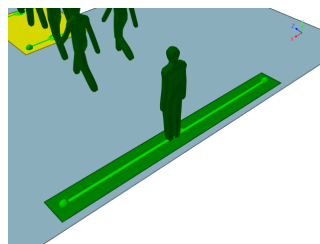
**Connections Between Floors**

People may only move between floors where they are connected by [connection objects](#). [Links](#) represent simple flat doorways or turnstiles. [Stairs](#), [ramps](#), [escalators](#), and [elevators](#) connect floors at different elevations. [Paths](#) connect floors at any elevation and restrict movement to single file.



**Entrances and Exits**

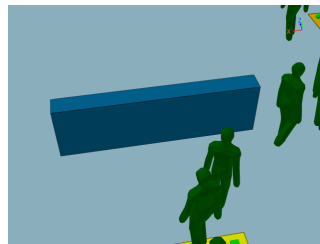
[Portals](#) serve two main functions: they mark areas where people can enter the simulation and they represent destinations to which people can be sent.



Portal

**Obstructions**

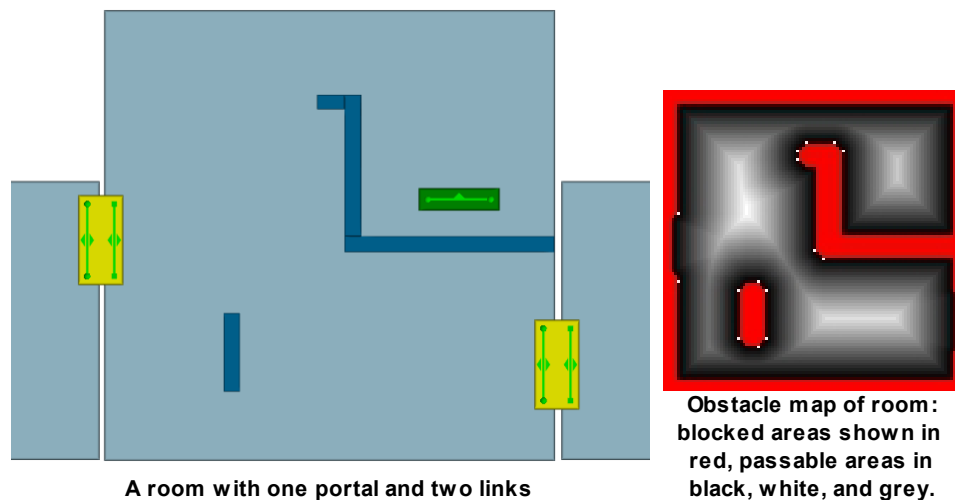
[Barriers](#) represent walls, columns, tables, benches, and anything else that can constrain movement on a floor.



Barrier

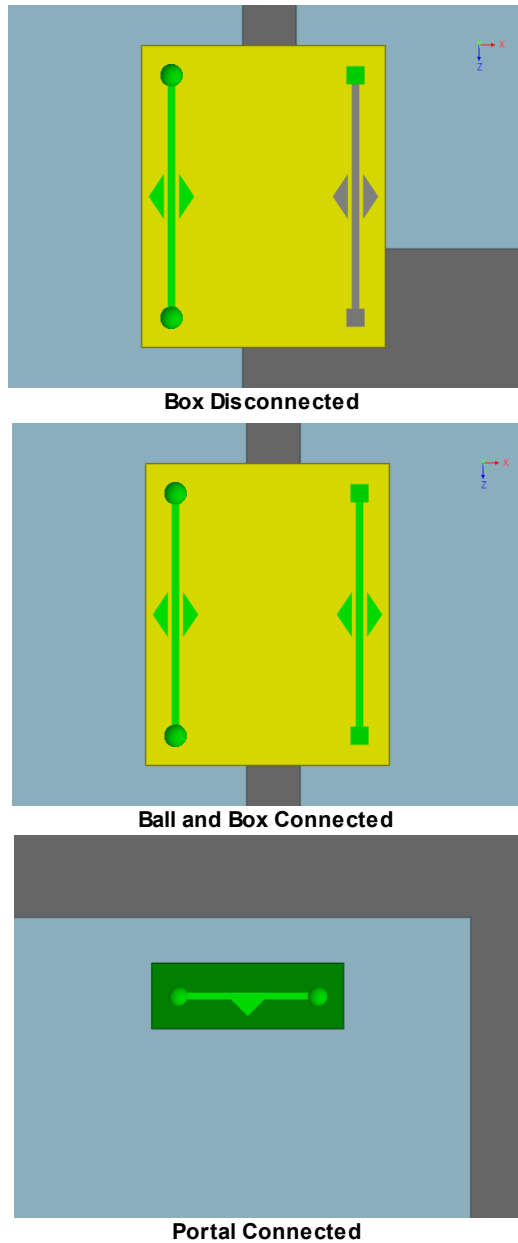
### 3.2.1.1 Determining Walkable Space

Floors, links, stairs, ramps, and escalators all define areas in which people can walk. People are constrained by the edges of an object and by any barriers placed on the object. MassMotion distinguishes between passable areas and obstructed areas using [surface maps](#). Each object will result in several different surface maps, all describing different aspects of the walkable space on the object.



### 3.2.1.2 Connecting Spaces Together

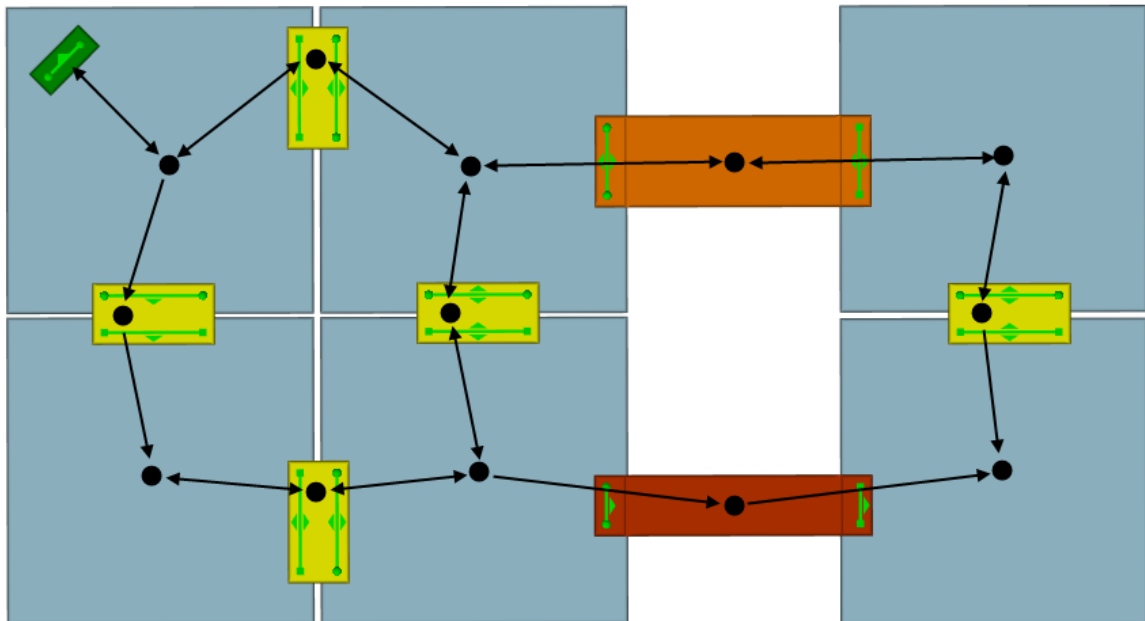
Different scene elements can be connected together by goal lines. Each [connection object](#) (i.e. [link](#), [stair](#), [ramp](#), [escalator](#) or [path](#)) has a box goal line at one end and a ball goal line at the other. [Portals](#) have a single goal line. [Elevators](#) have a goal line for each stop. An object will automatically connect to a floor when the goal line is just above the floor and not too close to the floor edge.



### 3.2.1.3 The Network

The arrangement of connected [floors](#), [connection objects](#) and [portals](#) is termed the network. The network describes all possible routes in the scene and remains fixed throughout the simulation.

The network is used by people when trying to find a particular goal. Each connection object attached to a floor is like a decision point. While standing on a floor, people will evaluate nearby connection objects and choose the one that provides the best route to their goal.



A simple scene. The black circles and lines represent nodes and routes through the network.

### 3.2.2 People as Agents

Every person in a MassMotion simulation is an autonomous agent. Each agent has the ability to monitor and react to its environment according to a unique set of characteristics and goals.

#### Characteristics

The physical characteristics and personality of an agent are defined by a [profile](#). The profile defines a distribution of values for properties such as size, speed, and route preferences.

#### Scheduling

Agents are placed in the scene through [events](#). All events will specify where an agent should start, at what time it should appear, and give the agent an initial goal or purpose.

#### Behaviour

Agent behaviour is determined by [task](#). Agents are given one or more tasks when initially placed in the scene and will work through tasks one at a time in order. Different types of events will assign different goals or behaviours. A [journey](#) will produce agents that move from A to B. [Circulate](#) will create agents that move between a series of waypoints, waiting at each stop. [Evacuate](#) will create agents that initially wait for a period and then seek the closest exit.

Agents execute tasks using two independent agent systems: the navigation system and the movement system.

#### Navigation

The [navigation system](#) is responsible for determining how best to accomplish a task. When seeking a particular portal, the agent must evaluate its surroundings and determine the best route to that portal. This determination is based on an awareness of the environment, both in terms of the distances involved and a limited sense of congestion at some of the near decision points. Once a route has been chosen, that choice is periodically re-evaluated as the agent progresses along the route.

#### Movement

Once the agent has chosen where to go, its [movement system](#) guides the agent across the floor towards its choice. This system relies on a modified version of the Social Forces<sup>1</sup> algorithm. A series of forces are generated based on the direction the agent wants to go, the location and movement of neighbouring agents, and the position of nearby obstacles. These forces are summed at every time step and used to determine the agent's heading and velocity.

[1] Dirk Helbing and Péter Molnár Social force model for pedestrian dynamics II. Institute of Theoretical Physics, University of Stuttgart, 70550 Stuttgart, Germany, January 1995

### 3.2.3 Using MassMotion

#### Main Window

Each area in the [main window](#) provides access to different parts of the project discussed in detail in the [reference](#) section. In general the main components are:

Area	Description
<b>Ribbon</b>	The ribbon is a tabbed toolbar which provides buttons for performing high level operations. The project tab provides buttons for managing the project and importing or exporting content. The scene and activities tabs provide buttons for creating scene and activity related objects. The analysis tab provides buttons for validating the project, running a simulation, and creating analysis objects.
<b>List View</b>	The list view provides a mechanism for accessing the objects in the project by name or type. A filter at the top controls the type of object displayed. The search bar can be used to find objects by name or type. The capsule drawn beside each object displays information on the type, colour, and status of the object. See <a href="#">List View</a> for more information.
<b>Scene View</b>	The <a href="#">3D scene view</a> is a graphical representation of the scene objects in the project. The camera can be moved by holding down the 'S' key while dragging the left or right mouse buttons. It is possible to select objects by clicking on them or dragging over them with a selection box (see <a href="#">Selection</a> ). A menu bar along the top of the 3D scene view provides control over the current camera, bookmarks, and general view settings.
<b>Tool Panel</b>	The <a href="#">tool panel</a> contains a number of controls for interacting with the scene view. It displays and controls the current selection mode. It displays and controls the position, translation, rotation, and scale of the current selection. It also contains buttons for operating on the geometry of the selected object, measuring the distance between points in the scene, and tracing lines to generate new objects.
<b>Time Panel</b>	The <a href="#">time panel</a> is used to control the playback of data from one or more simulation runs. It is only available when the project has a simulation run with valid data.

#### Objects

Much of the data in MassMotion is represented in the form of objects and most of the work in setting up, running, and analyzing a simulation involves viewing and managing these objects. Some objects have a physical presence in the simulation, some objects describes demand schedules, some objects control operations, while other objects analyze simulation results.

#### Selection

The selection is a list of objects or object components chosen by the user. Objects can be selected in the [scene view](#), [list view](#), or through individual choosers. The selection mode, which toggles



between selecting objects, faces, edges, or vertices, can be changed through [keyboard shortcuts](#) or buttons in the [tool panel](#). The selection status of an object or object component is a property of the object and so visible in all parts of the user interface. If an object is selected in the [list view](#), it will also show as selected in the [scene](#).

## 3.3 Project Workflow

There are three main stages to working with a project: [authoring](#), [simulation](#), and [analysis](#). The evolution of a project is often iterative, with results from simulation or analysis leading back to authoring changes and additional simulation runs.

### 3.3.1 Authoring

Authoring involves creating a representation of a physical space in the simulation environment. This representation is referred to as the scene. Authoring also includes the construction of events for creating agents and controlling operations.

#### 3.3.1.1 Building the Scene

The physical environment in a MassMotion project is defined by scene geometry contained in one or more scene objects. The objects each have a particular purpose in a simulation based on their type (see [Modeling Strategies](#)).

It is possible to create new [default objects](#) with standard shapes and then [edit the geometry](#) to create the desired scene. It is also possible to [import content](#) from another application, and then [convert the imported objects and/or drawings](#) into MassMotion objects of the appropriate type.

##### 3.3.1.1.1 Creating Default Objects

New MassMotion objects can be created through the ribbon buttons in the Scene tab at the top of the [main window](#), or through the right click menu in the [3D scene view](#). Objects created through the ribbon can be given initial shapes depending on the type of object. All new objects can be [edited](#) to match requirements.

##### 3.3.1.1.2 Selecting Object Components

To select an object for editing, click on the object with the left mouse button. To select components for editing, first select the object and then change the selection mode in the top right of the main window. This will display the components of the selected object which can themselves be selected using the left mouse button.

It is not possible to select a new object when in component selection mode. First return to object mode, add the new object to the selection, then return to the component selection mode.

The following lists show object types and their corresponding components.

#### 3D Mesh Objects

- Face
- Edge
- Vertex

### Line Objects

- Edge
- Vertex

### Drawing Layers

- Line
- Point

#### 3.3.1.1.3 Importing Geometry or Drawings

Geometry, drawings, or images can be imported from files using the Import button in the 'Project' tab or the Import section of the 'File' menu. See [Import Files](#) for information on supported file formats.

Imported geometry is referred to as [reference geometry](#) and is primarily used to [generate MassMotion objects](#). It can also be used as a visual aid during playback or for recorded videos. Imported geometry is not used during simulation or analysis.

### IFC Geometry

The contents of IFC files are imported as [IFC geometry](#), where IFC type information is retained. This type information can be used to automatically [generate](#) corresponding MassMotion objects.

### 3D Geometry

All non-IFC geometry is imported as [generic geometry](#). This can also be used to generate MassMotion objects but the user must specify the target object type.

### 2D Points/Lines

All 2D elements within a file are imported as [drawing layer](#) objects. Drawing layers can be used separately or merged together. The lines and enclosed spaces/regions within a drawing can be used to [generate](#) MassMotion objects.

### Images

Image files can be imported as textures which are automatically placed on a 2D rectangle. These textured surfaces can then be used to trace new objects.

#### 3.3.1.1.4 Creating Objects from Geometry or Drawings

MassMotion objects can be generated from traced lines or [imported geometry or drawings](#) using the right-click 'Generate' menu commands or the 'Generate' button in the [tool panel](#).

### Tracing new Objects

Lines can be traced over existing drawings, imported images, or 3D objects. For information on using traced lines to generate objects see [Tracing New Objects](#).

### Generating from Reference Geometry

Imported reference geometry objects can be used to generate new scene objects. When working with IFC geometry, the target type for a given source object can be determined automatically. See [IFC Geometry](#) for a mapping of source IFC type to MassMotion type.

When a specific target type is chosen, a transformation is applied to the source object. The transformation is based on the source and target types. See [Generating From Generic Geometry](#) for a list of the default transformations. It is also possible to customize the transformation operations through the [Generate Options](#).

When a new object is generated from reference geometry, the source object is deleted. This behaviour can be changed through the [Application Preferences](#).

### Generating from Drawing Lines/Regions

The lines and empty spaces within drawing layer objects can be used to generate objects (see [Generating From Drawings](#)). Drawing lines can be extruded out of the plane to form barriers, expanded in plane to form a link, or spanned (between two lines) to form stairs or escalators. Voids enclosed by drawing lines can also be used to generate floors or barriers. The voids are called regions, and once found can be selected and used to generate objects like any other component (see [Finding Regions](#) for information on regions and [Healing Drawings](#) for editing lines to ensure good region construction).

### Copying Reference Geometry, Faces, or Regions

The 'Convert' context menu available when right-clicking on selected reference geometry will convert the selected objects into a new object of the specified target type. The source geometry will not be modified. The same operation can be applied to mesh faces or drawing regions through the 'Copy to' context menu.

### Projecting Objects onto Drawings

Drawing layers can be used to create objects from the spaces between reference geometries (e.g. floors from surrounding walls). 3D geometry can be projected onto a new horizontal drawing layer through the 'Generate' right-click menu. The new drawing layer can then be used to generate other objects as required.

#### 3.3.1.1.5 Editing Geometry or Drawings

Geometry or drawings can be edited either by moving the entire object, or by manipulating individual components of the object.

Many of the tools for editing geometry have [shortcuts](#) for quick access.

#### 3.3.1.1.5.1 Setting Position and Length

##### Setting Positions

The position of objects or object components can be set to a specific location using the 'Position' section in the Main Window's [tool panel](#). This is useful when a number of vertices need to be along the same line or at the same height.

In the case of objects, the position of the object centroid is displayed. In the case of object components, the position of the vertices connected to the components is displayed.

##### Setting Length

The 'Positions' section of the Main Window's [tool panel](#) can also be used to set the length of an edge. This is useful when a stair or door is required to be a certain width. Select the appropriate edges and enter the desired length in the L field.



Position section of the Model Panel

3.3.1.1.5.2 Geometry Manipulation

**Translating, Rotating, Scaling**

Objects or components can be moved around the scene using the transform [manipulators](#). These manipulators are available under the 'Transform' section of the [tool panel](#).

Select an object or component, select the transform, rotate, or scale manipulator, then either enter numbers directly or drag the manipulator to achieve the desired transform. In the case of the translate or rotate manipulator, the 'Shift' key can be held to snap movement to discrete increments.

In the case of translation, the axis arrows can be used to translate along a particular axis. Clicking between two axes will drag within the plane defined by the two axes (a yellow square indicates the plane of translation).

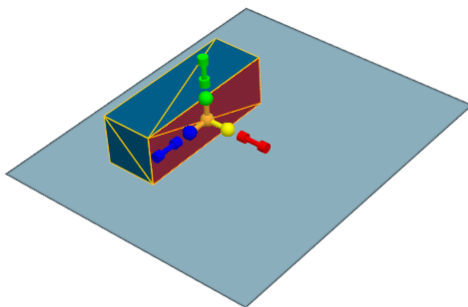
When scaling, the axis arrows can be used to scale only in a particular direction. Clicking between two axes will scale uniformly within that plane (both directions will scale equally). Clicking away from the manipulator (such that the entire manipulator is yellow) will scale equally in all directions.

**Snapping**

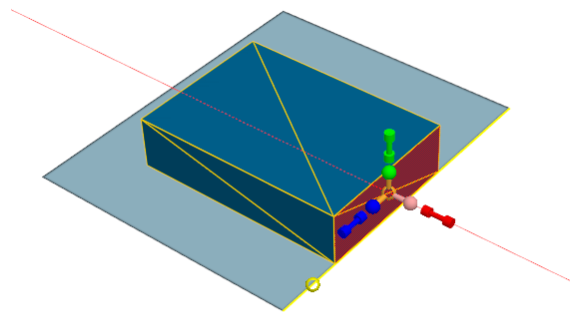
Objects or components can be moved in relation to each other using the [object snap](#) manipulator. This manipulator is also available under the 'Transform' section of the [tool panel](#).

Select an object or component, select the snap manipulator, then drag the manipulator to achieve the desired transform. The manipulator is comprised of the orange centre ball; the red, green and blue axis balls; and red, green and blue alignment barbells. The centre ball can be used to 'translate' to vertices, edges, edge midpoints or points on faces of other objects. The coloured axis balls perform the same operation, but 'constrained' to a specific axis direction. The alignment barbells are used to rotate the manipulator and selection so that the axis is parallel to a particular edge.

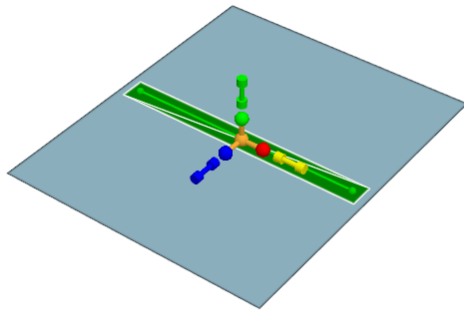
The vertex, edge or face being snapped to will be highlighted in yellow.



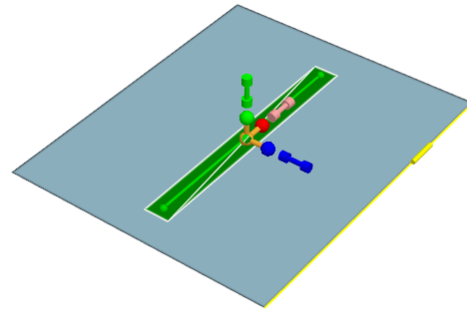
Before X constrained position snap



After X constrained position snap



Before alignment snap



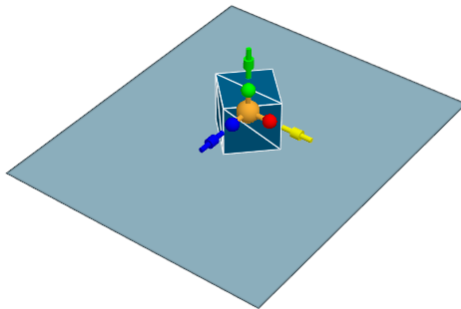
After alignment snap

See [Manipulator Tips and Tricks](#) for more uses of object snap.

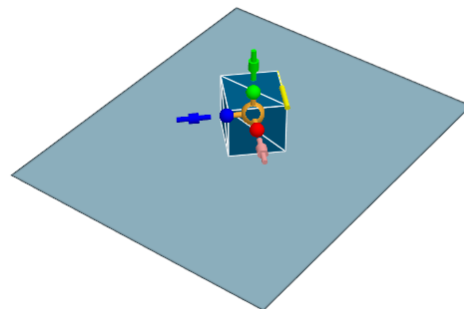
### Off-axis Operations

For off axis and uncentered operations, the manipulator snap sub-mode can be used to reposition or realign the manipulator with respect to the selected objects or components. Once repositioned the manipulator will retain its position and orientation for subsequent object transformations.

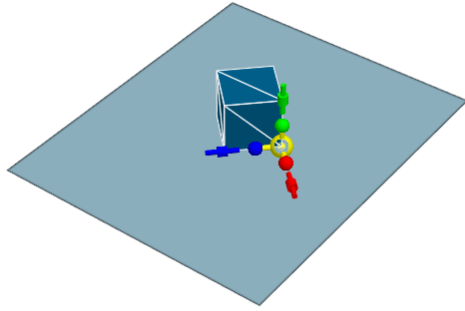
The following images show how to use manipulator snap to bring an off-axis barrier flush with a floor's edges. The first step is to use manipulator snap to re-orient the manipulator until it is aligned with the barrier, and then to move the manipulator to the barrier corner. Once the manipulator is positioned correctly, object snap can be used to position the barrier corner on the target floor corner and then to rotate the barrier until it is aligned with the floor.



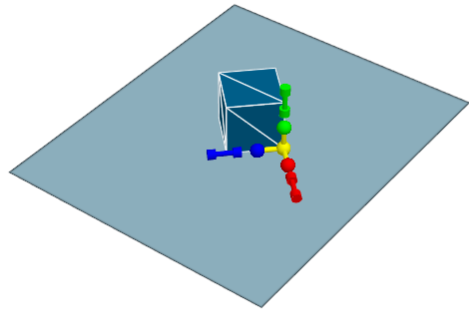
1) Activate "Move Manipulator"



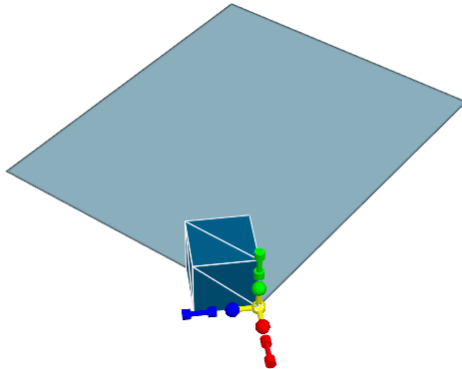
2) Drag red barbell to barrier edge



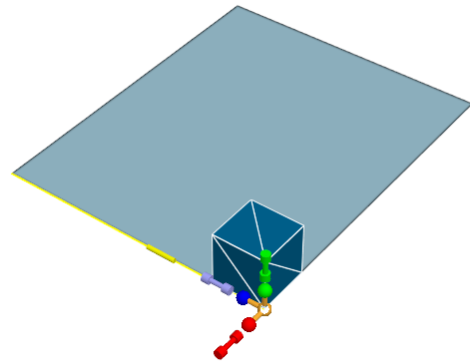
**3) Manipulator has been realigned**



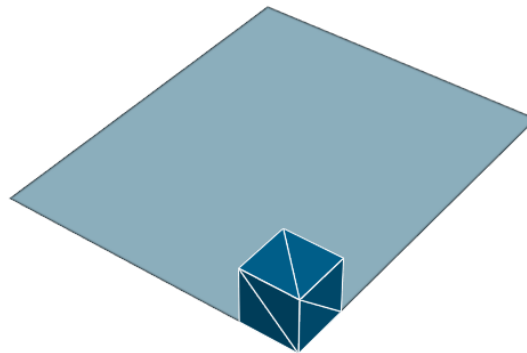
**4) Activate "Snap"**



**5) Drag orange to floor corner**



**6) Drag blue barbell to floor edge**

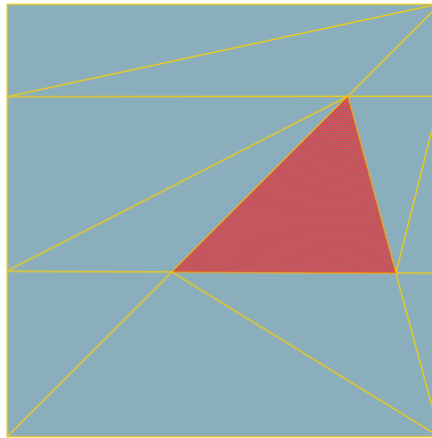


**7) Barrier now flush with floor edges**

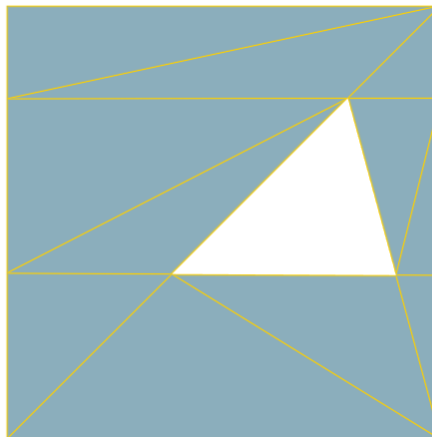
See [Manipulator Tips and Tricks](#) for more uses of manipulator snap.

## 3.3.1.1.5.3 Common Operations

Select an object or component and hit the delete key or use the delete button in the [tool panel](#) to delete the selection.

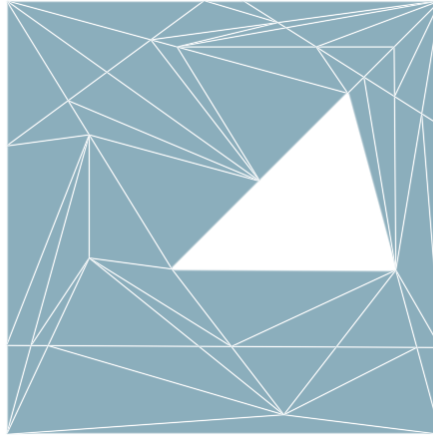


**Before delete face**

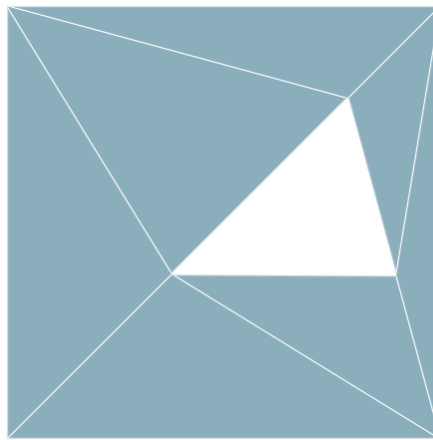


**After deletion of selected face**

If an object has been sliced, split, or grown many times, there can sometimes be an excess of faces. Use the 'Simplify' command in the [tool panel](#) or object right-click menu to reduce the object to its simplest form. The overall shape of the object will not be altered.

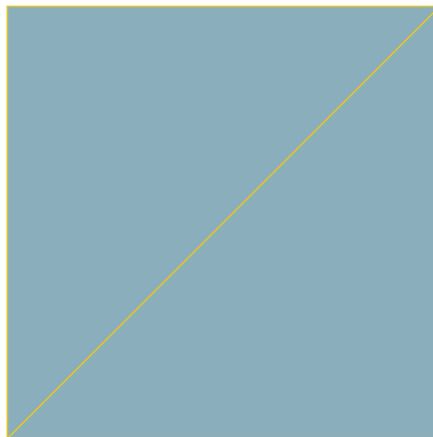


**Before simplify**



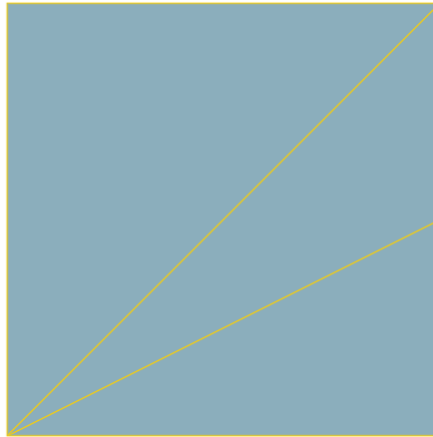
**After simplify operation on object**

Edges or faces can be split to provide additional edges or vertices for manipulation. Select one or more faces or edges and use the 'Split' button in the [tool panel](#) or component right-click menu.

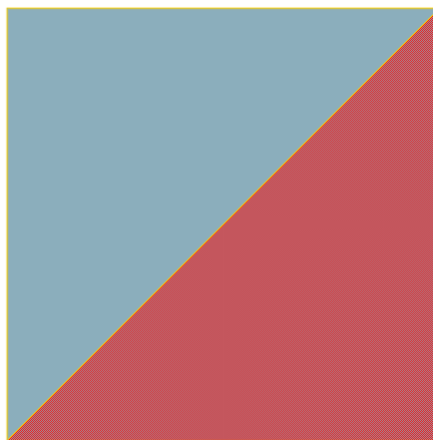


**Before split edge**

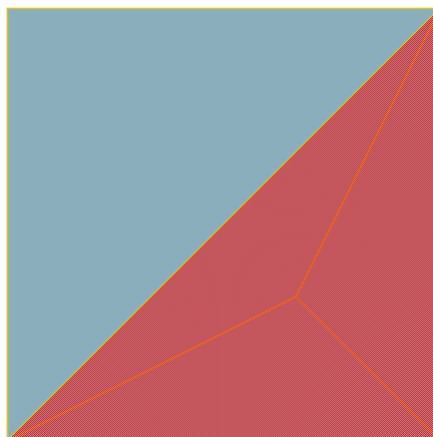




After split operation on the selected edge



Before split face



After split operation on the selected face

#### 3.3.1.1.5.4 Mesh Operations

Select the edges of a hole or gap in a 3D mesh object. Use the fill command, accessible from the right click menu, to repair the mesh. Fill can be performed on both edges and vertices.

It is useful to [simplify](#) after this process.

**Fill by Edges**

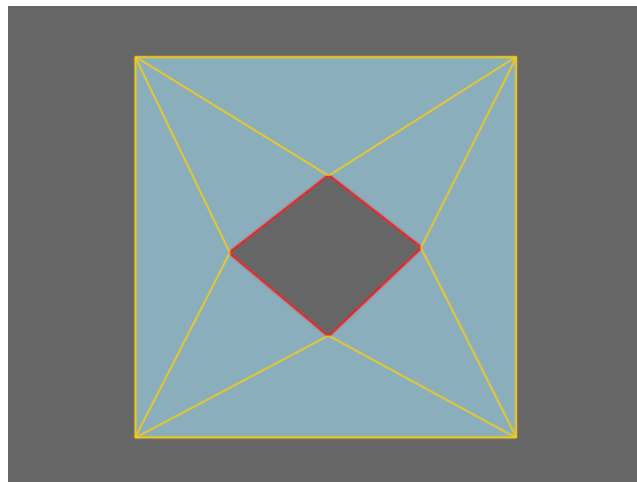
To fill a mesh using edges, select a series of connected edges outlining the hole or gap to be filled. The selected edges must be on the same plane.

Note: Any two connected edges are on the same plane. This can be used to fill in complex, non-planar gaps.

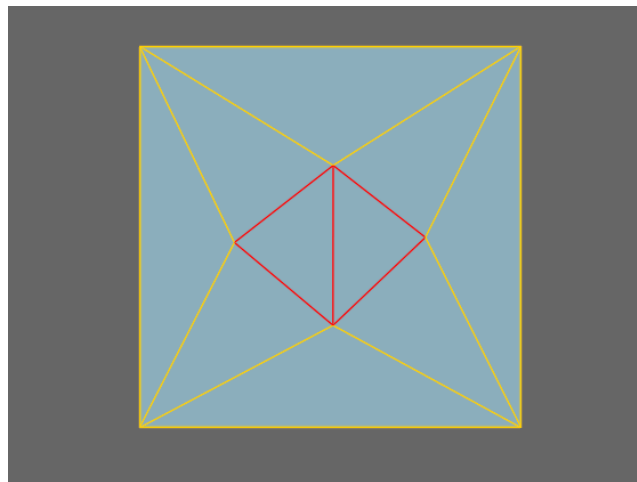
**Fill by Points**

Filling a mesh can also be done by selecting points on the outline of a hole or gap.

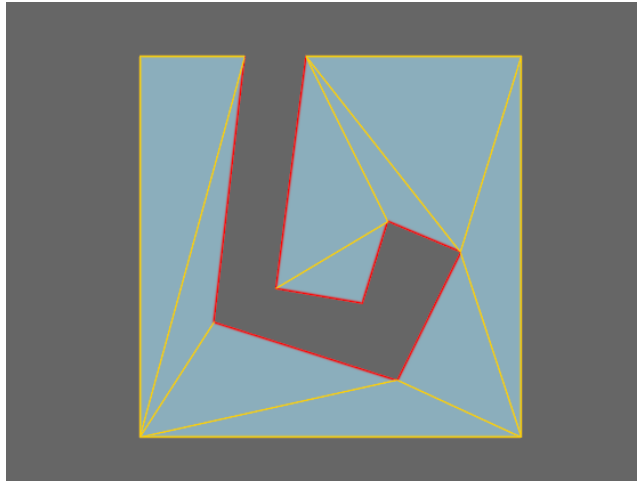
This is equivalent to selecting the edges which connect the vertices.



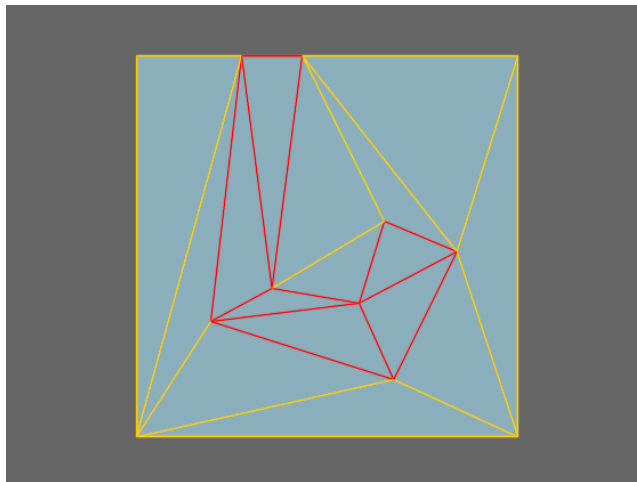
**Before**



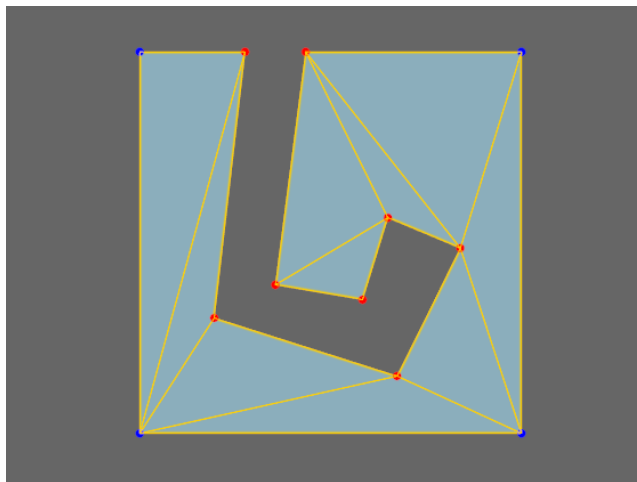
**After**



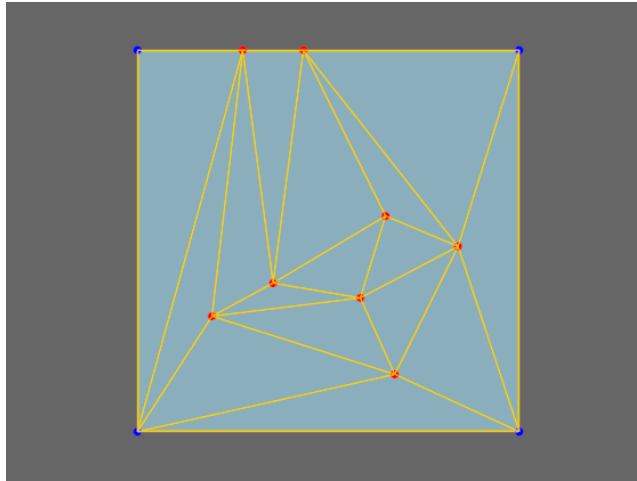
Before



After



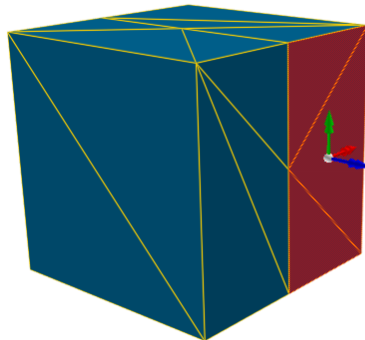
Before



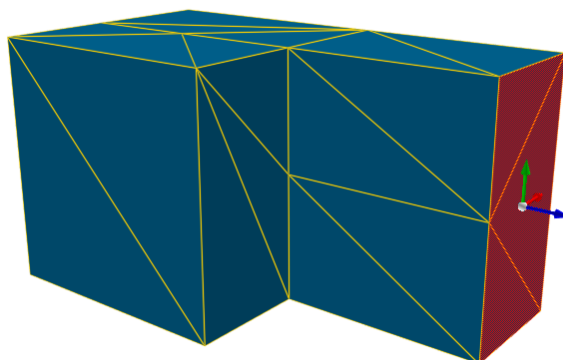
After

The grow manipulator can be used on mesh objects to extrude one or more edges or faces. In the case of [line based objects](#) like paths, end vertices can be grown to extend the line.

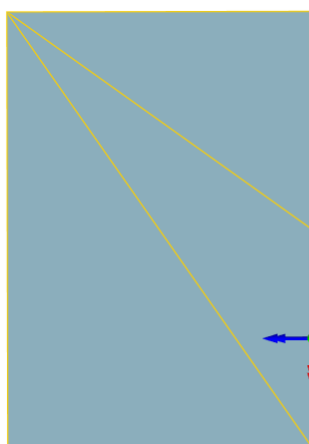
Select an edge (mesh object), or face (mesh object) and use the 'Grow' button in the [tool panel](#). The grow manipulator is shown similar to the translate manipulator. The first drag will create a duplicate of the selected component and begin a translation operation.



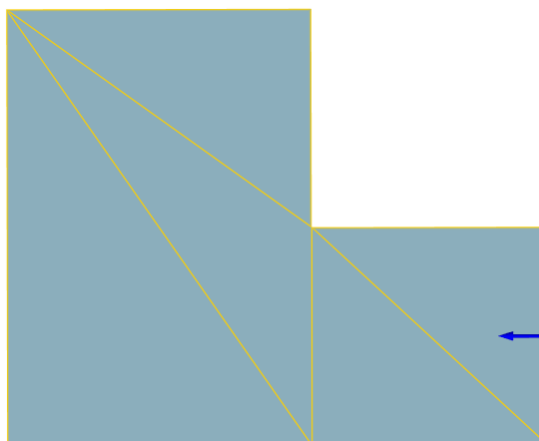
Before grow faces



**After grow operation on the selected faces**



**Before grow edge**



**After grow operation on the selected edge**

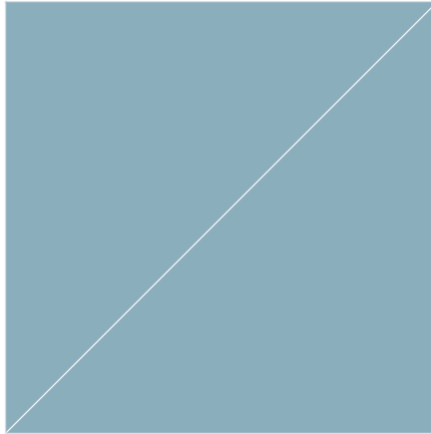
The slice tool is used to subdivide an object along a user defined line. The new edges can be useful for isolating a section of an object or can be deleted to create a hole or grown to produce an extrusion.

### **Object Slice**

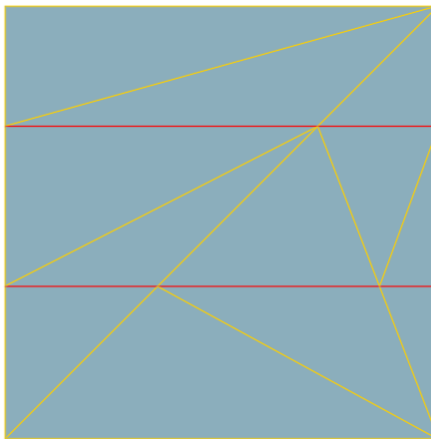
Select an object and enable the knife tool in the [tool panel](#). Drag a line across the object where the slice is to be made. The resulting slice will cut the entire object in two, extending before and after the line drawn.

**Component Slice**

It is possible to restrict a slice to a portion of an object by slicing object components. Select the desired faces or edges, enable the knife tool, then drag a line across the selection. Only the selected edges or faces will be cut.



**Before slice**



**After two horizontal knife operations on the object.**

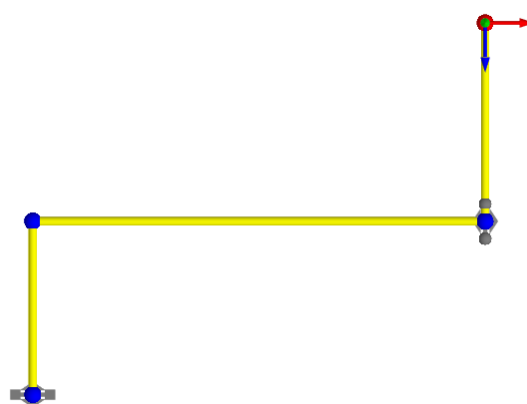
3.3.1.1.5.5 Working with Lines

Some objects such as [paths](#) and [servers](#) are represented as a series of connected line segments. The points and line segments within a line can be edited just like the edges and vertices of a 3D mesh.

To add segments to the start or end of a line, select the end vertex and use the 'Grow' (hotkey G) command to extend the line.



Before grow vertex



After grow operation on the selected vertex

### 3.3.1.2 Creating and Controlling Agents

#### 3.3.1.2.1 Scheduling & Events

[Events](#) are objects which create agents or modify the scene during a simulation. Events can be set to turn on at a specified time, or in response to changing conditions in the scene through [triggers](#).

#### Creating Agents Using Events

All agent related events will place agents in the scene at specified portal locations. Once placed in the scene, agents are given one or more tasks to accomplish. Many events give only a single task, usually to seek one portal. Some events will give a series of tasks that are to be executed in order.

Events that Create Agents	
<a href="#">Journey</a>	Create a single wave of agents, each with a single origin and a single destination, where the origins and destinations are assigned from a set.
<a href="#">Circulate</a>	Create agents which move some number of times between a set of "circulation" portals.
<a href="#">Evacuate</a>	Create agents, tell them to wait for a specified period, then evacuate the scene through one of a set of destinations.

Events that Create Agents	
<a href="#">Timetable</a>	Create agents, assign agent tasks, and open gates based on a series of coordinated text input files. Suitable for modeling train schedules, flight schedules, bus schedules, university lectures, or intersection gate timings.
<a href="#">Vehicle</a>	Create a series of vehicle arrivals. Gates can be opened at each arrival to control access to the vehicle area. Boarding and alighting agents can also be created for each arrival.

**General Events**

General events do not create agents. They modify the scene, modify existing agents, or provide information to other events.

General Events	
<a href="#">Broadcast</a>	Applies an <a href="#">action</a> to agents at a specified time.
<a href="#">Cache Change</a>	Changes the value of one or more <a href="#">tally</a> objects.
<a href="#">Gate Access</a>	Opens or closes gated <a href="#">connection objects</a> .
<a href="#">Server Access</a>	Opens or closes access to <a href="#">servers</a> .
<a href="#">Time</a>	This virtual event does nothing to the scene, but can be used by the <a href="#">reference times</a> of other events to refer to a common time.

3.3.1.2.2 Actions

While traditional origin-destination based pedestrian simulators model the flow of people from A to B, MassMotion can target a sub population mid-route and direct those individuals to accomplish any number of alternate [Tasks](#).

Actions are operators that can be selectively applied to agents as they move through the scene. Actions can be applied as an agent enters the simulation, as it transitions between links and floors, as it enters or exits zones, as it reaches destination portals, as it finishes processing at a server, or through the use of triggered broadcast events. Possible operations include changing the agent colour, giving the agent a new goal, or instructing the agent to wait for a specific interval.

For a full list of available operations, see [Agent Actions](#). For information on how to apply actions to agents, see [Where to Use Actions](#). For details on the order in which actions are applied to an agent, see [Order of Action Execution](#).

3.3.1.2.3 Tests

[Tests](#) are Boolean operations that will return true or false when applied to an agent. Tests are used throughout the scene to control how events and other control mechanisms interact with agents. The most common use for tests is in determining whether or not [actions](#) will be applied to a particular agent, but they are also used in [process chains](#) and many [events](#).



---

For a full list of where tests can be used see [Where to use Tests](#).

#### 3.3.1.2.4 Triggers

[Triggers](#) can be used to control when events turn on or off. Most events will use triggers implicitly in the setting of their start and/or end conditions. It is also possible to create standalone named trigger objects which can then be referenced by multiple events.

The most basic triggers fire at a specific time. This time can be absolute or set relative to the simulation start or a reference [Time](#).

More advanced triggers will fire when certain conditions in the scene are met. This could be based on the population in an area, a count of the number of people crossing through a door, the changing of a gate from open to closed, or any number of logical combinations of these. It is possible to have the same event fire multiple times during a simulation, once for each time the trigger conditions are met.

For a full list of available triggers, see [Triggers](#). For information on using triggers to drive events, see [Triggering Events](#).

#### 3.3.1.2.5 Process Chains

A process chain provides a mechanism for modeling the progression of agents through a series of capacity constrained stations or [servers](#). Agents can be distributed evenly across a number of servers, wait in single file for their turn at each server, be held for a specified period of time by the server, then be released when downstream servers have the available capacity to receive the agents. Process chains are ideal for designing security, ticketing, or other processes which are evaluated on their throughput efficiency and wait times.

See [Using Process Chains](#) for information on creating, configuring, and sending agents to process chains.

#### 3.3.1.2.6 Tallies

A [tally object](#) stores a numerical value that can be used to drive operational changes in the simulation. It can be configured to measure the count of agents in an area, the count of agents on a server, the combined total of other tallies, or act as an abstract cache which can be increased and decreased arbitrarily over the course of the simulation.

A tally based [trigger](#) could be used by a [gate access](#) event to open a gate when the population on a train platform reaches a certain level. A tally based [test](#) could be used to send agents to an overflow server when the population on the first server exceeds a threshold. An [action](#) could be used to increase the value of a tally cache whenever an agent holding a [token](#) enters a floor.

Tally objects can also be used to record values over the course of the simulation. These values can then be graphed or observed without the need for complicated queries.

### 3.3.1.3 Modeling Strategies

The contained sections detail strategies for modeling real world objects.

#### 3.3.1.3.1 Floors

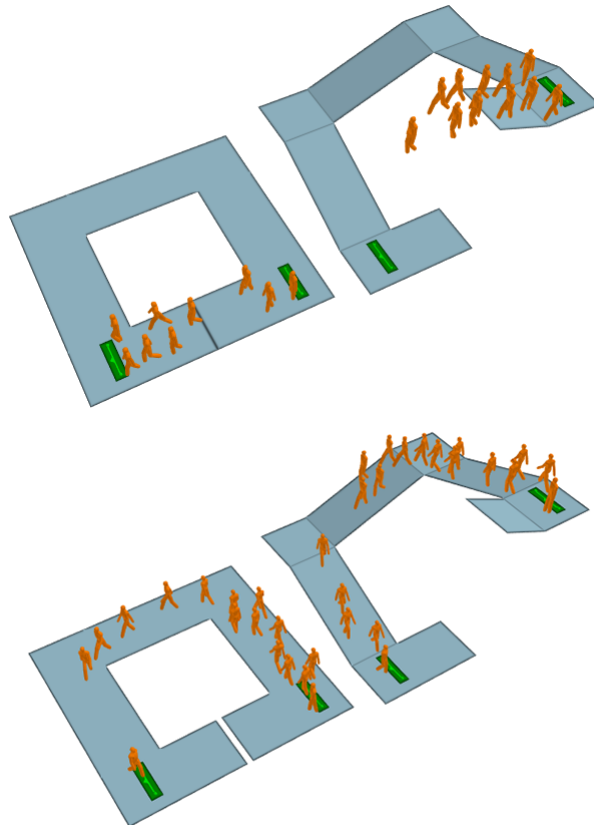
[Floors](#) represent the spaces (rooms, hallways, plazas, sidewalks, train platforms) which define the program areas of a design. Each floor defines a separate walkable area, with agent movement constrained by floor boundaries.

##### Flat vs Sloped

Floors should not be used to model changes in elevation. Agents walking unobstructed on a floor will always move at the same horizontal speed regardless of the slope of the geometry. In the images above, the agents on both the flat and sloped configurations will take the same time to reach the exit. Agents will adjust their vertical position by tracking the top surface of a walkable object, but this has no impact on their horizontal speed. It is recommended that [stairs, escalators, or ramps](#) be used to model changes in elevation.

##### Overlap

Any part of a floor that overlaps with itself will be seen by agents as one continuous space. This includes overlaps that occur at different elevations. If agents are not meant to traverse across a floor from one area to another, the two areas should be separated by a small gap.



##### Size

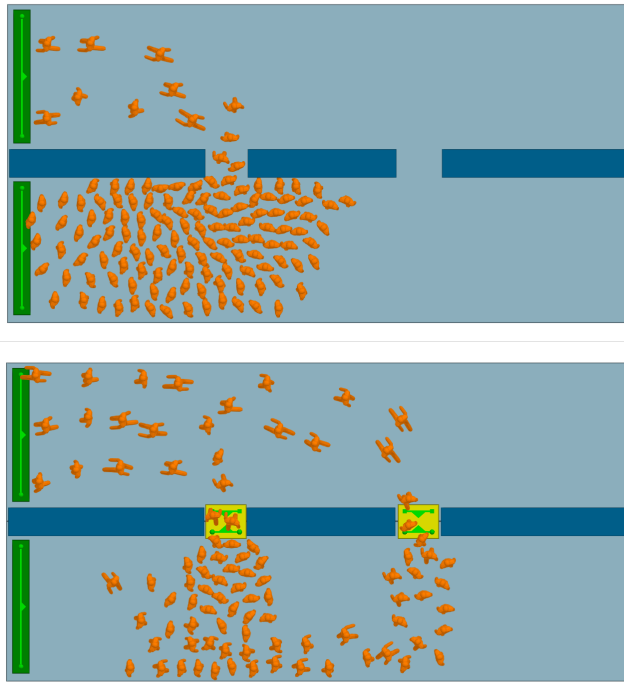
The size of floor objects is not explicitly constrained in MassMotion but large floors will consume a large amount of memory and significantly increase simulation setup time. Consider breaking very large floors into smaller areas joined together by [connection objects](#).

### 3.3.1.3.2 Connection Objects

Connection objects are used to provide access between pairs of floors. They act as decision points, providing options for agents during route selection (see [The Network](#)). They can be used as statistic collection points for [graphs](#) or [tables](#). The available connection objects are [escalators](#), [links](#), [paths](#), [ramps](#) and [stairs](#). While [elevators](#) can also be used to connect floors together, they are not considered connection objects because they are constructed differently, have different properties, and can bridge more than two floors.

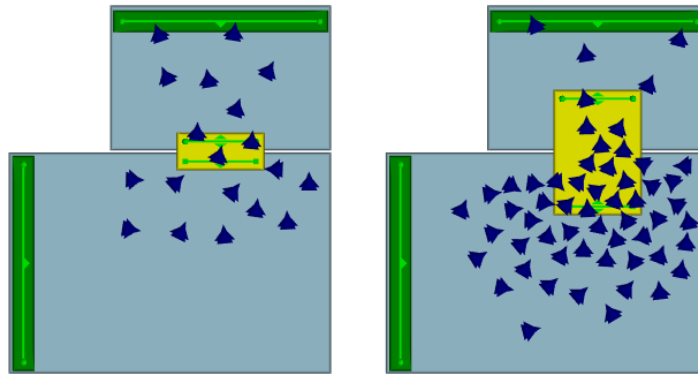
#### Route Choice

Agents on a floor



#### Placement and Properties

As described in [Connecting Objects Together](#), connection objects must be a reasonable distance above the target floor and must overlap the floor by at minimum 0.4m. The overlap distance should be minimized as large overlaps can negatively impact agent movement.



The same number of agents pass over the link with small floor overlap (left) and large floor overlap (right). The extra overlap makes it difficult for agents to smoothly transition from floor to link and will reduce flow.

All connection objects share a set of common properties:

**Direction**

Connection objects may allow agents to enter from either direction or restrict access to only a single direction.

**Gates**

All connection objects can be configured to be gated. Gated connections that are closed do not let agents enter unless opened by an [event](#). Available events which open gates are [open gate events](#) and [vehicle events](#). Agents will use the object's "Cost of waiting" to determine how a closed gate will impact their route selection. Agents that are already on the connection when the gate closes will not be prevented from exiting the object.

**Flow Limits**

Connection objects can be configured to limit the flow of agents entering the object. When demand exceeds the specific limit, agents are held at the object goal line until there is available capacity.

**Priority Flow**

Priority flow sets whether agents traversing a connection object can have priority. If agents traversing a connection object have priority, agents moving in the opposite direction will yield and wait until there are no more agents with priority.

**Delay on Enter and Exit**

Agents can be set to pause while entering or exiting a connection object.

**Banks and Perimeters**

Each connection object can be added to a single [route bank](#) and/or any number of [perimeters](#). Banks and perimeters control how the link is used by the agent when navigating the network.

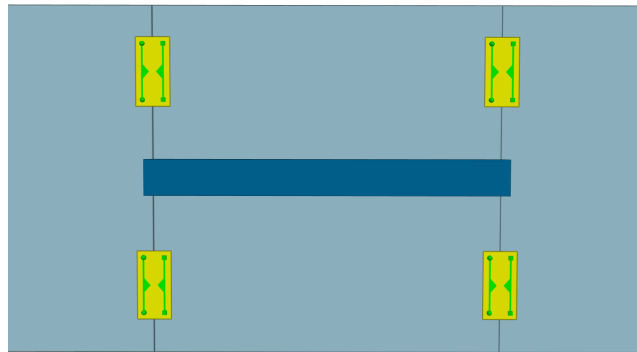
3.3.1.3.3 Complex Spaces

There are a few points to consider when determining how best to break a complex space into constituent floors and connection objects.

**Ensure Areas are Continuous**

Any space on a floor that is used by an agent must be reachable from every other used space on

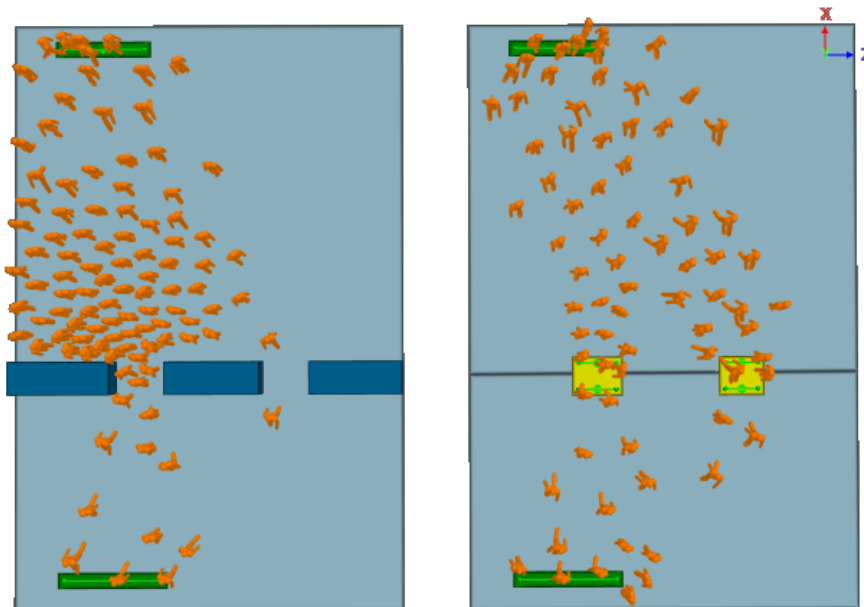
that same floor. If a barrier completely separates two areas of a floor and it is impossible for agents to move from one area to the other, that floor should be split into two separate floor objects.



**The middle floor is incorrect as it is split in two by the barrier. The floor should be broken into two separate floor objects.**

### Add Network Choice to Reduce Congestion

Large and complex floors are often broken into smaller floors in order to allow agents to make better decisions on the route they take through a space. Agents traveling across a single floor will always try and take the shortest path to their target. They will avoid other agents, but won't make any attempt to go around congestion on that floor. Agents are only aware of congestion at decision points in the route network ([connection objects](#)). In the images below, the same scenario is modeled in two different ways. In the image on the left, two doorways are modeled by placing barriers on a single floor. Most agents try and take the door on the left because that is the shortest and most direct route to the portal. In the image on the right, the floor is split into two separate floors joined by two links. Agents are still biased towards taking the shorter path across the leftmost link, but as congestion builds the increased cost of waiting at the left link means some agents will choose to go via the longer route using the link on the right.



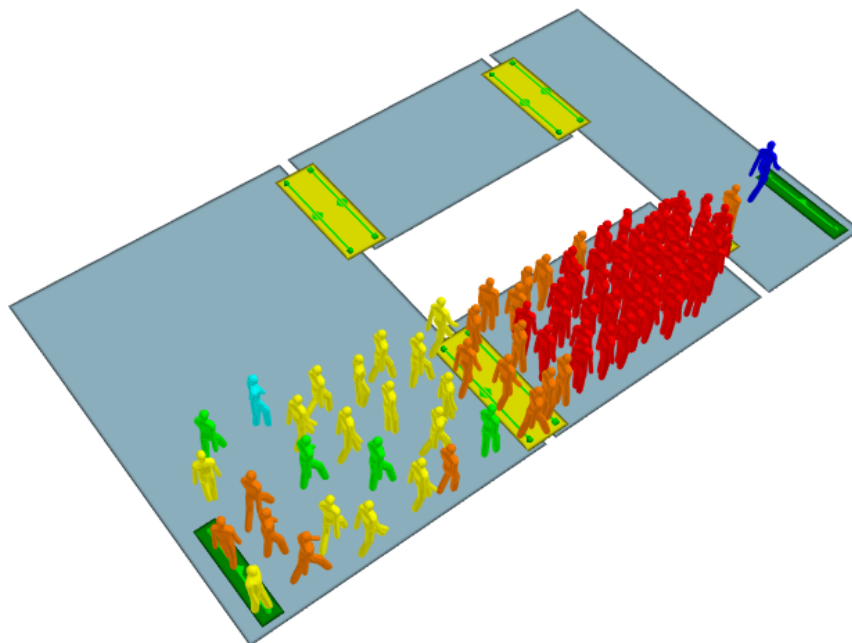
**Left: Agents are not aware of congestion at the left opening as it occurs in the middle of the floor and are not even aware of the right channel as an option. Right: Agents are aware of congestion at the left link and can choose to take the right, resulting in more balanced loading.**

This process of splitting floors to ensure agents fully consider route implications is useful in a number of situations including:

- Narrowing due to barriers (as above)
- Bifurcation of routes due to a barrier or void in a floor (as with stairs/escalators in the middle of train platforms)
- Heavy flows compressing around an obstruction despite available space (as with jogs in corridor alignments)

**Reduce Network Choice to Reduce Congestion**

While splitting floors and introducing additional links can increase the amount of choice in a network, there are counterbalancing factors to consider. Agents are only able to perceive congestion associated with links that are connected to their current floor. Overuse of links can serve to hide congestion from agents leading to poor route choice behaviour. In the image below it can be seen that agents will continue to select the nearest route despite the fact that the narrow link in the connecting corridor has become congested and the alternate route has become more efficient. This should be considered when deciding when and where to split up complex spaces.



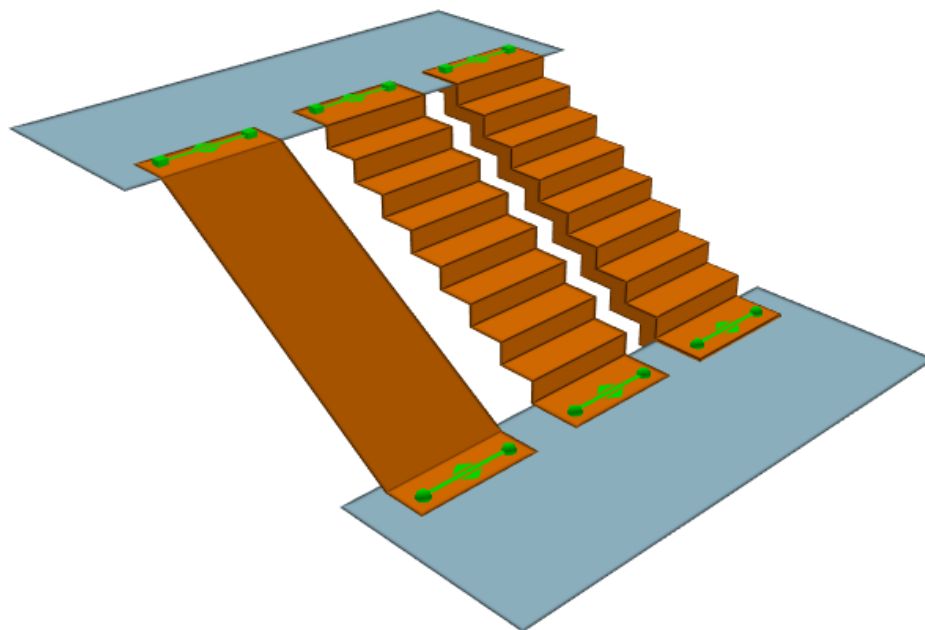
3.3.1.3.4 Vertical Circulation

Stairs, ramps, and escalators (vertical circulation elements or VCEs) are link objects that connect floors of different heights using a sloped vertical transition. They have much the same best practice guidelines that are described in the [Connection Objects](#) guide and in the [Connecting Objects Together](#) section.

The slope of VCEs must be between 0 and 50 degrees from level. Steeper slopes may result in agents not being able to track the surface during simulation

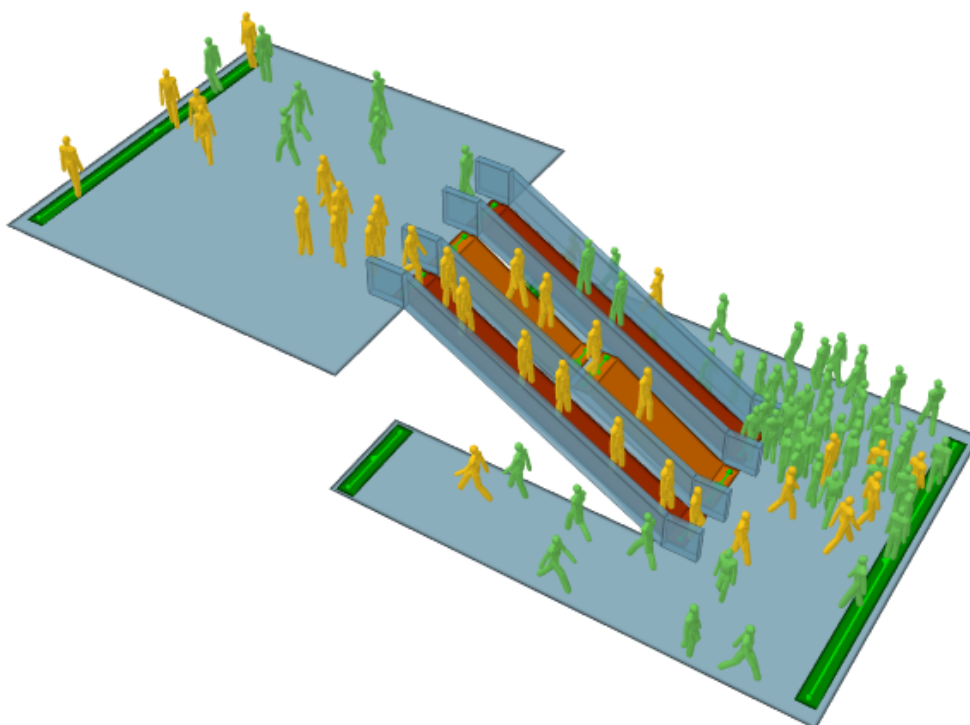
**Geometry**

The geometry of VCEs can be as detailed as the user likes. All the stair objects in the image below are functionally equivalent and the agents will traverse them in the same way. As with [floors](#), the geometry may not overlap itself. The slope must be between 0 and 50 degrees from level as agents will have trouble tracking surfaces at a larger grade.



### Common Configurations

Escalators and stairs are often constructed in sets with one or more of each type located in close proximity to one another. The image below shows a configuration common in transit stations where a stair and a set of down/up escalators are arranged in the middle of a platform. As in the real world, the agents will perceive the escalators to have a greater utility than the stairs and queues will form for the escalators before there is substantial traffic on the stairs.



**Banks of Vertical Circulation Elements**

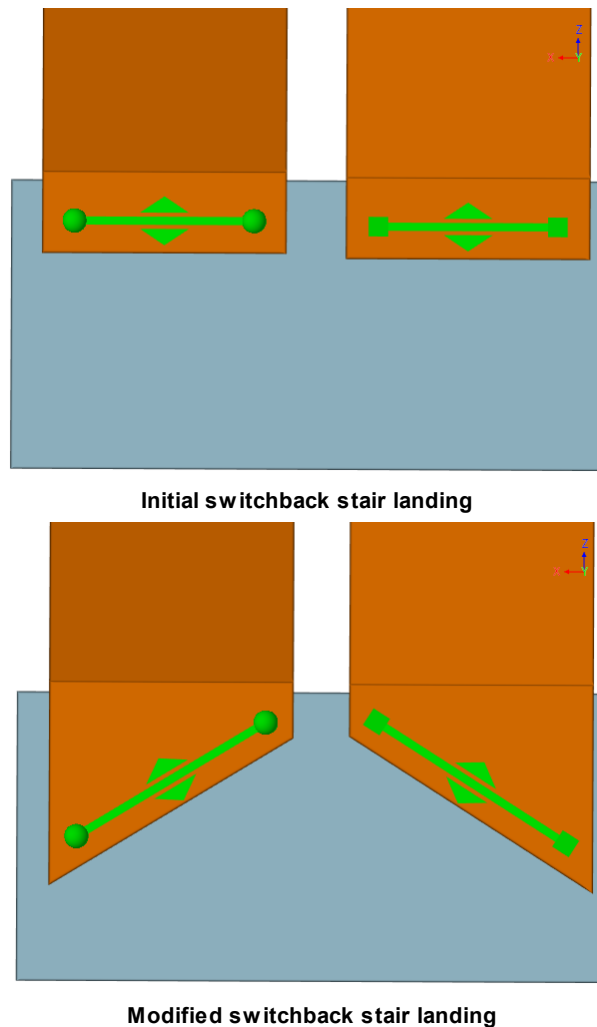
When multiple stairs, escalators, or ramps are positioned beside one another, have the same directionality, and bridge the same two floors, they should be banked to ensure agents make use of all available elements. In Figure 2, the two escalators are in opposite directions, while the stair is bidirectional. Because an object can only be part of one bank, it is not possible to bank the stair with both escalators. A single bank should be created for the stair and whichever escalator is likely to see the most traffic during the simulation. Please see [Banks](#) for more information.

3.3.1.3.4.1 Switchback Stairs

Switchback stairs such as those commonly found in the evacuation cores of buildings present a set of unique challenges. These stairs are generally narrow, with constrained landing areas where agents must make abrupt changes in direction.

**Geometry**

If flow rates on switchback stairs are not at the desired level, angle the ends of the stair landings to help guide agents in a smooth transition from one stair to the other.



Note that if goal lines are placed at too great an angle to one another, the outside corners of the landings will present such acute angles that agents can have trouble squeezing onto the landings along the outside edges. Moderate angles like those demonstrated above are recommended.



### Properties

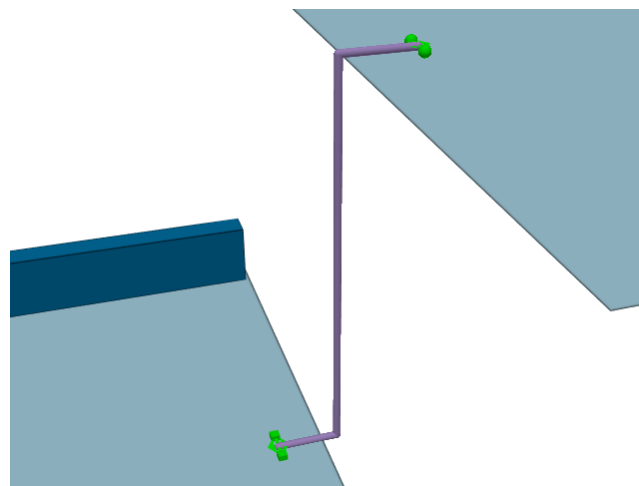
Given a demand at or below level of service D, switchback stairs process agents at a rate roughly equal to similarly sized straight stairs. For level of service E or F, there can be a 10% drop in processing rates despite the geometric adjustments described above. To achieve the same rates as straight stairs even in high density situations, set the switchback stair property for agent body radius to a value of 0.2m.

#### 3.3.1.3.4.2 Ladders

A ladder can be modelled using a [path](#) object.

### Geometry

Ensure that the final line segments on either end of the path are flat.



A ladder modelled using a path.

### Properties

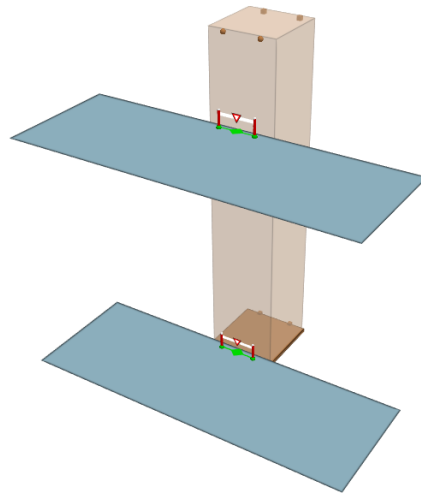
If the ladder is only to be used in one direction, set the path to the desired direction. If two-way travel is required, enable priority access. If there is a clear priority direction, set the priority direction appropriately and ensure that the option 'Primary will yield' is checked so that agents will only use the path in one direction at a time.

#### 3.3.1.3.4.3 Elevators

Elevators or lifts can be modeled using the built in [elevator](#) object. A single elevator can carry agents between two or more connected floors. Elevators can be configured to respond to agent calls or set to move automatically between floors.

Elevators that service similar floors can be grouped together using an [elevator bank](#). The bank will coordinate elevator movement and determine which elevator to dispatch to a particular call.

Elevators are part of the overall [scene network](#), bridging between floors just like other [connection objects](#). Agents will consider routes that include elevators as they would any other route.



An elevator

### 3.3.1.3.5 Turnstiles

MassMotion is used extensively for analyzing transit stations, and the modelling of turnstile geometry and related agent behaviour is a common requirement.

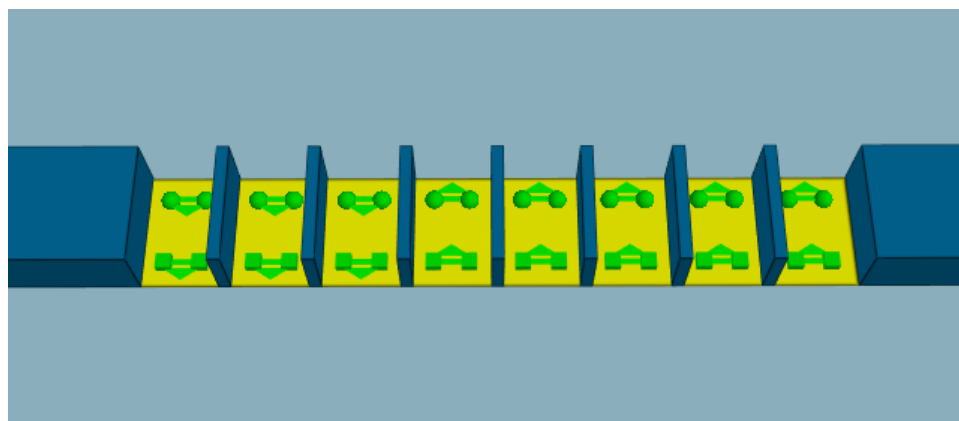
Turnstiles can be modeled either as a [series of links](#) or a [series of paths](#). A Link looks more like a natural approximation of a turnstile, but when turnstiles are exceptionally narrow (less than 0.6m), are not aligned with the scene axes, or require higher sustained flow rates, paths can produce better results.

#### 3.3.1.3.5.1 Link Turnstiles

##### Link Geometry

The geometry of the turnstiles will be similar to that of standard doors made from [links](#). In order to achieve the desired flow rates across the links, consider the following steps:

1. Determine the centre to centre dimension of the real world turnstiles and use that for the width of the link.
2. Make the width of the separating barrier geometry as narrow as is practically possible.
3. At the end conditions, extend the width of the barrier geometry to prevent agents from ending up beside their target links.



A set of 0.68m wide turnstiles with barriers 0.1m wide

### Link Properties

If the turnstiles are to be used in one direction only, set that direction in the turnstile property window. If the turnstiles are to be bidirectional, enable priority access and set the priority direction to bidirectional. In this case it is recommended that priority 'move aside' be turned off so that agents do not move to block one turnstile when waiting for access to another.

The limit flow property can be used to ensure the processing rate does not exceed operational expectations. An additional delay on exit can be used to simulate the brief pause from dealing with fares or navigating the turnstile. Delay on exit is recommended over delay on enter so that paused agents do not interfere with the limit flow control over inbound flow rates.

If the turnstile widths are exceptionally narrow (less than 0.6m), if the turnstile is not aligned with the scene axes, or if the links are not producing the desired flow rates, consider modeling the turnstiles [using paths](#). Alternatively, it may be help to do the following:

1. For each turnstile object, set the traversal type property (on the agent behaviour tab of the link's property window) to ignore barriers. This will free up space for the agents along the edges of the link surface.
2. For each floor on either side, set the physical map resolution to 0.05m or even 0.02m. This will enhance the resolution of the edge/barrier condition on the approach to the turnstile channels and provide more effective width within the approach channel.
3. Reduce the agent radius for each turnstile. MassMotion agents are represented as cylinders and are unable to move sideways to fit through narrow gaps as people can in real life. Reducing agent radius can serve to approximate 'squeezing' through narrow openings.

### Banks of Turnstiles

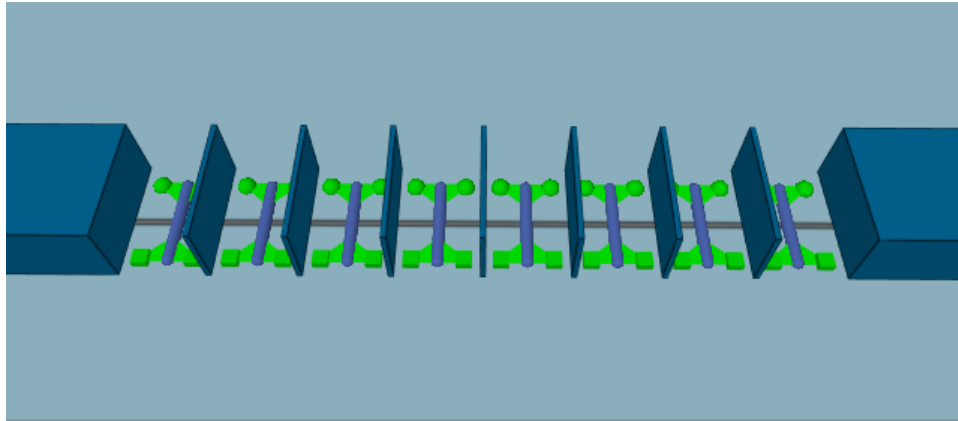
Turnstiles are often positioned in groups bridging the same two floors. To ensure that agents use all available turnstiles and consider all turnstiles equally, turnstiles in the same direction should be banked. Please see [route banks](#) for more information.

#### 3.3.1.3.5.2 Path Turnstiles

### Path Geometry

The geometry of each turnstile will be that of a simple short [path](#). In order to achieve the desired flow rates across the paths, consider the following steps:

1. Determine the centre to centre dimension of the real world turnstiles and use that for the placement of the paths.
2. Make the width of the separating barrier geometry as narrow as is practically possible.
3. At the end conditions, extend the width of the barrier geometry to prevent agents from ending up beside their target paths.



A set of 0.68m wide turnstiles with barriers 0.1m wide

### Path Properties

If the turnstiles are to be used in one direction only, set that direction in the turnstile property window. If the turnstiles are to be bidirectional, enable priority access and set the priority direction to bidirectional. In this case it is recommended that priority 'move aside' be turned off so that agents do not move to block one turnstile when waiting for access to another.

The limit flow property can be used to ensure the processing rate does not exceed operational expectations. An additional delay on exit can be used to simulate the brief pause from dealing with fares or navigating the turnstile. Delay on exit is recommended over delay on enter so that paused agents do not interfere with the limit flow control over inbound flow rates.

If high flow rates are required it may be necessary to also do the following:

1. For each floor on either side, set the physical map resolution to 0.05m or even 0.02m. This will enhance the resolution of the edge/barrier condition on the approach to the turnstile channels and provide more effective width within the approach channel.
2. For each path, set the speed density relationship to "Unconstrained" (available in the agent behaviour tab). This ensures that agents will board and traverse the path at full speed and is necessary to achieve higher flow rates across the path.

### Banks of Turnstiles

Turnstiles are often positioned in groups bridging the same two floors. To ensure that agents use all available turnstiles and consider all turnstiles equally, turnstiles in the same direction should be banked. Please see [Route Banks](#) for more information.

#### 3.3.1.3.6 Collections

A [collection](#) is a group of one or more objects. Some collections, such as banks, perimeters, and zones, have a particular function within the simulation. All collections can be used to help manage a scene or perform [analysis](#).

### Visibility

When a collection is shown or hidden, all members are shown or hidden. This can be useful for quickly controlling which elements of a scene are visible. In an office tower, there might be one collection for each floor or level. Showing/hiding levels then is just a matter of showing/hiding the appropriate collections.

### Intelligent Member Selection

A collection can be specified as an input to many objects (for instance the entrance portals in a journey event). The object will pull only those members from the collection that make sense in the

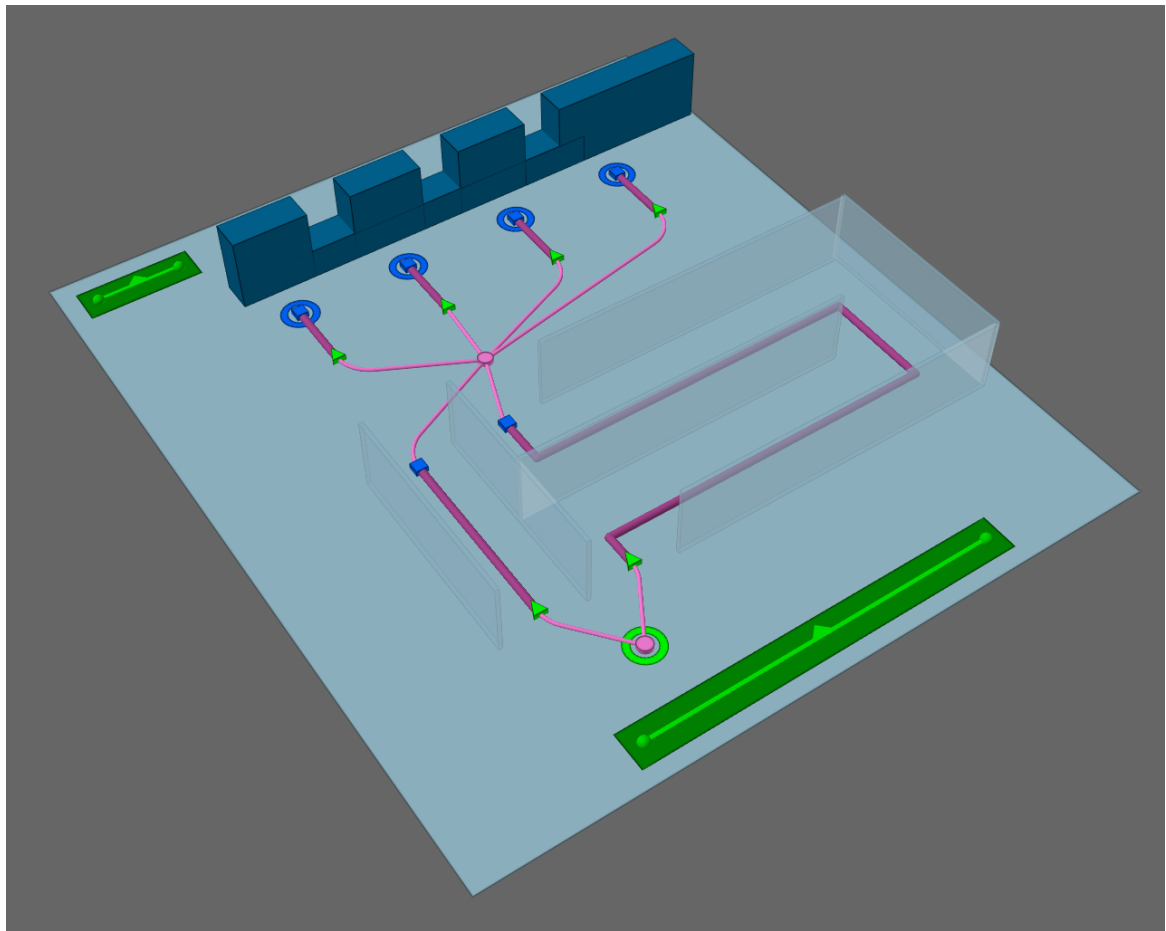
particular context. If a single collection contains all of the floors, door links, and portals associated with a train car, that same collection can be used to specify the gates in a gate event and the portal origins in a journey event.

Please see [Collections](#) for more information on referencing collections from other objects.

Collection Type	
<b>Bank (Route)</b>	A route bank is a collection of <a href="#">connection objects</a> (i.e. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">ramps</a> , <a href="#">escalators</a> and <a href="#">paths</a> ) that bridge the same two <a href="#">floors</a> and are spatially close to one another. When an agent chooses a route through a bank, it selects which of the bank members to take based only on near distance and queue cost, ignoring any downstream costs. This can be useful for maximizing flow through a set of turnstiles. See <a href="#">Route Bank</a> for more information.
<b>Bank (Elevator)</b>	An elevator bank coordinates the movement of member elevators to ensure maximum efficiency when responding to calls. See <a href="#">Elevator Bank</a> for more information.
<b>Collection</b>	A general purpose collection with no special purpose beyond the grouping of member objects. See <a href="#">Collection</a> for more information.
<b>Perimeter</b>	A perimeter is a collection of <a href="#">connection objects</a> (ie. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">ramps</a> , <a href="#">escalators</a> and <a href="#">paths</a> ) marking a boundary around a special area. Routes that cross this boundary more than once are disallowed. By adding all turnstile links to a perimeter, routes leading into and out of the fair paid area will be permitted, but through-traffic agents will be prevented from taking shortcut through the fair paid area. See <a href="#">Perimeter</a> for more information.
<b>Zone</b>	A zone is a collection of floors, links, stairs, ramps, escalators, or paths. Zones are used primarily for evacuations where agents inside a zone can be instructed to prioritize leaving the zone before heading to any other destination. This is useful in scenarios where agents evacuating a subway station must first evacuate the platform before worrying about evacuating the station. See <a href="#">Zone</a> for more information.

3.3.1.3.7 Ticket Desks

The definition of ticket desks or similar processes is a common task in MassMotion. Pictured below is a standard multi-fare airline-style ticketing/check-in configuration:



Ticketing Layout

Ticketing layouts are constructed using [servers](#) that are [chained together](#) to represent the flow of people from one process component to the next. In the [above configuration](#) there are two accumulator queues for business and economy passengers which both feed into 4 service positions. The two accumulator queue entrance points should be connected together such that agents are sent to a single dispatch. The dispatch can then be configured to use different techniques to sort agents between the two queues.

It is good practice to setup [barriers](#) around the queues (as in the real world) to ensure that circulating people don't try to pass through the queuing areas.

**Properties**

Agents are assumed to have either a business or economy [token](#) assigned through earlier [actions](#). To manage access control, the server objects representing the accumulator queues are set to require the appropriate token, for example, business tokens for the business class queue.

The accumulator queues should be set to infinite capacity and contact times of zero. This will allow agents to continue queuing regardless of occupancy and they will only stop at the queue exit points if the downstream service positions are all busy. This is an automatic consequence of the connection between both the exit points of the accumulator servers and the entry points to the service position servers.

The capacity of the service position servers should be set to 1 to ensure that once the position is occupied, no other agents will be released from the accumulator queues until the position becomes free again. The contact time for the service positions should be set to correspond to the range of times that it typically takes to process a passenger. The service positions may be set to prefer business or economy tokens which means the position will prioritize the release of agents with the designated token from the accumulator queues. If no preferred token agents are queuing the positions will then accept other agent types.

### 3.3.2 Simulation

Once authoring has reached a stage where the scene contains a viable network and there are events for generating agents, the project can be simulated. See [running a simulation](#) for information on how to launch a simulation. See [simulation data](#) for information on the data produced by a simulation.

#### Validation

When a simulation is launched, the project is automatically validated. Validation involves verifying that all objects and their properties are in a consistent state, that all object references can be resolved, and that the resulting simulation network will be well formed. Any errors must be addressed before proceeding with a simulation.

Validation can also be performed manually using the Validate button in the analysis tab of the main window's ribbon.

#### Compiling

When starting a simulation, a copy of the project is created and compiled into an optimized form. It is the optimized copy which is used for the simulation, meaning that changes in the authoring environment have no impact on a simulation that is already running.

Compilation involves converting the various object properties into raw data, resolving references between different objects, generating obstacle and approach maps for walkable objects, converting object goal lines into network waypoints, building the network, constructing cost trees for each destination in the scene, and creating a new database file.

#### Execution

The simulation can be run either in console mode or debug mode (see [Running a Simulation](#)). The [console window](#) provides text based feedback on the simulation. The [debug window](#) contains a graphical scene view and provides a mechanism for interrogating agents and objects as the simulation progresses. Both methods make use of all available processors on the machine, but a console simulation will execute much faster than a debug.

#### Review

Once a simulation is complete, the results can be played back or analysed through queries using the [Analysis](#) system.

#### 3.3.2.1 Simulation Data

Each iteration of a simulation is associated with a [simulation run](#) object in the project. The simulation run is connected to a [database file](#) on disk and is the handle through which MassMotion accesses the data for both playback and analysis.

All of the simulation data is stored in a single mmdb file. This is a standard SQLite database file and can be interrogated using any third party SQLite tool. Many of the [analysis queries](#) provided by MassMotion are convenience wrappers for the execution of raw SQL queries. Agent position

information is packed into an optimized form and difficult to read directly, but all other information can be extracted directly from the database with a basic understanding of SQL. For example, tables exist in this database to indicate what floors agents were on at every instant, and what tokens they were holding. A separate tool for [extracting and exporting agent positions](#) is available from the main window.

#### **Local Storage Recommended**

The performance of simulation execution, playback, and analysis queries can all be negatively affected by slow database access times. As a result, it is recommended that simulation runs point only to local drives and not a USB or network drive. SSD drives give the best performance, but note that the results database for a large model can be 100 GB or more. A rule of thumb is that 1 agent for 1 hour will take approximately 1 MB; therefore, a 2 hour simulation with a sustained population of 10 000 agents will result in a database approximately 20 GB in size.

#### **Recovering a Project**

The database file referenced by the simulation run contains not just the results of the simulation, but an embedded copy of the project used to generate the simulation. The original project can be recovered by opening the mmdb database file as if it were a regular project file (\*.mm) using the Open button in the main window's project ribbon.

### **3.3.3 Analysis**

A completed simulation can be analysed and communicated using a number of different systems and tools.

#### **Playback**

Agent movement during the simulation can be reviewed using the the 3D view and the time panel. See [Playback](#) for more information.

#### **Queries**

Analysis queries take the form of [Graphs](#), [Tables](#), or [Maps](#). Graphs display information such as population counts or flow counts over time. Tables provide summary information about a particular population of agents. Maps represent spatial information such as density by painting colours directly onto the 3D scene objects. Queries can be customized for a particular project, saved, and re-applied to additional runs.

#### **Agent Filters**

[Agent filters](#) allow for further customization of simulation run playback or queries by separating out specific sub-populations of agents. Agent filters can isolate agents that entered at a particular portal, or who are on a specified floor, or who have ever crossed a certain link. The filters can then be fed into maps, graphs, tables, or general playback to customize the query or playback output.

#### **Reporting**

Simulation run data can be exported to text files, images, or videos. See [Reporting](#) for more information.

#### **Presenting**

Simulation results can be exported for playback in the free Viewer application. See [working with the viewer](#) for a description of how to export results and make the most of the viewer.



### 3.3.3.1 Playback

A simulation can be reviewed in the scene view through the main window's [playback controls](#). The run will populate the view with agents as they were at the specified time in the recorded simulation.

Each simulation run in the project will play back a separate population of agents. When all simulation runs are shown, the different populations are presented over top of one another. This can be very useful when comparing populations from runs with slightly different setups. To choose a single simulation run for playback, use the source button at the top of the [3D scene view](#). Hidden simulation runs can still be used to drive queries but will not have any impact on the visual scene during playback.

Gate state is only displayed when a single simulation run is visible. If multiple simulation runs are showing, then gates will be drawn in white to indicate possible conflicting sources.

The colour of agents is taken from the colouring in the simulation. The colouring can be changed through the properties of the [simulation run](#) object. Agents can be set to a single colour in order to distinguish the populations of one run from the population of another, or the agents can be coloured by density. It is also possible to use an agent filter to colour a sub population.

The current time and count of agents shown in the scene can be displayed in the [3D scene view](#) by enabling the appropriate scene view overlay.

### 3.3.3.2 Reporting

Data can be exported from MassMotion in a number of different ways:

#### Graph and Table Data

When a table or graph query has been created and evaluated, the resulting data can be exported to a text CSV file using the 'File' button in the graph or table properties window.

#### Graph Images

When a graph query has been created and evaluated, the resulting data can be exported to an image file using the 'File' button in the graph properties window. Use the style options in the property window to control general graph formatting.

#### Scene Images and Videos

The [Movie and Image Export](#) window can be used to generate both snapshot images of the 3D scene view and videos of simulation playback. The export window is available from the 'Analysis' tab of the [Main Window](#) ribbon.

#### Agent Positions

The [Agent Position Export](#) window can be used to export agent positions to a CSV text file. The export window is available from the 'Analysis' tab of the [Main Window](#) ribbon.

#### Alembic

The [Alembic Export](#) window can be used to export agent playback information to alembic files. These files can be loaded into 3rd party applications like Softimage or 3ds Max to produce high quality renderings of the simulation.

#### Viewer

The playback of a simulation can be exported for use in the free Viewer application. Bookmarks can be used to highlights areas or times during the playback. See [working with the viewer](#) for more information.

## 3.4 Troubleshooting

A complex project can encounter issues in setup or design. MassMotion provides a number of tools to help identify and resolve authoring related issues.

### 3.4.1 Auditing

Auditing an object will identify all properties that have non default values. This can sometimes uncover properties that were unintentionally modified or values that are incorrect. Audit results can be presented either by object or by property. Object auditing is available through an object's right-click menu. See the [Issue Window](#) for information on how to review any resulting issues.

### 3.4.2 Validation

Validation verifies general project integrity. Invalid property values in objects will be reported. Missing or incorrect references between objects will be flagged. All issues are collected and presented using the [Issue Window](#).

Full simulation validation is automatically executed when launching a new simulation. The project can also be manually validated using the Validate button in the Simulation & Analysis tab of the main window ribbon. Individual objects can be validated through the object's right-click menu.

Validation Types	
<b>Quick Simulation</b>	Examine all simulation objects and check properties for missing or incorrect values. This will also do simple tests to ensure basic route network connectivity.
<b>Full Simulation</b>	Perform a quick simulation validation, then construct the simulation network in full and report on any errors. This will also examine the relationship between objects and flag any circular or incorrect references.
<b>Analysis</b>	Examine all analysis objects and check properties for missing or incorrect values.
<b>Entire Project</b>	Same as a full simulation and analysis validation.
<b>Object (right-click)</b>	Perform a quick validation on the selection object(s), checking properties for missing or incorrect values.

### 3.4.3 Observing Agents or Objects

The observer windows display information about a particular agent or object for a given simulation run.

#### Observing Agents

The [Agent Observer](#) window can be used to interrogate the history of an agent from a particular simulation run. It displays lifetime information such as the starting portal, ending portal, creation event, route taken, and action history. It also displays some dynamic information such as speed, state, and density.

A focused observer window is available through the right-click menu on a playback agent. To open a generic agent observer window and enter the agent ID manually, use the 'View -> Window' menu or

right-click and 'Observe' a simulation run object.

### Observing Events

The [Event Observer](#) window can be used to determine when an event was active in a simulation. It will display the time at which an event became active and the type of activity it was engaged in.

### Observing Tallies

The [Tally Observer](#) window will display the value of one or more tally objects over the course of the simulation.

## 3.4.4 Finding Object References

The 'Find' command in an object's right-click menu will search a project for objects that relate to the selected object(s). For example it is possible to find all members of a collection, find all events that reference a particular gate, find the reference geometry used to generate a stair, or find all actions that use a particular token. This can be useful when determining why a gate is opening at an unexpected time or which events are creating agents at a given portal.

## 3.4.5 Debugging a Simulation

[Running a simulation](#) in debug mode provides more than just a graphical view of what is going on. It also gives access to many of the low level properties used by agents in their decision making. It can reveal why agents are being deleted or making unexpected decisions.

In the simulation launch dialog, specify a breakpoint time to have the debug simulation automatically pause when it reaches that time.

### Object Properties

The right panel of the [Debug Simulation Window](#) shows the compiled properties of the selected object. See [Object Properties](#) for a detailed list of these properties. Agent properties are particularly useful when tracking down why an agent is being deleted or not behaving as expected. The task panel shows the list of tasks in the agent's to-do list, including the current active task. The action panel shows a history of all actions that have been applied to an agent and summarizes the results.

### Agent Display Options

Enable specialized display options on a per agent basis to gain a better understanding of why an agent is behaving in a particular manner. The display options are available from the right-click menu for an agent in the debug scene view. 'Route Costing' will display perceived costs for the various route options the agent is considering. 'Neighbourhood' will indicate the neighbours that an agent is aware of. 'Surface Probe' will show the direction in which the agent believes it should move to reach its target. 'Social Forces' draws arrows representing the various forces acting on the agent. These options are described in detail in the [Simulation Scene View](#).

### Obstacle and Approach Maps

Enable the display of an object's obstacle map to visualize the walkable space on a floor, link, stair, ramp, or escalator. Areas marked in red are unavailable to agents either because they are off the floor or covered by a nearby barrier. Sometimes barriers that don't look like they should be having an impact on an area will still cut a link in two or block access to a stair. Use an object's right-click menu to display the obstacle map.

### Dumping Surface Maps

In the [project settings](#), enable the dumping of surface maps. When the simulation is compiled, all approach maps and obstacle maps will be written out as images to a debug folder in the simulation run results folder. Examine images for those objects that are involved in a problem and verify that goal lines appear as expected and that there aren't any unexpected barriers interfering with walkable

space.

### 3.4.6 Using Analysis to Diagnose Issues

The [Analysis](#) system of queries and agent filters can be very useful for verifying the integrity of a project or tracking down the cause of any problems.

The [Origin/Destination](#) table is useful for quickly making sure the expected number of agents are entering and leaving at the expected places. The [Agent Path](#) map will trace the route taken by a set of agents. The [Agent Summary](#) table can indicate which agents failed to exit the simulation cleanly.

When diagnosing issues with agent route choice, the [Static Cost](#) map can help understand cost or distance gradients throughout the scene. Unintentional distance penalties on links, forgotten one way stairs, or accidentally virtual floors will all present as discontinuities in the map.

The 'Has end state' [Agent Filter](#) can be useful for isolating those agents that were deleted with errors, or that unexpectedly remained in the simulation after completion. Use the filter within a [Simulation Run](#) to only show those agents that have been deleted, or use in combination with an [Agent Path](#) to examine the path problem agents took through the scene.

The 'Selected agents' [Agent Filter](#) can also be useful for limiting a query to only the selected agents.

### 3.4.7 Auto Save

MassMotion will periodically save the current project to a temporary folder. These project files are automatically deleted when MassMotion exits. If MassMotion is still running or stops unexpectedly due to a crash, the project files remain in the folder and can be opened like any other project. Auto save can be disabled or configured through the [application preferences](#).

On Microsoft Windows the default location of the auto save folder is:

```
C:\Users\<%USER%>\AppData\Local\Temp\Oasys\MassMotion
```

### 3.4.8 Graphics

MassMotion requires OpenGL 3.1 or greater. When the graphics card does not support advanced graphics the 3D scene view will present as empty black.

In order to resolve the issue:

- Verify that the installed graphics card supports OpenGL 3.1.
- Update all graphics drivers.
- Run the "Graphics Driver Diagnostic" command from the Help menu and note any errors.
- On some laptops or other systems with both a discrete GPU and an integrated GPU it may be necessary to create a custom profile for MassMotion in order to insure MassMotion runs using the more powerful discrete GPU.

# Part IV

---

## 4 Reference

The reference section provides details about the individual components of MassMotion. It is intended to work in concert with the [User Guide](#). The core of the reference section is in [Objects](#), which contains detailed descriptions of the various properties of both objects that are part of the simulation (such as floors and schedules) and analyses that are used to query the results of the simulation.

### 4.1 Project

#### 4.1.1 Opening Projects

It is possible to open projects from the following file types

File Type	Description
.mm	The standard MassMotion project file. Contains all <a href="#">objects</a> and the <a href="#">project settings</a> .
.mmdb	A MassMotion or Flow database file produced by running a simulation. Each database file contains a complete record of the project used to create the simulation. On open, all previous simulation runs are removed and a new simulation run is created and connected to the mmdb database.
.mmxsi	A project exported from the Softimage workbench using MassMotion 7.

When opening projects saved or generated using previous version of MassMotion, the project is [automatically updated](#) to the latest version.

##### 4.1.1.1 Merging Projects

MassMotion includes functionality for merging data from another project file into the currently open project. This will attempt to take all [Objects](#) and the [Project Settings](#) from the file and import them into the current project. A dialog window will be shown to indicate what data will be merged and allow for the resolution of any conflicts.

Merge can be found under the File menu.

#### Objects

- Any objects that exist in the file but not in the current project will be added to the current project as a new object. If file objects happen to have the same name as objects in the current project, the file objects will be renamed by adding a numerical suffix.
- If the same object exists in both the file and current project, and both versions are equal (same name, geometry and properties), the file object is ignored.
- If the same object exists in both the file and current project, and the versions are different, then a conflict is reported.

#### Project Settings

If the project settings from the imported project are different in any way from the current project, they will be marked as in conflict and must be resolved.

#### Resolving Conflicts

There are three options for dealing with object or settings conflicts:

- **Use existing** will use the version from the current project and ignore the file.
- **Use imported** will use the version from the file, overwriting the current project.
- **Advanced** allows the user to choose between 'Use existing' and 'Use imported' for settings and

conflicted objects on an individual basis.

#### 4.1.1.2 Upgrading Older Projects

Projects saved using an older version of MassMotion will automatically be upgraded to version 10 when opened. No user intervention is required. Projects saved in version 10 cannot be opened in an older version of MassMotion.

#### Simulation Results

Database files are not compatible between versions. Older simulation runs cannot be used for playback or analysis and will have to be rerun using version 10.0. Projects saved within old mmdb files can be opened from newer versions though the database will be unavailable for analysis or playback.

#### Converting Workbench Object(s)

Some objects created in the MassMotion 7 Softimage workbench cannot be authored in MassMotion 10. These objects will import into a version 10 project and can be used to execute a simulation, but cannot be modified. To change the behaviour of these objects they must be converted into comparable MassMotion 10 objects. Some information may be lost in the conversion. To convert an object right click on it in the main window list view and select "Convert Workbench Object(s)" or use the button in the workbench object's properties window.

Workbench Event	MassMotion Event	Notes
Workbench Schedule	Journey	<p>Start time, demand curve, population count, profile, actions, origins, destinations are all converted with no loss.</p> <p>Agent colours are specified using a rule generated from the origin/destination avatar assignment from the workbench schedule.</p> <p>Avatar assignment is lost during conversion and all agents use the same avatar from the specified profile.</p>
Workbench Evacuation	Broadcast	<p>An inline action is created to mimic exactly the behaviour of the workbench evacuation event. Agents are given a wait task, a seek best portal task, and then an exit simulation task. No information is lost in the conversion.</p>

#### MassMotion 7

- Goal line tolerances were tightened. Some links, stairs, ramps, escalators, or portals that validated in version 6 may have to be adjusted in version 7.
- Unique names enforced. In version 6 it was possible to have objects of different types with the same name. As of version 7 all objects must have a unique name regardless of type.

#### MassMotion 8

- Validation errors must now be resolved before a simulation can execute.
- Old style process chains with connected server port groups will automatically be converted to the new dispatch process chain style.

#### MassMotion 9

- The 'effective width' property has been removed from [Connection Objects](#). This property only affected agent perception of queue costs and was seldom used.

## 4.1.2 Importing Data

MassMotion can import 3D objects, 2D drawings, images, or exported MassMotion objects through the import button in the project ribbon of the [Main Window](#) or through the 'File' menu. See [Import File Formats](#) for a list of supported file formats.

During import a prompt will appear with [options](#) for controlling how the file will be handled. Not every file format will ask for every option. All imported objects are grouped under a single [Transform](#) collection which can be used to scale, rotate, or translate the imported geometry as a group.

3D objects are imported as either [Generic Geometry](#) or [IFC Geometry](#) while 2D objects or layers are imported as [Drawing Layers](#). Images are imported as a rectangle with the image projected onto it.

### 4.1.2.1 Import File Formats

MassMotion can import the following file formats:

3D Objects	
<b>3ds</b>	3D Studio Scene
<b>dae</b>	COLLADA
<a href="#">dgn</a>	Bentley Systems' MicroStation
<a href="#">dwg</a>	AutoCAD Drawing Database
<a href="#">dxf</a>	Drawing Exchange Format
<a href="#">fbx</a>	Autodesk FBX
<b>ifc</b>	Industry Foundation Classes
<b>obj</b>	Wavefront 3D Object
<b>skp</b>	Sketchup: all versions up to and including 2016

2D Drawings	
<a href="#">dgn</a>	Bentley Systems' MicroStation
<a href="#">dwg</a>	AutoCAD Drawing Database
<b>skp</b>	Sketchup

Images	
--------	--



<b>bmp</b>	Microsoft Windows Bitmap
<b>jpg/jpeg</b>	Joint Photographic Experts Group (JFIF)
<b>png</b>	Portable Network Graphics

<b>MassMotion Objects</b>	
<a href="#">mmxml</a>	MassMotion exported objects

## 4.1.2.1.1 dgn

**DGN** files can be imported as both drawing layers and generic geometry. Support for various elements is listed in the table below:

Type	Supported DGN Elements	Unsupported DGN Elements
<b>2D elements (imported as drawing layer)</b>	arcs, planar splines, conics, ellipses, lines, planar line strings, quad grid, quad list, triangle grid, triangle list, 2D shapes, closed 2D splines	non-planar line strings and splines
<b>3D elements (imported as generic geometry)</b>	3D shapes, closed 3D splines	
<b>Other</b>		dimensions, linear patterns, text

## 4.1.2.1.2 dwg

**DWG** files can be imported as both drawing layers and generic geometry. Support for various elements is listed in the table below:

Type	Supported DWG Elements	Unsupported DWG Elements
<b>2D elements (imported as drawing layer)</b>	lines, circles, arcs, ellipses, planar polylines, planar splines, regions	non-planar polylines and splines
<b>3D elements (imported as generic geometry)</b>	meshes, polyface meshes, polygon meshes, solids, faces, regions	NURBs based geometry (ie. surfaces and 3D solids)
<b>Other</b>		dimensions, hatches, leaders, text, wipeouts

Note: Regions will be imported as both a drawing layer for its outline and a generic geometry for the filled area.

Imported 2D elements will be grouped into drawing layer objects by the layers from the source DWG

file.

4.1.2.1.3 dxf

**DXF** 3D object support in MassMotion is currently limited to explicit mesh geometry definitions. DXF files that represent geometry through ACIS definitions are not yet supported.

4.1.2.1.4 fbx

**FBX** files created with AutoCAD use a very old version of the file format and may not be read correctly by MassMotion. To upgrade AutoCAD exported .fbx files it is recommended to use the [FBX 2013.3 Converter](#) which is available free of charge from Autodesk.

4.1.2.1.5 mmxml

MMXML files are imported as a set of MassMotion objects. When an imported file contains objects with the same name as existing objects, the newly imported objects will be renamed by adding a numerical suffix. In some situations, importing objects will lead to a conflict with objects currently in the project. In this case the [Merging Projects](#) window will be used to resolve the conflicts

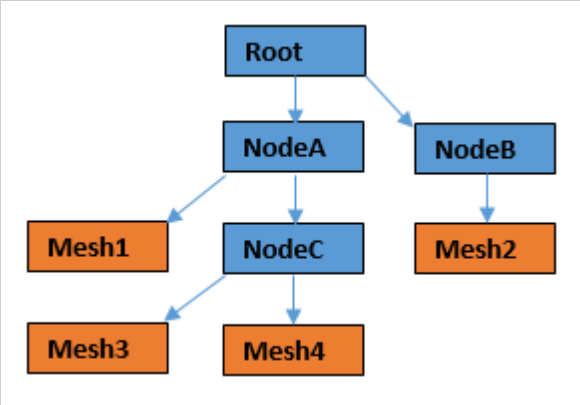
If the imported objects reference one another or reference objects already in the existing project, those references are maintained or re-established. If there are references to objects not included in the import or the existing project, those references are deleted.

See [MassMotion Objects](#) for information on exporting mmxml files.

**4.1.2.2 Import Options**

Importing a file will present the following options:

Option	File Formats	Effect
<b>Geometry - Import</b>	.dgn, .dwg, .dxf, .skp	Options: <ul style="list-style-type: none"> <li>• All Objects: Import both drawing layers and generic geometry</li> <li>• Meshes Only: Import only generic geometry</li> <li>• Drawings Only: Import only drawing layers</li> </ul>
<b>Geometry - Include Reference Attachments</b>	.dgn	DGN files may reference other DGN files. These files can be imported together with the specified file.
<b>Geometry - NURB Quality</b>	.3ds, .fbx	Choose the level of detail when importing NURBs.  MassMotion only supports mesh geometry and so other forms of geometry such as NURBs will be converted into a mesh upon import. This is the number of subdivisions between the control points of the surface.  Users will be prompted for this before importing regardless of whether the file contains a NURB or not. If there are no NURBs, this option has no effect
<b>Geometry - Group faces</b>	.skp	Sketchup files may only contain planar elements (line sketches and faces oriented in 3D). For complex files, this restriction can lead to a very large number of objects on import. To reduce this amount, faces can be grouped together with the following

		<p>options:</p> <ul style="list-style-type: none"> <li>• Group faces by layer and appearance: Faces with the same layer and sharing the same texturing and colour properties will be combined into a single object.</li> <li>• Group faces by layer: Faces with the same layer will be combined into a single object. Texturing and colour are discarded.</li> <li>• Don't group faces: No grouping is performed. Texturing and colour are preserved. This may create a very large number of objects.</li> </ul>
<p><b>Object Names - Prefix</b></p>	<p><b>.3ds, .dae, .dgn, .dwg, .dxf, .fbx, .obj, .skp</b></p>	<p>Add a prefix to the names of every imported object. This is useful when searching for objects</p>
<p><b>Object Names - Naming Scheme</b></p>	<p><b>All except images</b></p>	<p><b>3ds, dae, fbx, and obj</b> files are internally organized such that geometry is associated with a node. Each node then has a 'parent' up to the 'root' node, forming a tree like structure:</p>  <pre> graph TD     Root[Root] --&gt; NodeA[NodeA]     Root --&gt; NodeB[NodeB]     NodeA --&gt; Mesh1[Mesh1]     NodeA --&gt; NodeC[NodeC]     NodeA --&gt; Mesh2[Mesh2]     NodeC --&gt; Mesh3[Mesh3]     NodeC --&gt; Mesh4[Mesh4]     NodeB --&gt; Mesh2     </pre> <p>These file formats give the following naming scheme options:</p> <ul style="list-style-type: none"> <li>• Full name: Use the geometry name and the names of every node up to the root node. The above example will create four objects named: Root_NodeA_Mesh1, Root_NodeB_Mesh2, Root_NodeA_NodeC_Mesh3, Root_NodeA_NodeC_Mesh4</li> <li>• Root name: Use only the root node's name. The above example will create four objects named: Root_0, Root_1, Root_2, Root_3</li> <li>• Leaf name: Use the geometry name and an optional number of nodes. The above example with a leaf depth of 2 will create four objects named: NodeA_Mesh1, NodeB_Mesh2, NodeC_Mesh3, NodeC_Mesh4</li> <li>• Root and leaf name: Use the geometry name, the root node's name and an optional number of nodes. The above example with a leaf depth of 2 will create four objects named: Root_NodeA_Mesh1, Root_NodeB_Mesh2, Root_NodeC_Mesh3, Root_NodeC_Mesh4</li> </ul> <p><b>dgn, dwg, dxf, and skp</b> files are internally organized using layers and blocks. A block is typically a 'model space' block which represents the entire model and possibly other user defined blocks. Blocks contain meshes (and other geometry elements) as well as references to other blocks. The meshes are also organized into one or many layers.</p>

		<p>These file formats give a number of options to name imported objects based on imported object's layer, name, original block and type (drawing or mesh).</p> <p>For example, Mesh3 will be imported 3 times, once via BlockA referenced by the Model block and twice via BlockB. The &lt;Layer&gt;_&lt;Block&gt;_&lt;Name&gt; option will give the names: Layer0_BlockA_Mesh3_0, Layer0_BlockA_Mesh3_1, Layer0_BlockA_Mesh3_2.</p>
<p><b>Image - Size</b></p>	<p><b>Images only</b></p>	<p>Imported images can have their imported dimensions set either as fixed size per pixel or by directly setting the width and height.</p>

### 4.1.3 Exporting Data

#### 4.1.3.1 Agent Position Export

The agent position and other physical attribute data generated by a MassMotion simulation can be exported to a CSV file using the "Agent Position Table Export" dialog. This dialog can be accessed through a button in the project tab of the main window's ribbon.

The exported table will include the frame number, agent ID and XYZ position of each agent. Additional data can be included by checking the "Optional columns" options.

Columns are in the following order:

Frame Number, Agent ID, X Position, Y Position, Z Position, Clock Time (optional), Speed (optional)

#### Options

Option Name	Description
File	The CSV file to which the table will be written.
Simulation run	The simulation run for which data will be written to the table
Time range	The time period over which agent positions will be included.
Sampling period	Modifies the number of samples included in the table. By default, every frame will be sampled, producing data for every agent for every frame. However, the number of samples can be reduced to once per second or more.

<b>Agent Filter</b>	Used to select which agents will be included in the table.
<b>Optional columns</b>	<p>Additional information can be included in the agent position table, toggled by the following options:</p> <p><b>Clock time:</b> The simulation clock time corresponding to the frame number.</p> <p><b>Speed:</b> The agent's speed in m/s. See the note on agent speed below.</p> <p><b>Heading:</b> The agent's heading in degrees. An agent facing the same direction as the Z axis will have a heading of 0, and the angle increases as the agent turns counterclockwise.</p> <p><b>Move State:</b> The agent's move-state as a number. The meaning of move state values is listed below.</p> <p><b>Animation time:</b> The time an agent has spent in a given move state, useful for controlling the speed of animations. When walking on flat ground, the value is the number of strides taken by the agent.</p>

### Agent Speed

The exported agent speed is the speed used to calculate the agent position for that frame. In areas of high density, positions are also adjusted so as to minimize agent overlap. These adjustments are made after initial positions are calculated. As a result, the exported speed might not match the speed calculated from the displacement of an agent from one frame to the next.

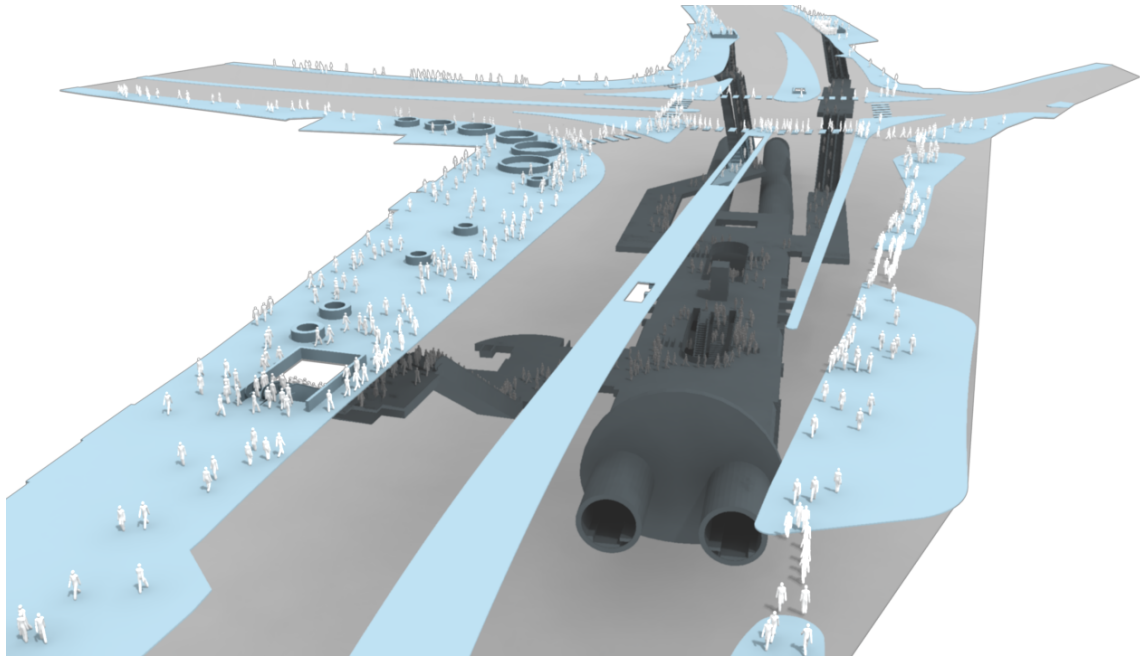
### Move State

Integer Value	Description
0	Walking on a flat surface
1	Shuffling (occurs in crowded places)
2	Standing still
3	Walking Up
4	Walking Down

#### 4.1.3.2 Alembic Export

Alembic is an open computer graphics interchange framework. Alembic distils complex, animated scenes into a non-procedural, application-independent set of baked geometric results. For more information please visit [www.alembic.io](http://www.alembic.io).

MassMotion provides the ability to export Alembic files (\*.abc) containing animated agent meshes that can be imported into visualization software such 3DS Max or Maya for inclusion in rendered scenes. The Alembic export option can be found on the Simulation & Analysis tab of the main window. Note that the exported data does not currently include the scene geometry (floors, walls etc.); if desired, scene geometry can be exported separately from the [main menu](#).



Alembic Export Options	
<b>Simulation run</b>	Which simulation run to use as the source for agent data.
<b>Time range</b>	The time range over which to export agent data.
<b><a href="#">Agent filter</a></b>	Which agents to include in the output.

#### 4.1.3.3 Movie and Image Export

Movies and images of the scene can be produced using the movie/image export window. The export window can be accessed from the project tab of the main window's ribbon.

When the export window is first opened, the initial camera position, background colour, object visibility, and other settings are copied from the scene in the main window. Initial export options are copied from the defaults defined in the [Application Preferences](#). The export window operates on a copy of the current project and so any modifications made through the main window will not affect the project displayed in the export window. Both the appearance of the scene and the project in general can be reloaded from the main window by pressing the "Reload" button.

[Simulation databases](#) are locked to ensure continuous access during movie recording and therefore simulations cannot be re-run while the export window is open.

<b>Ribbon</b>	
<b>Reload</b>	Reload changes to the project and scene's appearance from the main window.
<b>Render movie</b>	Start rendering a movie or frames. This process may take some time.
<b>Pause</b>	Pause current render.

<b>Abort</b>	Stop current render and possibly discard any progress.
<b>Render Image</b>	Capture and save a single image based on the current view. The following file formats are supported: *.png, *.jpg, *.tiff

<b>File Options</b>	
<b>Save Movie</b>	A movie file will be produced with the desired quality setting. Increased quality will increase file size. The following file formats are supported: *.mov, *.mp4, *.m4v, *.wmv
<b>Save Individual Frames</b>	Individual frames will be produced and placed in the specified directory. All images will have the *.png format.
<b>Save Movie and Frames</b>	Both a movie file and individual frames will be produced.

<b>Timing</b>	
<b>Time Range</b>	The period of time to export.
<b>Frame Rate</b>	The frame rate of the movie to be exported. A higher frame rate will increase the movie file size and the number of frames produced.
<b>Play Speed</b>	How fast the movie will appear to play. "x2" will result in a movie with agents moving twice as fast as normal.
<b>Time Lapse</b>	Create a movie or series of frames where frames are recorded regularly at the specified interval. The resulting movie will appear to play in fast forward.

<b>Appearance</b>	
<b>Resolution</b>	The resolution of movies, frames and images produced. This can dramatically affect the size of file outputs. The chosen resolution is displayed in the view through the use of dark letterbox frames at the edges of the view.
<b>Background</b>	The background colour.

<b>Overlay</b>	
<b>Text</b>	The colour of overlay text (if any).
<b>Population count</b>	Whether to include text which shows the current population.
<b>Simulation time</b>	Whether to include overlay text that shows the current simulation time.

<b>Map legend</b>	Whether to include a legend for the displayed map (if any).
<b>Reference axes</b>	Whether to include the reference axes in the top right corner.

#### 4.1.3.4 Simulation Slice Export

A database slice is a database file exported from a source simulation run. The exported slice often covers a smaller time range within the larger simulation. Slices can be used for playback but cannot be used for analysis queries such as maps, tables, or graphs.

Slices can be exported as a MassMotion database (mmdb) file or MassMotion Viewer (mmv) file. Both formats can be viewed in the [MassMotion Viewer](#) but only the MassMotion database file can be opened again in MassMotion or Flow.

If a [simulation run is connected](#) to an existing database file, the project embedded in the simulation run might be different from the active project displayed in the authoring environment. When exporting a slice of an existing database file, it is the project embedded in the existing database that will also be embedded in the exported slice. It is possible to add objects from the authoring environment to the exported project by specifying them explicitly during export using the "Bookmarks" or "Add or Replace Objects" fields.

Slice export can be found in the Project tab of the ribbon on the main window. It can also be accessed through the File menu.

#### Options

Option Name	Description
<b>Simulation run</b>	The simulation run for which data will be exported.
<b>Time range</b>	The time interval for which data will be exported.
<b>Bookmarks</b>	The <a href="#">bookmarks</a> to be included in the mmdb slice. These objects are from the currently active project and will be added to the embedded project. If no bookmarks are specified, the embedded project will not contain any bookmarks.
<b>Default Bookmark</b>	The default bookmark to use when opening the mmdb or mmv project.
<b>Add or Replace Objects</b>	Objects from the project currently active in the authoring environment that should be added to the project embedded in the mmdb slice. If the embedded project already contains a version of a specified object the embedded version is replaced.



#### 4.1.3.5 Geometry Export

MassMotion can export geometry objects through the export button in the project ribbon of the [Main Window](#). The following file formats are supported:

3D Objects	
<b>dae</b>	COLLADA
<b>fbx</b>	Autodesk FBX (2012, 2013, 2014, 2016)
<b>obj</b>	Wavefront 3D Object  Files will be allowed UTF-8 encoded file names, however, UTF-8 encoded object names are not supported. All non-ASCII characters in object names will be converted to under-scores: '_'.  

MassMotion Objects	
<b>mmxml</b>	MassMotion objects

#### 4.1.3.6 MassMotion Objects

Objects such as floors, journeys, actions, bookmarks, can be moved between projects or versions of the same project through the mmxml file format. Exported objects will include object properties and geometry.

References between objects included in the export are maintained and will be re-established on [mmxml import](#). This can be useful when exporting a set of related objects such as a journey and all associated origin and destination portals.

#### 4.1.4 Project Settings

The project settings include critical information such as the timing of simulation runs and location of results. The project settings are available from the project tab of the main window's ribbon.

General tab	
Runtime: Start Time	Sets the starting time of the simulation. Values are in hh:mm:ss. See <a href="#">Working with Time</a> .
Runtime: Duration	Sets the duration of the simulation. Values are in hh:mm:ss. See <a href="#">Working with Time</a> .
Simulation: Random Seed	Sets the seed of the project. The project seed governs several project variables, such as the distribution of agent speeds and where and when agents are created. See <a href="#">Randomness</a> .
Simulation: Population Multiplier	Modifies the global population by the given factor. This scales the number of agents produced by all <a href="#">events</a> . This can be a fractional value of less than one or greater than one.
Results Working Folder	Sets the folder where the default simulation run and analysis outputs will be sent.

Authoring tab	
Bookmark	Sets an optional <a href="#">bookmark</a> to apply when opening a project. This can be used to set the default viewpoint, playback time and object visibility when the project is opened.

Debug files such as surface map images or route cost spreadsheets are placed in a 'debug' folder alongside the simulation run's results database file.

Debug tab	
Simulation Diagnostic Files: Surface Maps	Enables creation of debug surface maps for every object in the scene.
Simulation Diagnostic Files: Route Costs	Enables creation of route cost spreadsheets for every <a href="#">portal</a> in the scene.

### 4.1.5 Files

#### MassMotion Files

File Type	Description
.mm	The standard MassMotion project file. Contains all <a href="#">objects</a> and the <a href="#">project settings</a> .
.mmdb	The results of a MassMotion simulation.  A <a href="#">simulation run</a> is used to access a mmdb file from within the project. A simulation also produces several other files as listed in <a href="#">Generated Simulation Files</a> .  Each mmdb file contains: <ol style="list-style-type: none"> <li>1. Playback data used to show agent movement in the 3D view.</li> <li>2. A series of SQLite tables used by analysis queries such as <a href="#">Maps</a>, <a href="#">Graphs</a>, <a href="#">Tables</a>.</li> <li>3. An embedded copy of the project used to generate the simulation</li> </ol> See <a href="#">Simulation Data</a> for more information.
.mmv	An exported database slice for exclusive use by the <a href="#">MassMotion Viewer</a> .
.mmxml	Exported MassMotion objects.
.mmxsi	A project exported from the Softimage workbench using MassMotion 7.

#### Import Files

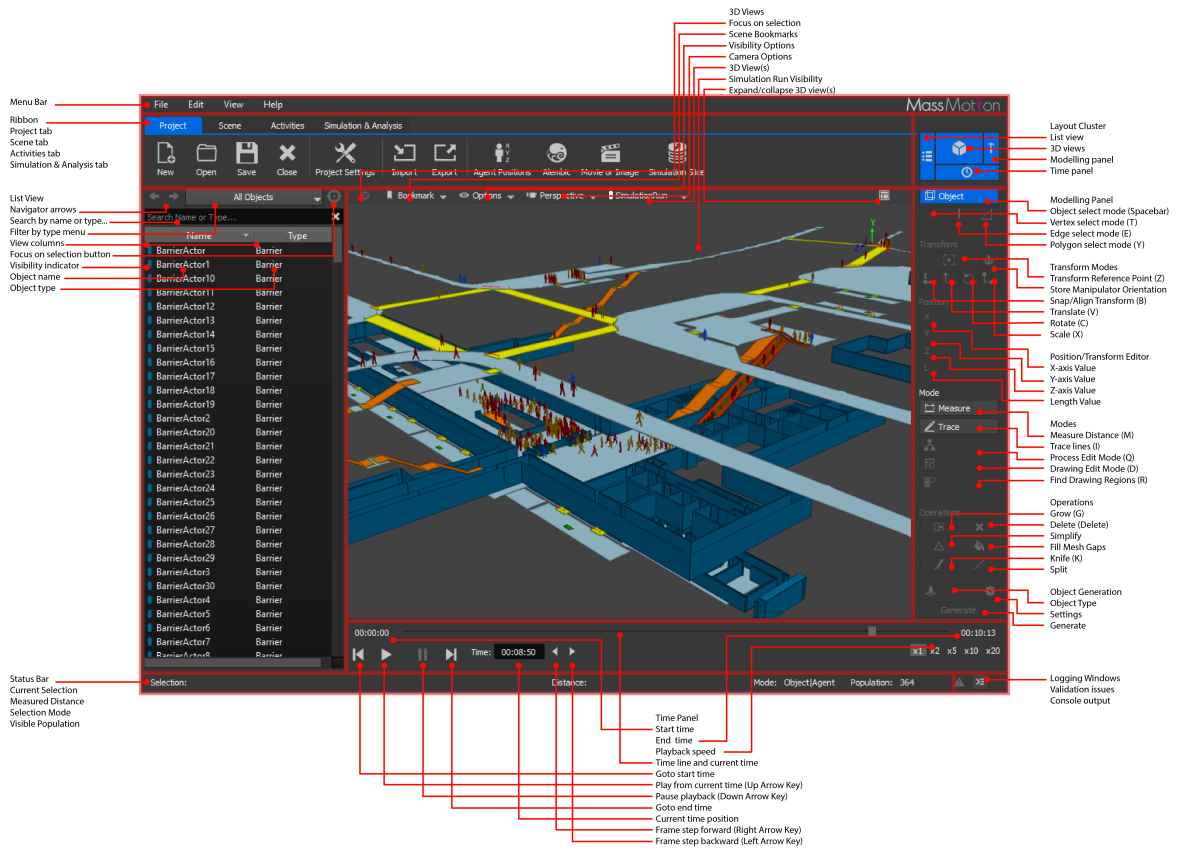
MassMotion can import geometry from a variety of sources as [Reference Geometry](#), which can be used to create other [scene objects](#). See [Importing Geometry](#) for more information.

#### Export Files

MassMotion's [analysis tools](#) can export data for further processing. All [tables](#) and [graphs](#) can export their datasets to a .csv file, and graphs can additionally be exported as an image. [Agent Position Export](#) can be used to output the position of agents from a simulation run to a .csv file. [Movie and Image Export](#) can be used to produce images and movies in various formats. [Alembic Export](#) can be used to export animated agent models for use in animation/visualization packages.

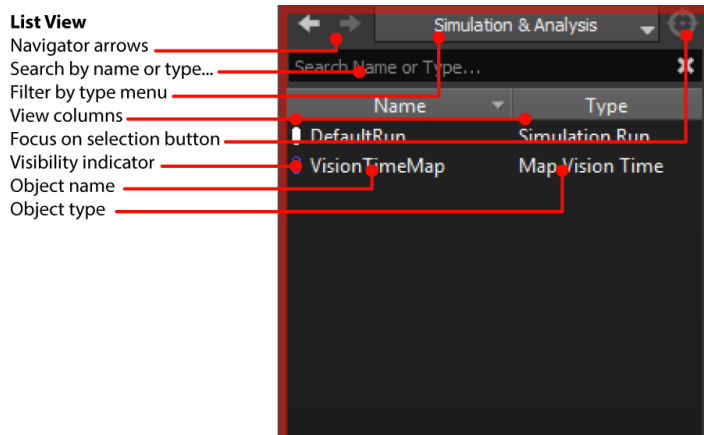
## 4.2 User Interface

### 4.2.1 Main Window



### 4.2.1.1 List View

The list view presents the name and type of all objects in the project. The list can be filtered to display only objects meeting certain criteria.



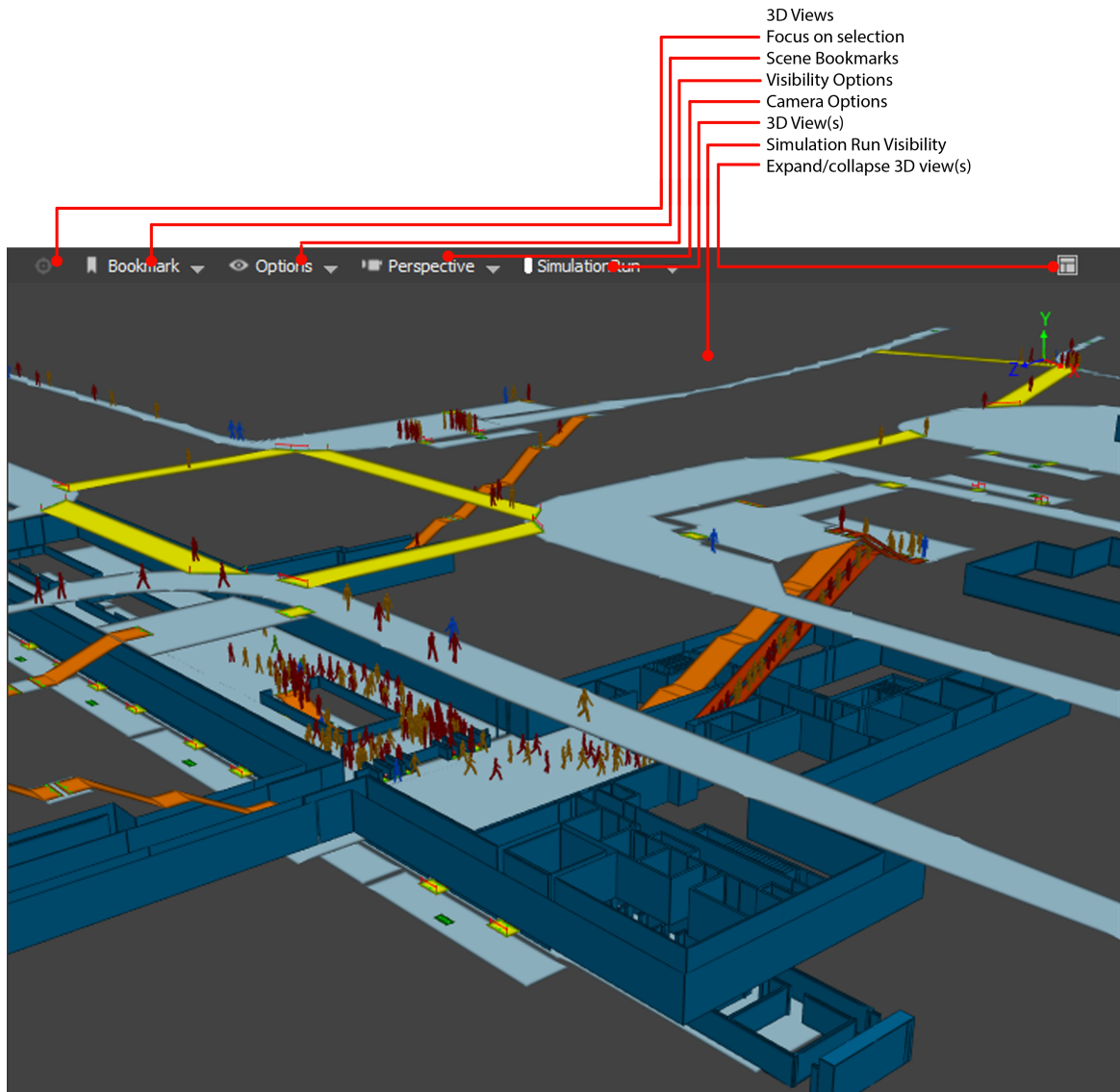
The list view has built in filters accessible through the "Filter by type menu". Alternatively, right clicking on objects in the list view or scene view and using a "Find" command will create a custom filter presenting only the found objects.

UI Element	Description
<b>Navigator Arrows</b>	Go backwards and forwards through the history of displayed objects.
<b>Search by name or type...</b>	Search objects by name or type. Objects will only be shown if either the object's name or its type contains the given text.
<b>Filter by type menu</b>	Filters the list by objects of a certain type. Categories of objects can be used, such as all <a href="#">collections</a> or all <a href="#">analysis objects</a> .
<b>View columns</b>	The column headers can be used to sort objects by name or type.
<b>Focus on selection button</b>	Filters the list to display all currently selected objects.
<b>Visibility Indicator</b>	Each object has an indicator which shows its colour and visibility. Hidden objects will have only an outline as an indicator. Disabled objects will have a small grey dot.
<b>Object Name</b>	Each object's name. The name can be edited by selecting one or more objects and pressing the 'F2' key.
<b>Object Type</b>	Each object's type.

### 4.2.1.2 3D Scene View

The 3D scene view shows the spatial arrangement of all [scene objects](#). Decorators display the status of various objects.

Right clicking on objects in the scene view produces a menu which allows users to manipulate the scene. Additional controls including how to move the camera view can be found in [Keyboard and Mouse Controls](#).



UI Elements	
<b>Focus on selection</b>	Centre and zoom in on the currently selected object(s).
<b>Bookmark</b>	Save the current view to a <a href="#">bookmark</a> or apply an existing bookmark. The following options can be used to save different types of bookmarks:  <b>Everything:</b> Saves the viewpoint, the simulation playback time and shown and hidden objects.

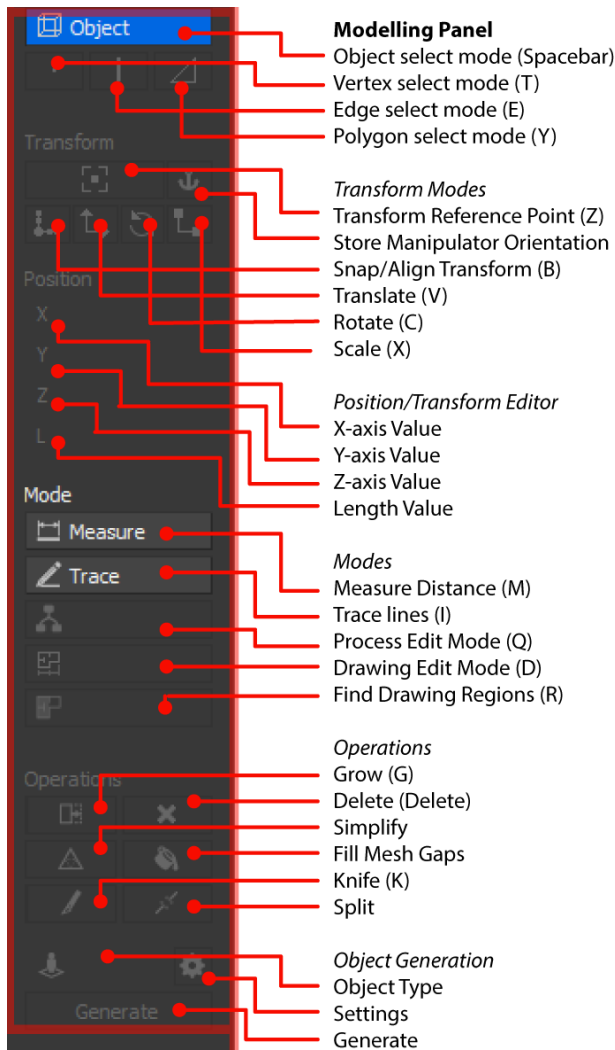
	<p><b>Viewpoint:</b> Saves only the viewpoint.  <b>Shown Objects:</b> Saves only the shown and hidden objects.</p> <p>These can be further modified in the bookmark's properties.</p> <p>Once bookmarks are saved, they will be displayed in the bookmark menu. Their names will be listed alongside icons indicating their type.</p>
<b>View options</b>	<p>Change the appearance of objects in the scene view. These settings are very useful when setting up the scene to <a href="#">export a movie or image</a>.</p> <p><b>Render Type:</b> Switch between <b>Shaded</b> and <b>Wireframe</b> representations.</p> <p><b>Agents:</b> Change the appearance of agents. See the table below for a description of the options.</p> <p><b>Decoration:</b> Toggle the visibility of all <a href="#">decorations</a> of a given type in the scene view. Decorations can be turned on or off for specific objects by right clicking on an object and using the 'Display' menu.</p> <p><b>Geometry:</b> Change the appearance of geometry. See the table below for a description of the options.</p> <p><b>Overlay:</b> Toggle the appearance of overlays which display information such as the current simulation time and visible population. It can also be used to hide the reference axes which appear at the top right, and the legend which can be displayed when <a href="#">maps</a> are shown in the scene.</p> <p><b>Preview Resolution:</b> Constrain the view to display a specified resolution.</p>
<b>Viewpoint</b>	<p>Change the camera between perspective and various orthographic views. Custom views can be saved and applied in this menu as well.</p> <p><b>Perspective:</b> The default perspective view.  <b>Side/Top:</b> Orthogonal views. Camera can be panned and rotated horizontally.  <b>Front/Back/Left/Right:</b> Orthogonal views. Camera can be panned but not rotated.</p> <p>The following views can be activated when an agent is selected:</p> <p><b>First Person:</b> See from the perspective of an agent.  <b>Third Person:</b> See from over the shoulder of an agent.  <b>Track:</b> Follow agent position but don't rotate view.</p> <p>Agent cameras can also be activated by right clicking an agent.</p>
<b>Simulation run</b>	<p>Change the data source driving playback within the view (see <a href="#">Simulation Run</a>).</p>
<b>3D scene view</b>	<p>A graphical representation of all visible scene objects. See <a href="#">Keyboard and Mouse Controls</a> for information on moving the camera.</p>
<b>Expand/collapse view</b>	<p>Expand and collapse scene views. By default, only one scene view is displayed, but three are available. If only one view is visible, clicking this button will shrink that view so all three are shown. If all three views are visible, clicking any one of them will expand it to be the only one shown.</p>
<b>Agent</b>	

Visibility Options	
<b>Animated</b>	The default animated avatar with articulated arms and legs.
<b>Debug</b>	A low resolution simple peg. The size of the peg will change with the radius of the agent.
<b>Show Custom</b>	For a project containing advanced features, agents can be assigned <a href="#">custom geometry</a> .  This option enables the display of agents using that geometry. If the project does not contain custom agent geometry, the animated or debug agents are used.
<b>Hide Agents On Hidden Floors</b>	Agents on hidden <a href="#">floors</a> and other geometry will not be shown.

Geometry Visibility Options	
<b>Show Geometry Edges</b>	Toggle the drawing of edges on unselected objects. Only affects shaded rendering.
<b>Show Selection Highlights</b>	Toggle the highlighting of selected objects. The highlighting shows the outlines of selected objects even when occluded by others.
<b>Show Drawing Layer Titles</b>	Toggle the display of drawing layer titles.
<b>Darken Back Faces</b>	Darken the back side of mesh faces to help identify which is the front and which is the back.

### 4.2.1.3 Tool Panel

The tool panel provides tools for manipulating scene objects. Many of these tools can be accessed using [keyboard and mouse controls](#).



Selection Mode	
<b>Object select</b>	Cancel any interaction modes and return to object selection mode. See <a href="#">Selecting Object Components</a> .
<b>Vertex select</b>	Allow the selection of vertices for any selected mesh or line object. See <a href="#">Selecting Object Components</a> .
<b>Edge select</b>	Allow the selection of edges for any selected mesh or line object. See <a href="#">Selecting Object Components</a> .
<b>Polygon/ Face select</b>	Allow the selection of faces for any selected mesh object. See <a href="#">Selecting Object Components</a> .



<b>Position/ Transform</b>	
<b>Transform Manipulator</b>	Move the manipulator independent of the selection, changing the location or orientation of snap, translate, rotate, or scale. See <a href="#">Changing the Reference Frame</a> .
<b>Store Manipulator Orientation</b>	Store a custom manipulator orientation as the new default. See <a href="#">Storing the Manipulator Orientation</a> .
<b>Snap</b>	Enable snapping selected object/components. See <a href="#">Transforming Objects</a> .
<b>Translate</b>	Enable translation of selected objects/components. See <a href="#">Transforming Objects</a> .
<b>Rotate</b>	Enable rotation of selected objects/components. See <a href="#">Transforming Objects</a> .
<b>Scale</b>	Enable scaling of selected objects/components. See <a href="#">Transforming Objects</a> .
<b>X/Y/Z/L Values</b>	Allows for direct editing of the position or length of selected objects or components. When using the manipulator these can be used to precisely manipulate selected objects/components. See <a href="#">Transforming Objects</a> .

<b>Modes</b>	
<b>Measure</b>	Measure the distance between two points. Additional measurement points can be added by holding the shift key.  The total distance is displayed in the Distance section of the <a href="#">Main Status Bar</a> .
<b>Trace</b>	Trace over existing objects or drawings and use the lines/regions to generate new objects. See <a href="#">Tracing New Objects</a> .
<b>Edit Process</b>	Edit the connections between servers. See <a href="#">Process Chains</a> .
<b>Edit Drawings</b>	Edit the lines and points inside a drawing. See <a href="#">Working with Drawings</a> .
<b>Find Regions</b>	Find enclosed regions inside selected drawings and use them to generate new objects. See <a href="#">Generating From Drawings</a> .

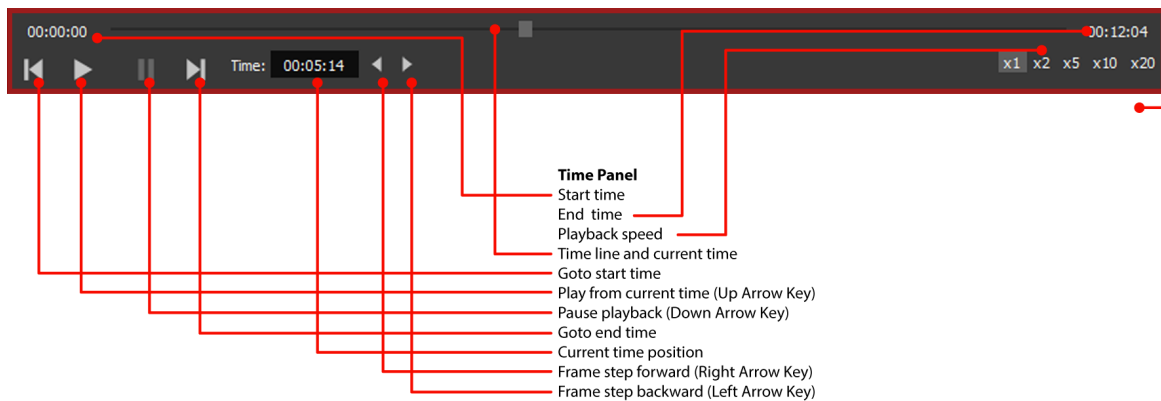
<b>Operations</b>	
<b>Grow</b>	Extrude a selected edge or face. See <a href="#">Extrude (Grow)</a> .
<b>Delete</b>	Delete selected objects or components.
<b>Simplify</b>	Merge edges and faces to reduce geometric complexity without changing the object shape. See <a href="#">Editing Geometry</a> .
<b>Fill Mesh Gaps</b>	Add faces to close any holes or gaps in a mesh. See <a href="#">Fill</a> .

<b>Slice</b>	Cut selected objects along a cutting plane. See <a href="#">Slice</a> .
<b>Split</b>	Subdivide a selected edge or face. See <a href="#">Editing Geometry</a> .

<b>Object Generation</b>	
<b>Target Type</b>	Change the type of object generated. See <a href="#">Creation Widget</a> .
<b>Options</b>	A toggle button which displays the 'Generate Options' window for editing operations used when generating objects. See <a href="#">Creation Widget</a> .
<b>Generate</b>	Generate a new object of the target type using the current selection (hotkey 'N'). See <a href="#">Creation Widget</a> .

#### 4.2.1.4 Time Panel

The time panel displays the current time and can be used to [play back](#) simulations in the [scene view](#).

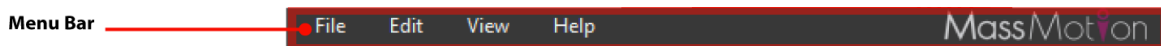


UI Element	Description
<b>Start Time</b>	The simulation's start time. See <a href="#">Project Settings</a> for more information. If multiple <a href="#">simulation runs</a> have different start times, the earliest one is used.
<b>End Time</b>	The simulation's end time. See <a href="#">Project Settings</a> for more information. If multiple <a href="#">simulation runs</a> have different end times, the latest one is used.
<b>Playback Speed</b>	1x is "real time" and higher values effectively "fast forward" through playback.
<b>Timeline and current time</b>	A slider which indicates playback progress. The slider can be repositioned to change the current time.
<b>Go to start time</b>	Sets the current time to the start time.
<b>Play from</b>	Plays back simulations from the current time. MassMotion simulations are

<b>current time (Up arrow)</b>	calculated and recorded in 0.2 second increments. When playback is active the animation system will use linear frame interpolation to provide the smoothest possible representation of the currently selected playback speed in the 3D views.
<b>Pause playback (Down arrow)</b>	Pause playback.
<b>Go to end time</b>	Sets the current time to the end time.
<b>Current time position</b>	Displays the current time. Can be edited to skip to a specific moment.
<b>Frame step forward</b>	Advance the current time by 1 simulation frame (no frame interpolation is applied)
<b>Frame step backward</b>	Move the current time back 1 simulation frame (no frame interpolation is applied)

#### 4.2.1.5 Main Menu Bar

The MassMotion Application Menu is available from the menu bar at the top of the MassMotion window.



File	
<b>New</b>	Create a new empty project.
<b>Open</b>	Open either a previously saved MassMotion project file (.mm), a project exported from the Softimage workbench (.mmxsi), or a project contained in a MassMotion results database (.mmdb).
<b>Open Recent</b>	Open a project from a list of recently used project files.
<b>Save</b>	Save the project as a .mm project file. Note that this will save as a new file if the project was initially opened from a Softimage export (.mmxsi).
<b>Save As</b>	Save the project as an alternative .mm project file with a new name.
<b>Close</b>	Stop all running simulations and close the current project.
<b>Merge</b>	Merges the open MassMotion project with data from another MassMotion project file (.mm), a project exported from the Softimage workbench (.mmxsi), or with project data contained in a MassMotion results database (.mmdb). See <a href="#">Merging Projects</a> for more information.
<b>Import</b>	Import either: <ul style="list-style-type: none"> <li>• Objects exported from other projects (.mmxml) (see <a href="#">Importing and Exporting Objects</a>).</li> <li>• Analysis objects from MassMotion 6.0 or 6.1 projects (Analysis.xml) (see <a href="#">Importing and Exporting Objects</a>).</li> </ul>

	<ul style="list-style-type: none"> <li>• <a href="#">Avatar</a> geometry</li> <li>• <a href="#">Reference geometry</a> exported from other CAD/geometric authoring applications.</li> </ul>
<b>Export</b>	<p>Export:</p> <ul style="list-style-type: none"> <li>• Objects for import into other MassMotion projects (.mmxml) (see <a href="#">MassMotion Objects</a>).</li> <li>• Geometry to neutral file formats such as Collada (.dae) for import into other CAD/geometric authoring applications</li> <li>• Agent positions data in a simple text file (see <a href="#">Agent Position Export</a>).</li> <li>• Alembic animation data for rendering agent movement in applications like Softimage or 3ds Max (see <a href="#">Alembic Export</a>).</li> <li>• A movie or image generated from the current project (see <a href="#">Image and Movie Export</a>).</li> </ul>
<b>View Auto Saves</b>	Open an explorer window showing the location in which projects are automatically saved.
<b>Exit</b>	Stop all running simulations and quit the application.

<b>Edit</b>	
<b>Undo</b>	Undoes the previous command. Can be stepped back multiple times.
<b>Redo</b>	Steps forward if the Undo command is used. Can be used multiple times.
<b>Select All</b>	Select all objects in the project.
<b>Select Inverse</b>	Select all objects other than those in the current selection.
<b>Deselect All</b>	Deselect all objects currently selected.
<b>Project Settings</b>	Edit the <a href="#">Project Settings</a> .
<b>Application Preferences</b>	Edit the <a href="#">application preferences</a> .

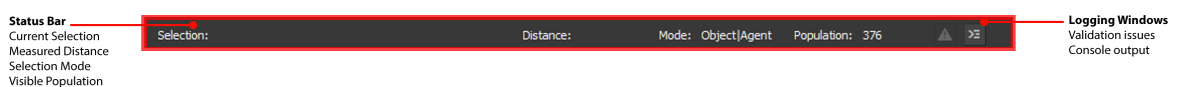
<b>View</b>	
<b>List View</b>	Toggle display of left-hand side <a href="#">list view</a> showing list of all existing object (s) in the project.
<b>Scene View</b>	Toggle display of all <a href="#">3D scene views</a> .
<b>Time View</b>	Toggle display of bottom pane <a href="#">time panel</a> with playback controls (See <a href="#">Playback</a> ).
<b>Context View</b>	Toggle display of right-hand side pane with contextual tools for operating on currently selected objects; see <a href="#">tool panel</a> for details.
<b>Other Windows</b>	Toggle the display of additional windows:

	<p><b>Console:</b> Show a console window with MassMotion diagnostic output.</p> <p><b>Issues:</b> Show the <a href="#">Issue Window</a> with the most recent list of warnings, errors, or audit information; see <a href="#">Validation</a> for details</p> <p><b>Observe Agents:</b> Display information about a particular agent from a particular run (see <a href="#">Agent Observer</a>).</p>
<b>Close All Windows</b>	Closes any pop-up properties or analysis windows.
<b>Show All</b>	Unhide all objects.
<b>Show Selected</b>	Unhide all selected objects.
<b>Hide Selected</b>	Hide all selected objects.
<b>Hide All But Selected</b>	Hide all objects that are not part of the current selection (leaving only selected objects visible).
<b>Hide Maps On Selected</b>	If any of the selected surface objects have been covered by an analysis map texture, the map texture is removed (does not apply to textured reference geometry)
<b>Hide Maps On All</b>	Remove all analysis map textures from all surface objects in the scene (does not apply to textured reference geometry)

Help	
<b>Show Help...</b>	Display the MassMotion user guide.
<b>Keyboard Controls...</b>	Display the user guide page describing keyboard shortcuts and controls.
<b>Licensing...</b>	Open the license manager.
<b>Check for Updates</b>	Check for more recent releases of MassMotion.
<b>Graphics Driver Diagnostics</b>	Displays the current OpenGL version as well as any issues detected; MassMotion requires OpenGL version 3.0 or higher for full functionality. If you are having problems with graphics not being displayed properly, or crashes when attempting to open or create a project, please include this diagnostic information in any support request.
<b>About MassMotion...</b>	Display information on the current version of MassMotion.

#### 4.2.1.6 Main Status Bar

The main window status bar is at the bottom of the [main window](#).



#### Main Status Bar

<b>Selection</b>	Displays the name of the object currently selected. If more than one object is selected the number of objects selected is displayed. If object components are selected, the number of components is displayed.
<b>Distance</b>	Displays the length of the edge currently selected, along with the component horizontal and vertical lengths. If more than one edge is selected, the total length of all edges is displayed. If two vertices are selected, the distance between the vertices is displayed. If the <a href="#">measure tool</a> is active, the total distance measured is displayed. If two straight lines from <a href="#">Drawing Layers</a> are selected, it shows the angle between them if they were connected end to end.  In all cases the number is in metres or degrees.
<b>Mode</b>	The current selection mode: object, face, edge, vertex.
<b>Population</b>	The total of all agents from all connected simulation runs that are alive at the current time. This counts includes hidden agents, ignoring the simulation run filter and visibility state of the simulation run.
<b>Issue Window</b>	Show the <a href="#">issue window</a> including the most recent list of issues or audits.
<b>Console Window</b>	Show the console window.

## 4.2.2 Selection

The selection is a list of objects or object components chosen by the user. Objects can be selected in the [scene view](#), [list view](#), or through individual choosers. The selection mode, which toggles between selecting objects, faces, edges, or vertices, can be changed through [keyboard shortcuts](#) or buttons in the [tool panel](#).

### Select Here, Selected Everywhere

The selection status of an object or object component is a property of the object and so visible in all parts of the user interface. If an object is selected in the [list view](#), it will show as selected in the [scene](#). This behaviour can be leveraged when choosing objects. If a dialog is presented which asks for an object but the user is unsure of the name of the object, the object can be selected in the scene and that choice will automatically be reflected in the dialog. Similarly, if a user is unsure about which object a dialog is describing, that object can be selected in the dialog, then the scene 'focus' button can be used to view the selected object graphically.

### Selection Mode

The default selection mode is 'Object|Agent'. In this mode, entire objects or playback agents can be selected. The component selection modes are only available when one or more objects are selected. The component selection modes allow for the selection of face, edge, or vertex components of the selected object(s). The selection mode can be changed from the [tool panel](#) in the Main Window or using [keyboard shortcuts](#).

### Selecting in the Scene View

In the scene view, an object can be selected by clicking on it with the left mouse button. Holding down CTRL will add each subsequent object to the selection. A selection box can be used by left-clicking and dragging the mouse. If the mouse is dragged from left to right, only those objects or components which are entirely inside the box will be selected. If the mouse is dragged from right to left, all objects or components which intersect with the box will be selected. See [3D Scene View](#) for

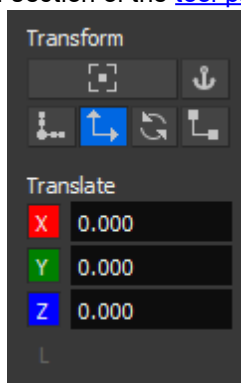
more information.

### 4.2.3 Manipulator

The manipulator is a tool used to [transform objects](#) or object components (vertex, edge, face, drawing). The manipulator has 4 modes: **Object Translate**, **Object Rotate**, **Object Scale** and **Object Snap**. There are additionally two sub-modes: one for [changing the reference frame](#) of the manipulator and another for [growing objects](#)

Each mode will generally allow the user to independently transform in each of the X, Y and Z axes. These axes are coloured red green and blue respectively.

The manipulator will appear in the scene at the centroid of the current selection when accessed via keyboard shortcuts or via the Transform section of the [tool panel](#):



The top left button enables the manipulator snap sub-mode.  
 The top right button toggles saving the axis alignment (see below: Saving the Axis Alignment).  
 The next four buttons can be used to open the manipulator in a given mode (snap, translate, rotate, scale) or to close it if that mode is already active.  
 The X, Y, and Z buttons can be pressed to disable an axis and prevent it from changing.  
 The three entries beside them can be used to enter precise values for a transformation (ie. translating along Z by exactly 4m).

See [Editing Geometry](#) for examples of use.

#### Keyboard Shortcut Summary

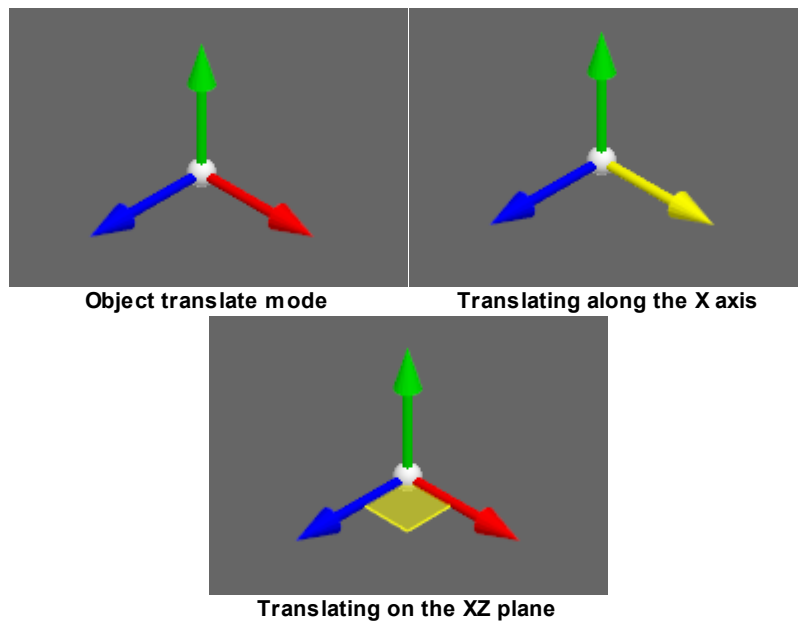
Sho rtcu t	
<b>V</b>	Translate objects/components
<b>C</b>	Rotate objects/components
<b>X</b>	Scale objects/components
<b>B</b>	Snap to position/orientation
<b>G</b>	Grow components
<b>Z</b>	Translate or rotate the manipulator through snap

<b>1</b>	Enable/disable X axis
<b>2</b>	Enable/disable Y axis
<b>3</b>	Enable/disable Z axis
<b>Shift</b>	Rotate: Constrain the rotation to 15 degree increments Snap: Reverse the axis alignment direction

**4.2.3.1 Transforming Objects**

**Object Translate (Shortcut: V)**

This mode is used to move the current selection along an axis or a plane. The axes orientation can be adjusted via [manipulator snap](#) for off-axis translations.



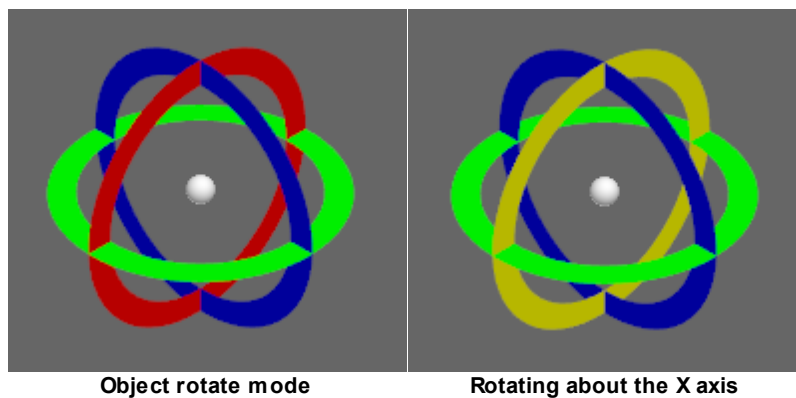
Element	Control
<b>Axis sticks</b>	Click and drag along the axis to translate in that direction.
<b>Between two axis sticks</b>	Click and drag on the axes' common plane to translate.

**Object Rotate (Shortcut: C)**

This mode is used to rotate the current selection about the manipulator centre. The centre of rotation can be adjusted via [manipulator snap](#). The axes can also be adjusted for off axis rotations.

Holding shift will constrain the rotation to 15 degree increments.

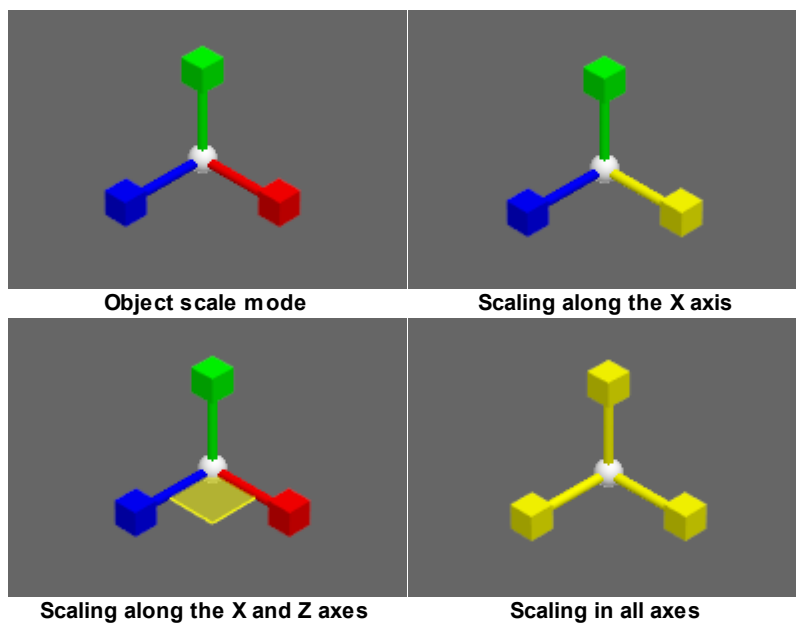




Control	Function
Ring	Click and drag around the centre to rotate.

### Object Scale (Shortcut: X)

This mode is used to scale the current selection about the manipulator centre. This can be used to both stretch and shrink an object/component. The axes can be adjusted via [manipulator snap](#) for off-axis scaling.

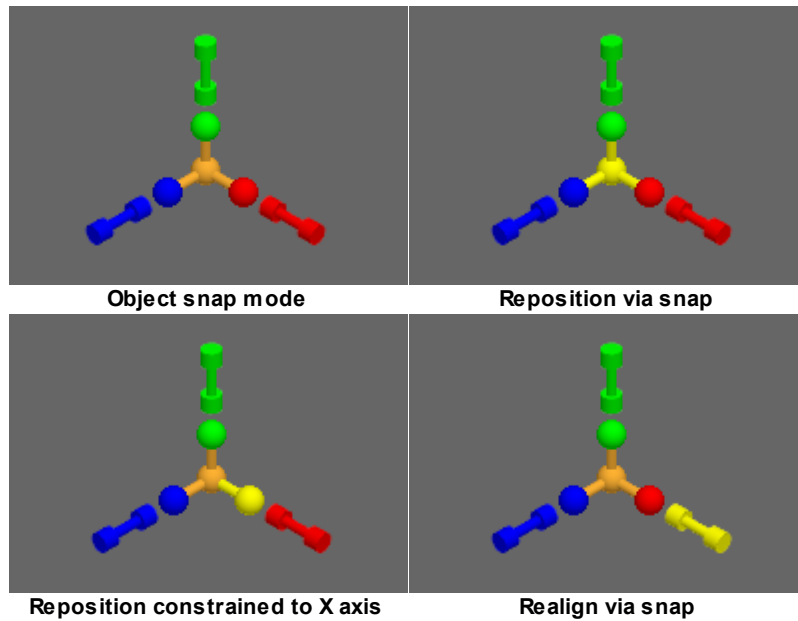


Control	Function
Axis sticks	Click and drag along the axis to scale in that direction.
Between two axis sticks	Click and drag towards/away from the manipulator centre to scale equally in both axes.
Elsewhere	Click and drag towards/away from the manipulator centre to scale equally in all axes.

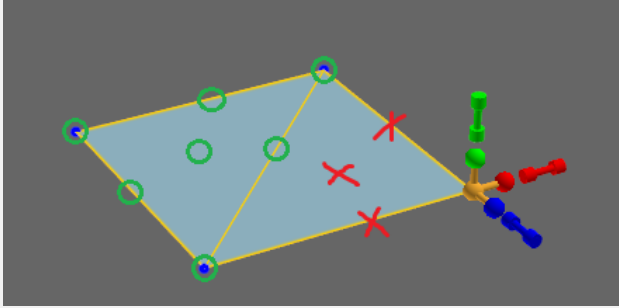
**Object Snap (Shortcut: B)**

This mode is used to reposition or realign the current selection with respect to other objects and components. Unlike the other modes, this mode has two clickable elements per axis as well as a clickable orange centre ball. It is often practical to reposition the manipulator with [manipulator snap](#) before any operations.

When snapping components, a silhouette will appear showing the original geometry. Snapping to the silhouette will return the manipulator to its starting position, providing a convenient way to cancel the operation.



Control	Function
<p><b>Orange centre ball</b></p>	<p>Click and drag to another element in the scene. This will move the manipulator centre and all selected objects/components.</p> <p>Valid targets are:</p> <ul style="list-style-type: none"> <li>• Another object's vertex, edge, face component</li> <li>• Another object's drawing or shape component (see: Drawings)</li> <li>• The midpoint of an edge or drawing line shape</li> </ul> <p>When snapping components to each other, components of selected objects are also valid targets. However, only components which will not change as a result of the operation can be targeted.</p> <p>For example, when snapping the corner of a floor, the adjacent edges and face are not valid snap targets:</p>

Control	Function
	 <p>The snapped move can be constrained by disabling various axes. For example, disabling Y will disable motion in the Y direction. This can be used to reposition objects at different elevations.</p>
<b>Axis coloured balls</b>	<p>Click and drag to another element in the scene. This will move the manipulator centre and all selected objects/components. The motion will be constrained to the selected axis. For example, dragging the green Y ball will only change the elevation of the selection. A coloured guide line will be drawn to indicate the axis constraint.</p> <p>Valid targets are:</p> <ul style="list-style-type: none"> <li>• Another object's vertex, edge, face component</li> <li>• Another object's drawing or shape component (see: Drawings)</li> <li>• The midpoint of an edge or drawing line shape</li> </ul>
<b>Barbells</b>	<p>Click and drag to an edge or drawing line. This will rotate the manipulator so that the selected axis is aligned to the edge/line with all the selected objects following its motion. The axis can be aligned to an edge in either direction. Hold shift to reverse the alignment to the other direction.</p> <p>Valid targets are:</p> <ul style="list-style-type: none"> <li>• Another object's edge</li> <li>• Another object's drawing or shape component (see: Drawings)</li> </ul> <p>This will likely lead to the manipulator not being aligned to the global axes. This can be reset by closing and reopening the manipulator. Alternatively, the new alignment can be preserved (see <a href="#">Saving Reference Frame</a>).</p>

#### 4.2.3.2 Changing the Reference Frame

Manipulator snap can be used to change the centre of rotation or scaling, to work in an off-axis model, or to align two different objects and bring them flush with each other.

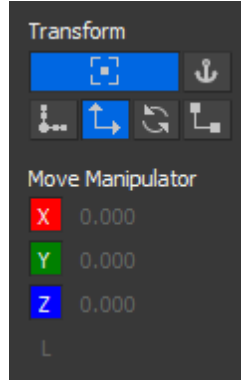
Changes made to the local reference frame persist for as long as the manipulator remains open. Once the manipulator is closed its position and orientation are reset to their default. See [Storing the Manipulator Orientation](#) for information on how to change the default orientation of the manipulator.

See [Tips and Tricks](#) for additional use cases for manipulator snap.

**Manipulator Snap (Shortcut: Z)**

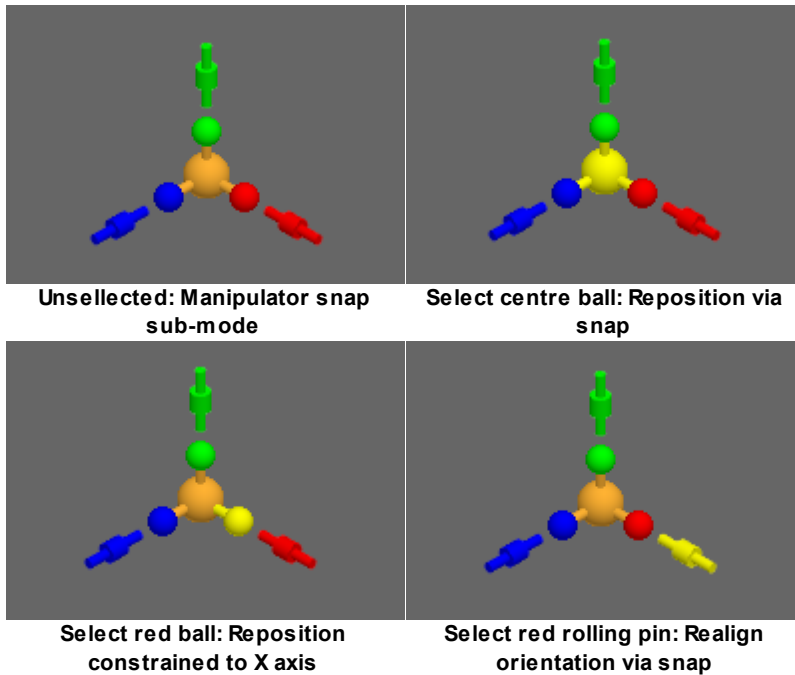
The manipulator snap sub-mode allows users to reposition and realign the manipulator without modifying any of the selected geometry. Its controls are almost identical to the [object snap](#).

Manipulator snap can be toggled on/off by pressing the 'Z' key in any manipulator mode. Pressing 'Z' without a manipulator will open an object snap manipulator in the manipulator snap sub-mode.



Transform section of the Tool Panel indicating translate with manipulator snap sub-mode

See: [Geometry Manipulation](#) or [Tips and Tricks](#) for examples of use.

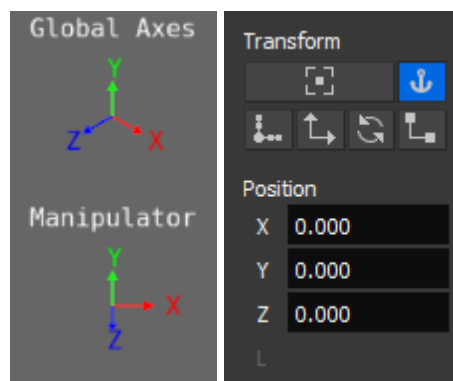


Control	Function
<b>Orange centre ball</b>	Click and drag to a another element in the scene. This will move the manipulator centre there.  Unlike Object Snap, components of selected objects can be targeted. Valid

	<p>targets are:</p> <ul style="list-style-type: none"> <li>• Any object's vertex, edge, face component</li> <li>• Any object's drawing or shape component (see: Drawings)</li> <li>• The midpoint of an edge or drawing line shape</li> </ul> <p>The snapped move can be constrained by disabling various axes in the transform section of the model panel. For example, disabling Y will disable motion in the Y direction.</p>
<b>Axis coloured balls</b>	<p>Click and drag to another element in the scene. This will move the manipulator centre while constraining motion to the selected axis. For example, dragging the green Y ball will only change the elevation of the selection.</p> <p>Valid targets are:</p> <ul style="list-style-type: none"> <li>• Any object's vertex, edge, face component</li> <li>• Any object's drawing or shape component (see: Drawings)</li> <li>• The midpoint of an edge or drawing line shape</li> </ul>
<b>Rolling pins</b>	<p>Click and drag to an edge or drawing line. This will rotate the manipulator so that the selected axis is aligned to the chosen edge/line. The axis can be aligned to an edge in either direction. Hold shift to reverse the alignment to the other direction.</p> <p>Valid targets are:</p> <ul style="list-style-type: none"> <li>• Another object's edge</li> <li>• Another object's drawing or shape component (see: Drawings)</li> </ul> <p>The new manipulator orientation can be preserved as the new default (see <a href="#">Saving Reference Frame</a>) or reset by closing and reopening the manipulator.</p>

#### 4.2.3.3 Storing the Manipulator Orientation

Storing the manipulator orientation is useful when working with several off-axis objects which share the same alignment. Toggling the "Store Manipulator Orientation" button in the [Tool Panel](#) will store the current orientation. Whenever the manipulator is closed and reopened, the stored orientation is used instead of the global axes.



The scene will show the stored axes as well as the global axes.

The anchor button is toggled on, indicating a new orientation has been stored.

Resetting the reference frame to the default frame aligned with the coordinate axes can be done by toggling the "Store Manipulator Orientation" button off.

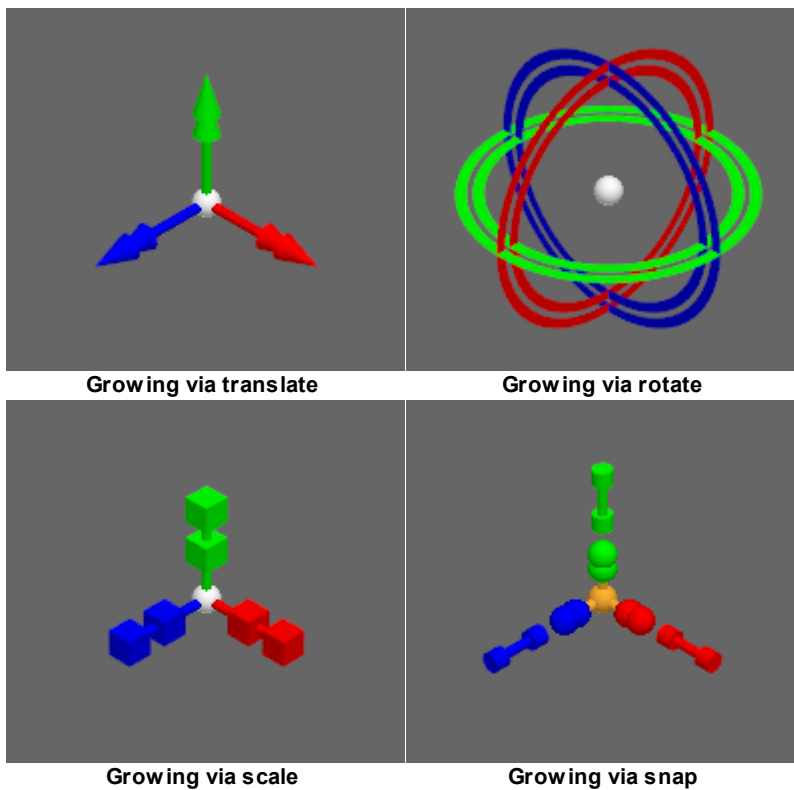
#### 4.2.3.4 Growing Objects

##### Component Grow (Shortcut: G)

A grow operation will create an extrusion from the selected components. The grow sub-mode can be easily identified by the doubling of particular manipulator elements.

Grow can be toggled on/off by pressing the 'G' key in any manipulator mode. Pressing 'G' without a manipulator will open an object translate manipulator in the grow sub-mode.

See: [Geometry Manipulation](#) or [Tips and Tricks](#) for examples of use.

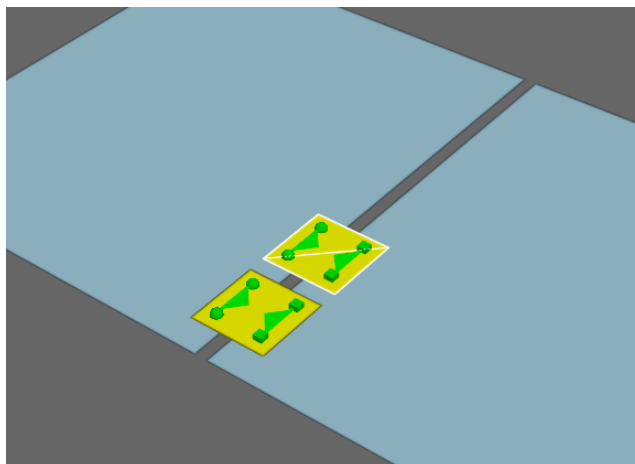


#### 4.2.3.5 Tips and Tricks

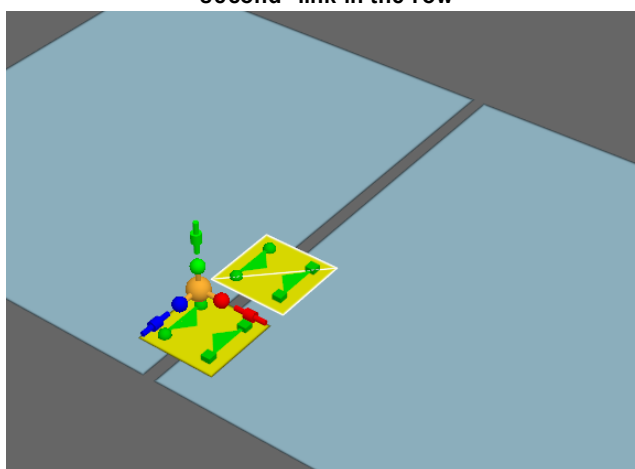
See [Geometry Manipulation](#) for simple uses of object and manipulator snap such as aligning off-axis geometry.

##### Evenly spaced links

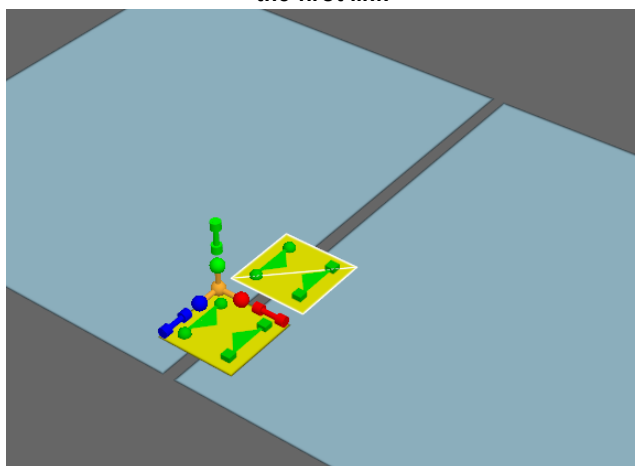
Duplicating an object (Ctrl-D) while the manipulator is open will preserve the current position of the manipulator and select the new objects. This can be combined with object snap to create evenly spaced links:



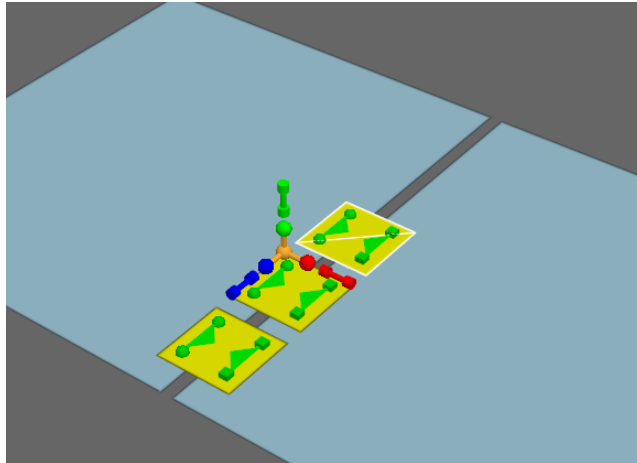
1) Two links with the desired spacing. Select the "second" link in the row



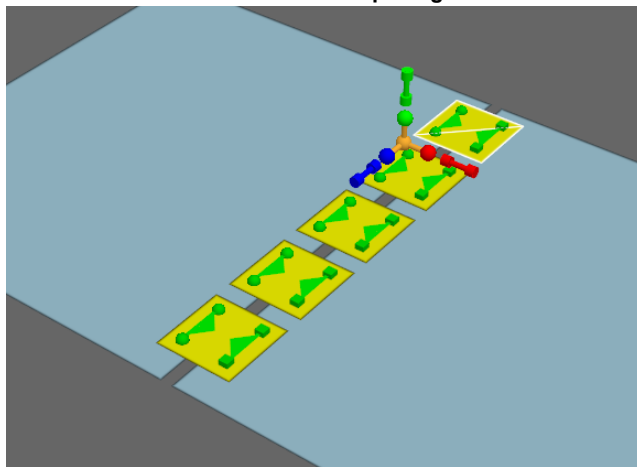
2) Using manipulator snap, position the manipulator on the first link



3) Duplicate the link and switch to object snap. The manipulator position is preserved.



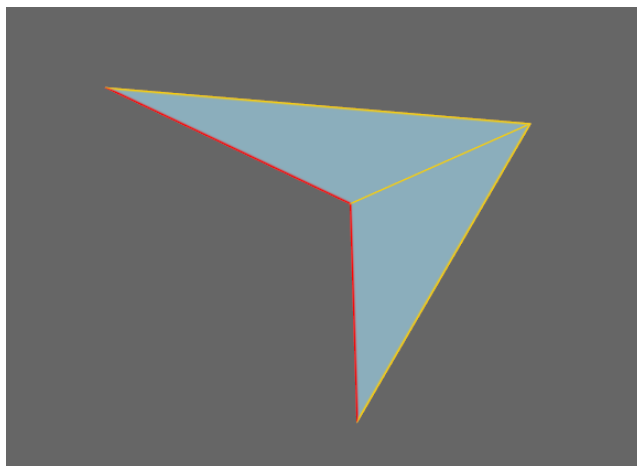
4) Snap to the same position on the second link. This ensures even spacing.



5) Repeat this process as many times as required.

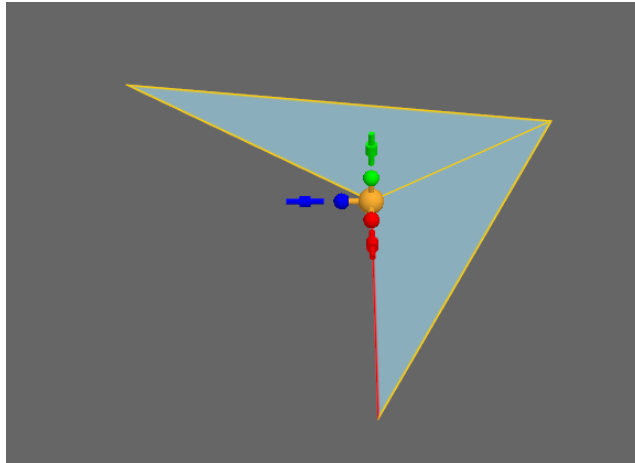
**Straightening an edge**

Straightening the edges of a floor (perhaps to simplify geometry) can be done with component level snaps:

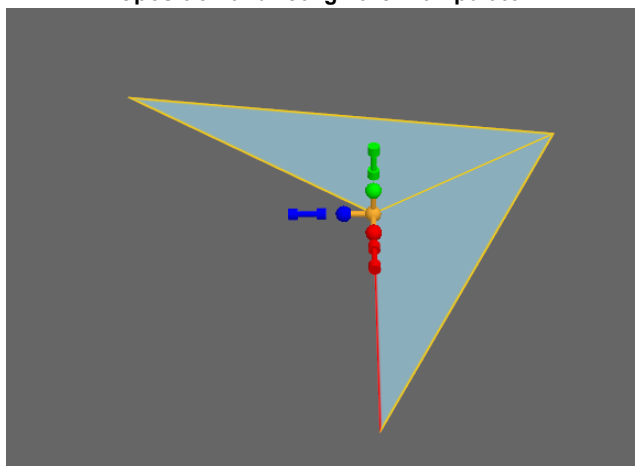


1) Aligning these two edges.

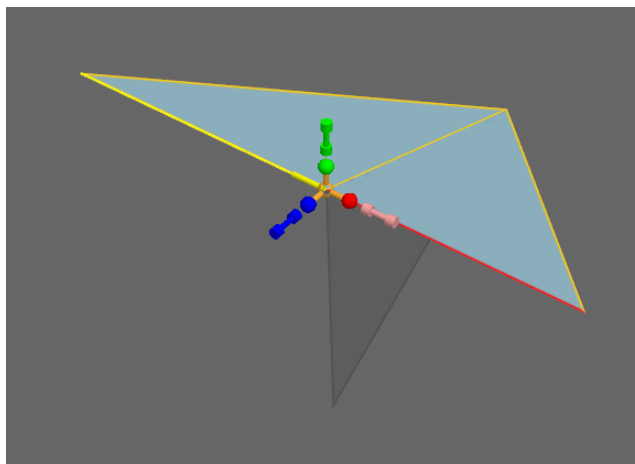




2) Select one edge and use manipulator snap to reposition and realign the manipulator.



3) Switch to object snap.

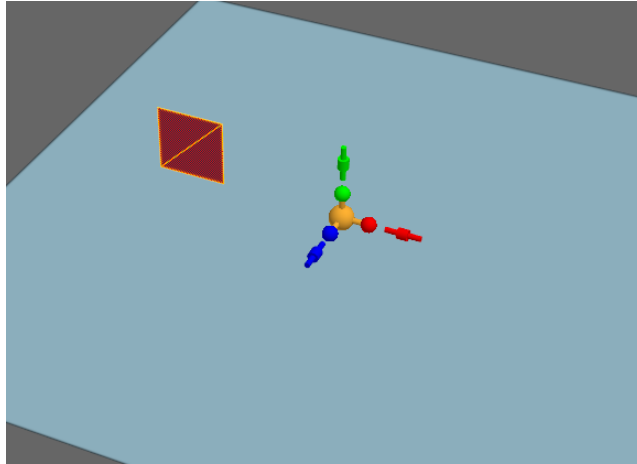


4) Align the edge to the other edge. You may need to hold shift to toggle the align direction.

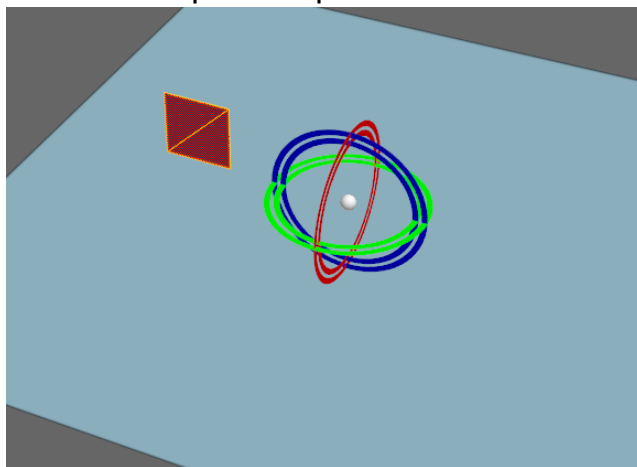
Note: The shadow is the silhouette of the pre-snap geometry.

#### **Scale/Rotate Grow**

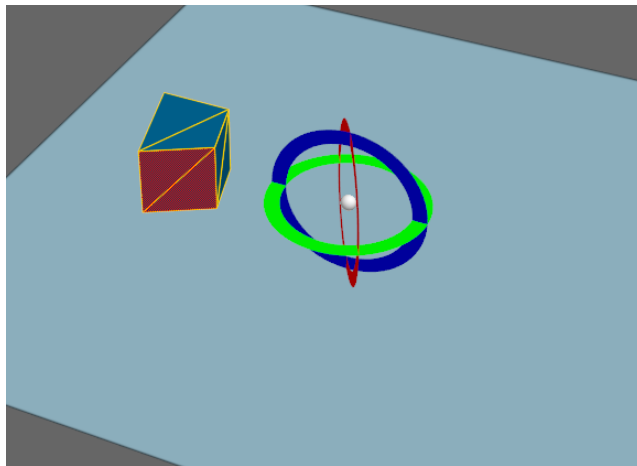
Scale and rotate grow are not typically useful functions, but can be useful with the manipulator snap. For example, it can be used to create a round barrier:



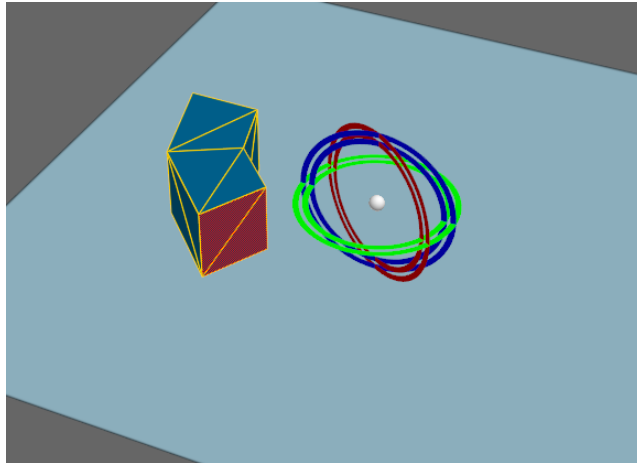
**1) Starting with the faces of a square barrier selected. Use manipulator snap to offset the centre.**



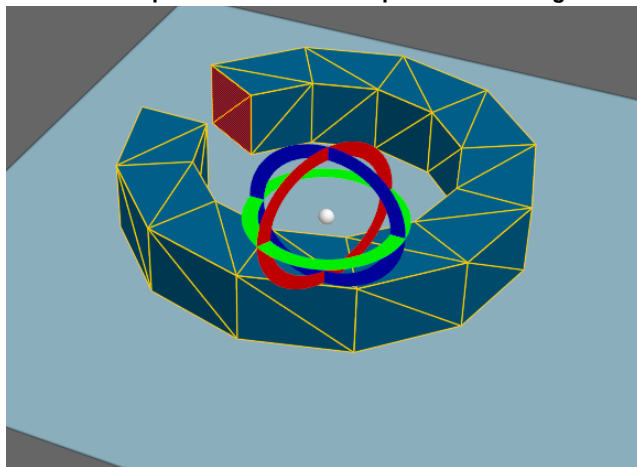
**2) Switch to object rotate with grow enabled.**



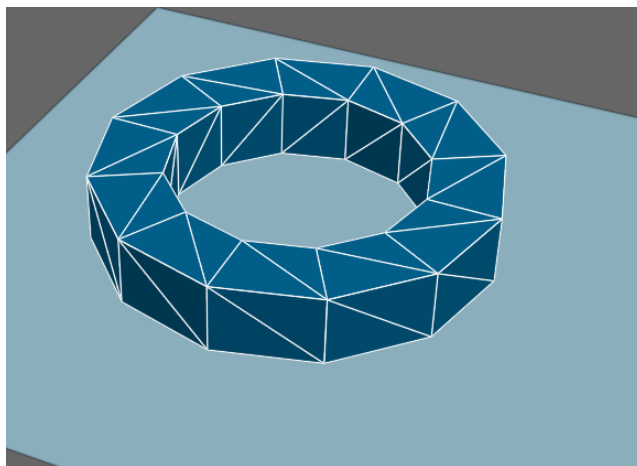
**3) Perform the rotate/grow. Hold shift to lock into 15 degree increments.**



4) Grow the next segment. If using irregular angles, rotate snap can be used to keep consistent angles.



5) Continue the process.

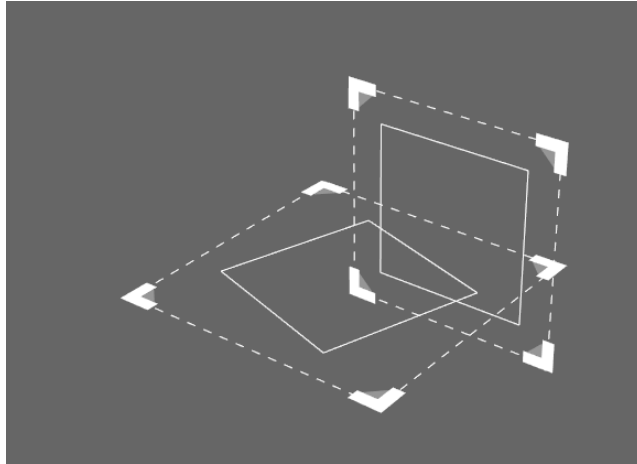


6) Final product.

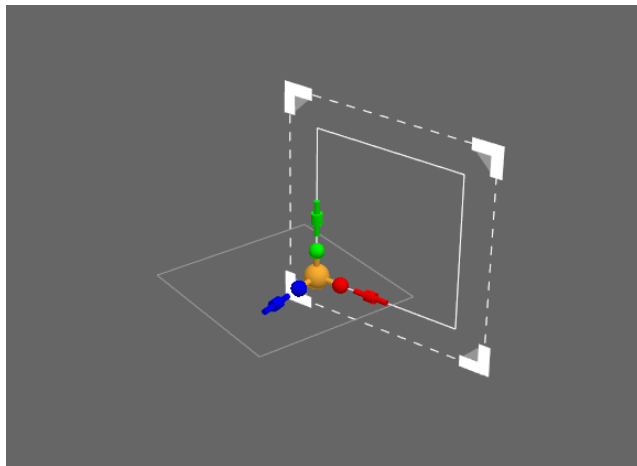
If using irregular angles, you can use a process similar to the evenly spaced links described above.

#### **Arranging section and plan drawings**

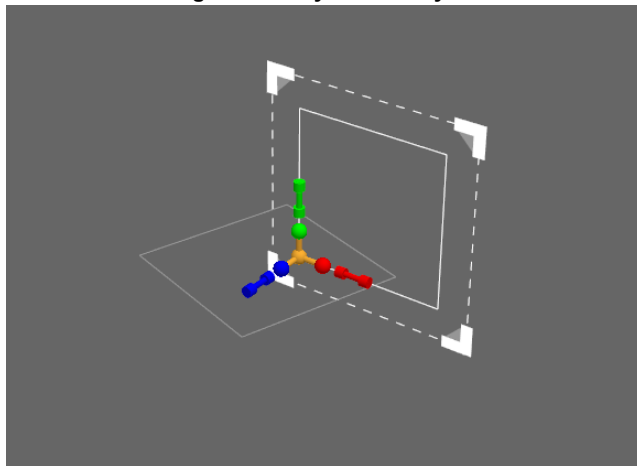
Like 3D scene geometry, imported drawings can be used as snap targets. This can help when aligning sections and plans.



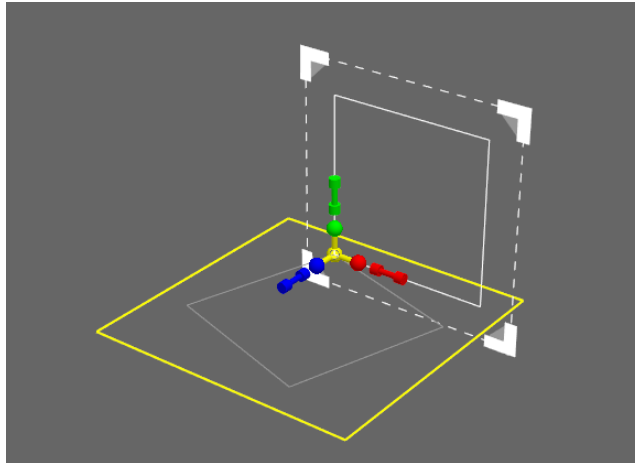
**1) A plan and a section drawing.**



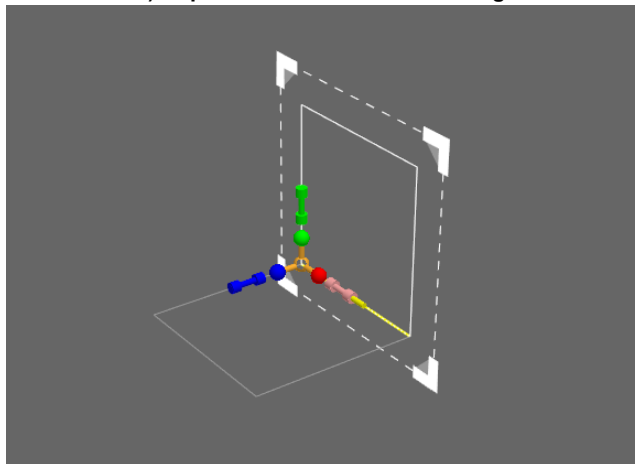
**2) Select the section drawing. Use manipulator snap to move the manipulator to a feature common to both drawings. Possibly a walkway corner.**



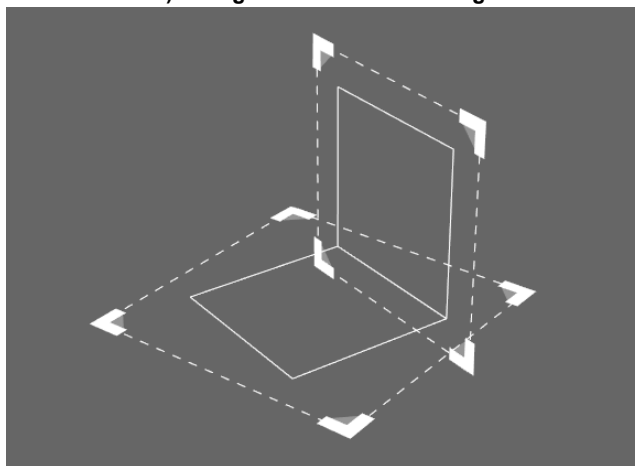
**3) Switch to object snap.**



4) Reposition the section drawing.



5) Realign the section drawing.



6) The plan and drawing are now correctly positioned.

#### 4.2.4 Generating Objects from Geometry

It is possible to generate new [scene objects](#) from [traced lines](#), [reference geometry](#), [drawing layers](#), or the faces or edges of other objects. There are two primary methods for generating objects: The right-click '[Generate](#)' [context menu](#), and the [creation widget](#) in the bottom of the [tool panel](#).

When working with IFC, it is possible to automatically generate new objects based on the IFC type. See [Generating From IFC](#) for the mapping between IFC type and scene object type.

#### 4.2.4.1 Creation Widget

The creation widget is in the bottom right of the [tool panel](#) in the main window. It configures the process of generating objects from source geometry or [traced lines](#).

The 'Generate' button (or hotkey 'N') will use the selected source geometry or traced lines to generate an object of the specified target type. The transformation applied to the source will depend on the source type and the target object type. It is possible to customize this transformation using the [Generate Options](#) window which can be accessed through the button beside the 'Generate' button.

Once options have been set for a given source/target pair, those options will be used any time a source of that type is used to generate an object of the target type. Options are saved for each possible source/target pair.

Property	Description
<b>Target Type</b>	The type of object to generate.
<b>Options</b>	A toggle button which displays the ' <a href="#">Generate Options</a> ' window for editing operations used when generating objects.
<b>Generate</b>	Generate an object of the target type from the selected geometry (hotkey 'N').

##### 4.2.4.1.1 Generate Options

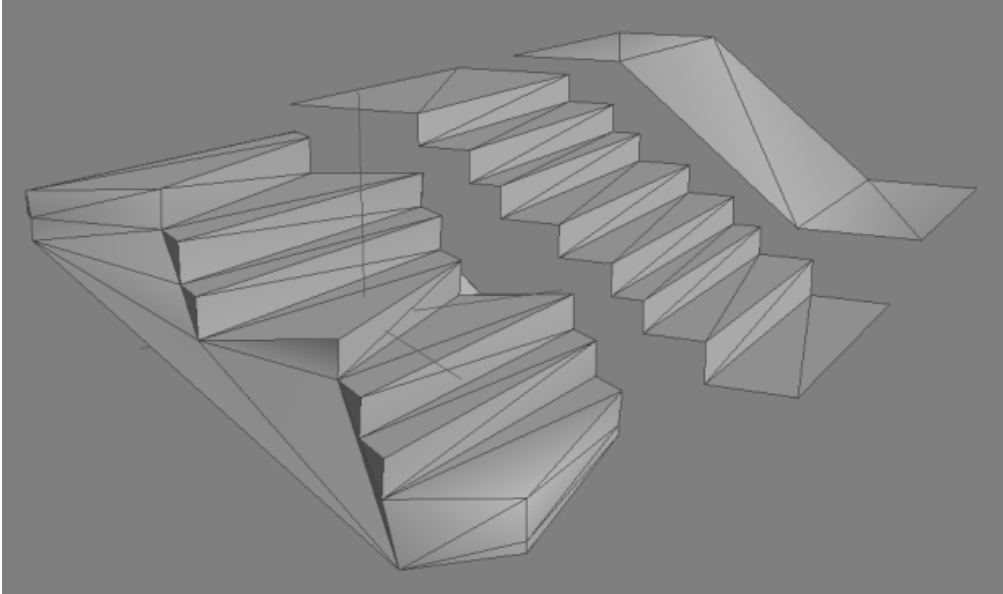
The Generation Options window can be used to customize the way in which objects are generated from the [Creation Widget](#) or 'N' hotkey. The window is available through the main window's 'View -> Windows' menu, the settings button in the [tool panel](#), or the 'Advanced' entry under 'Generate' in an object's right-click context menu.

Options are saved for each possible source/target pair.

Property	Description
<b>Target</b>	The type of object generated.
<b>Source Type</b>	The source component used to generate the target (Reference geometry, face, edge, region, line).
<b>Conversion Type</b>	The transformation applied to the source geometry (see below).
<b>Specify distance</b>	If checked, conversions that ask the user for a value will instead use the supplied value.
<b>Translate up</b>	If checked, the targeted geometry will be translated up the specified distance.
<b>Add landings with length</b>	If checked, landings of the specified length will be added to the target geometry. This can in some cases modify the target geometry and should be used with caution.
<b>Extend width</b>	Extend the target geometry by the specified distance on either side.

Property	Description
<b>Extend length</b>	Extend the target geometry by the specified distance on either side.
<b>Cleanup</b>	Determines how source and target geometry is handled after the target is generated.

Conversion Type	Source	Description
<b>Copy</b>	Object, Face, Region	Target geometry will be a copy of the source.
<b>Bottom surfaces</b>	Object, Face	Use only the top surfaces for the target geometry.
<b>Top surfaces</b>	Object, Face	Use only the bottom surfaces for the target geometry.
<b>Bottom of bounding box</b>	Object, Face	Construct a 'best fit' bounding box around the source geometry and use the bottom surface of that box.
<b>Top of bounding box</b>	Object, Face	Construct a 'best fit' bounding box around the source geometry and use the top surface of that box.
<b>Merge tightly packed points</b>	Object, Face	Combine adjacent points that are close together (within the tolerance distance) to create a simplified approximation of the mesh.
<b>Straighten smooth curves</b>	Object, Face	Combine adjacent faces that are within the tolerance angle of one another and within the specified length range to create a simplified approximation of the mesh.
<b>Extrude up / out of plane</b>	Face, Region, Edge, Line	Extrude or grow the selected component upwards (or out of the plane in the case of drawings).
<b>Expand to rectangle</b>	Edge, Line	Expand or stretch the edge/line into a rectangle (in the plane in the case of drawings).
<b>Span two edges</b>	Edge, Line	Create a surface between the two selected edges/lines.
<b>Ifc door to link</b>	Object, Face	Use the mid line of the bottom faces to expand into a rectangle.
<b>Smooth VCE</b>	Object, Face, Region	Find the top and bottom of the selected geometry, add landings if necessary, and replace the geometry between the landings with a smooth surface.
<b>Stepped VCE</b>	Object, Face, Region	Find the top and bottom of the selected geometry, add landings if necessary, then use the top surfaces only between the landings.



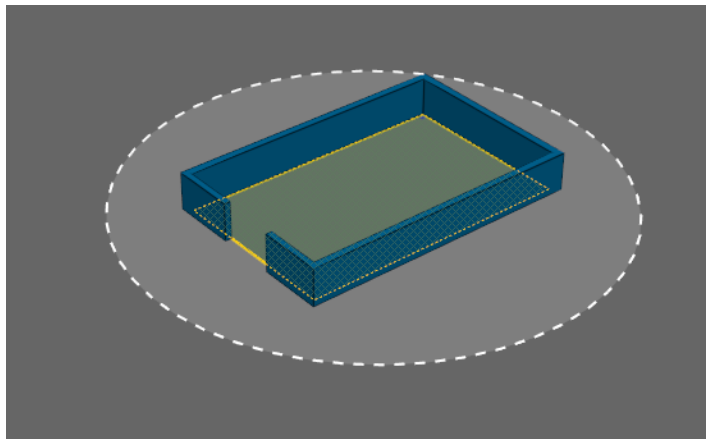
Example of applying the default 'Stepped VCE' and 'Smooth VCE' operations to an imported stair.

#### 4.2.4.2 Tracing New Objects

It is possible to create new simplified objects by tracing over existing geometry in the scene. The 'Trace' mode is available through a button in the main window [tool panel](#) or the [keyboard shortcut](#) 'I'.

##### Trace Mode

In trace mode, a series of lines are drawn by clicking on points in the scene. Each new point creates a line joined to the last. Once there are two non collinear lines, a tracing plane is formed and all subsequent points are projected onto this plane.



Tracing the inside of a room to create a floor. The yellow lines are the traced lines. The green area is the region enclosed by the lines. The circle indicates the tracing plane.

##### Generating Objects

Traced lines can be used to generate objects using the 'Generate' button in the [Creation Widget](#). When the lines completely enclose one or more areas, those areas are used to generate the object. When the traced lines do not enclose any areas, the lines themselves are used. Once lines or areas have been used to generate an object, the lines and areas are deleted.

The transformation operations applied to the lines or areas are taken from the drawing layer line or



drawing layer region settings in the [Generate Options](#) dialog.

### Tracing Within Drawings

It is possible to simulate tracing by [drawing new lines](#) within an existing [drawing layer](#) object. This involves selecting an existing drawing layer, or creating a new drawing layer and positioning it at the appropriate location. Lines drawn in this way persist within the drawing layer and so can be modified or used more than once.

#### 4.2.4.3 Generating From IFC

When [generating objects](#) from [IFC Geometry](#), the right-click 'Generate -> Auto' option will attempt to choose a target MassMotion type from the source IFC Type. The table below outlines the mapping between types.

IFC Type	IFC Geometry Type	Auto-converts to
Space	IfcSpace	<a href="#">Floor</a>
Slab Floor	IfcSlab_Floor	None - must be explicitly specified
Slab Base	IfcSlab_Baseslab	None - must be explicitly specified
Slab Landing	IfcSlab_Landing	<a href="#">Floor</a>
Slab Unrecognized	IfcSlab_Unrecognized	None - must be explicitly specified
Elevator	IfcTransportElement_Elevator	None - must be explicitly specified
Escalator	IfcTransportElement_Escalator	<a href="#">Escalator</a>
Moving Walkway	IfcTransportElement_MovingWalkway	<a href="#">Escalator</a>
Unrecognized Transport Element	IfcTransportElement_Unrecognized	None - must be explicitly specified
Door	IfcDoor	<a href="#">Link</a>
Stair	IfcStair	<a href="#">Stair</a>
Stair Flight	IfcStairFlight	<a href="#">Stair</a>
Wall	IfcWall	<a href="#">Barrier</a>
Wall Standard Case	IfcWallStandardCase	<a href="#">Barrier</a>
Railing	IfcRailing	<a href="#">Barrier</a>

<b>Column</b>	IfcColumn	<a href="#">Barrier</a>
<b>Furnishing Element</b>	IfcFurnishingElement	<a href="#">Barrier</a>
<b>Plate</b>	IfcPlate	None - must be explicitly specified
<b>Building Element Proxy</b>	IfcBuildingElementProxy	None - must be explicitly specified
<b>Ramp</b>	IfcRamp	<a href="#">Ramp</a>
<b>Ramp Flight</b>	IfcRampFlight	<a href="#">Ramp</a>
<b>Other</b>	IfcElement	None - must be explicitly specified

#### 4.2.4.4 Generating From Drawings

[Drawing Layers](#) can be used to create objects through selection of drawing lines or enclosed [regions](#). See [Working with Drawings](#) and [Drawing Commands](#).

##### Lines

When generating from lines, new objects will typically be extruded up out of the plane of the drawing, expanding into surfaces in the plane of the drawing, or constructed as a span between two selected lines. Objects can be generated using the ['Generate' Context Menu](#), or from the [Creation Widget](#).

##### Regions

When generating from [regions](#), it may be necessary to first ensure that all regions are fully closed (see [Healing Drawings](#)). A ['Copy to'](#) command from the right click menu will create an exact copy of the selected region(s), while a ['Generate'](#) command will apply appropriate transformations.

#### 4.2.4.5 'Generate' Context Menu

The 'Generate' context menu is available when right-clicking on a selection of [reference geometry](#) objects, mesh faces, mesh edges, drawing lines, or drawing regions. The menu will contain a list of object types. Choosing a type will then apply a transformation to the source geometry and create a new object of the target type.

The transformation applied to the source geometry depends on both the selection mode and the target type of the new object. The following subsections describe the default transformations. It is not possible to change the default transformations when generating from the context menu. To apply custom transformations use the [Creation Widget](#).

##### 4.2.4.5.1 From Reference Geometry

The following transformations are applied when generating a new object from selected [Reference Geometry](#) objects. A description of the transformations can be found in [Generate Options](#).

Target Object	Transformation
<a href="#">Floor</a>	Copy

<a href="#">Barrier</a>	Copy
<a href="#">Portal</a>	Top surfaces, translate up 0.02
<a href="#">Link</a>	Top surfaces, translate up 0.02
<a href="#">Stair</a>	Stepped VCE, translate up 0.02
<a href="#">Escalator</a>	Smooth VCE, translate up 0.02
<a href="#">Ramp</a>	Smooth VCE, translate up 0.02
<a href="#">Cordon</a>	Copy
<a href="#">Volume</a>	Copy
<a href="#">Visual</a>	Copy
<a href="#">Drawing Layer</a>	Project onto drawing

#### 4.2.4.5.2 From Mesh Faces

The following transformations are applied when generating a new object from selected mesh faces. A description of the transformations can be found in [Generate Options](#).

Target Object	Transformation
<a href="#">Floor</a>	Top surfaces
<a href="#">Barrier</a>	Extrude up
<a href="#">Portal</a>	Top surfaces, translate up 0.02
<a href="#">Link</a>	Top surfaces, translate up 0.02
<a href="#">Stair</a>	Stepped VCE, translate up 0.02
<a href="#">Escalator</a>	Smooth VCE, translate up 0.02
<a href="#">Ramp</a>	Smooth VCE, translate up 0.02
<a href="#">Volume</a>	Extrude up
<a href="#">Visual</a>	Copy
<a href="#">Drawing Layer</a>	Project onto drawing
<a href="#">Generic Geometry</a>	Copy

4.2.4.5.3 From Mesh Edges

The following transformations are applied when generating a new object from selected mesh faces. A description of the transformations can be found in [Generate Options](#).

Target Object	Transformation
<a href="#">Expanded Floor</a>	Expand to rectangle
<a href="#">Expanded Portal</a>	Expand to rectangle, translate up 0.02
<a href="#">Expanded Link</a>	Expand to rectangle, translate up 0.02
<a href="#">Expanded Generic Geometry</a>	Expand to rectangle
<a href="#">Extruded Barrier</a>	Extrude up
<a href="#">Extruded Cordon</a>	Extrude up
<a href="#">Extruded Generic Geometry</a>	Extrude up
<a href="#">Spanned Floor</a>	Span two edges
<a href="#">Spanned Portal</a>	Span two edges
<a href="#">Spanned Link</a>	Span two edges, translate up 0.02
<a href="#">Spanned Stair</a>	Span two edges, translate up 0.02, add landings of length 0.5
<a href="#">Spanned Escalator</a>	Span two edges, translate up 0.02, add landings of length 0.5
<a href="#">Spanned Ramp</a>	Span two edges, translate up 0.02, add landings of length 0.5
<a href="#">Spanned Generic Geometry</a>	Span two edges
<a href="#">Drawing Layer</a>	Project onto drawing

## 4.2.4.5.4 From Drawing Lines

The following transformations are applied when generating a new object from selected drawing lines. A description of the transformations can be found in [Generate Options](#).

Target Object	Transformation
<a href="#">Expanded Floor</a>	Expand to rectangle
<a href="#">Expanded Portal</a>	Expand to rectangle, translate up 0.02
<a href="#">Expanded Link</a>	Expand to rectangle, translate up 0.02
<a href="#">Expanded Generic Geometry</a>	Expand to rectangle
<a href="#">Extruded Floor</a>	Extrude up
<a href="#">Extruded Barrier</a>	Extrude up
<a href="#">Extruded Link</a>	Extrude up, translate up 0.02
<a href="#">Extruded Stair</a>	Extrude up, translate up 0.02, add landings of length 0.5
<a href="#">Extruded Escalator</a>	Extrude up, translate up 0.02, add landings of length 0.5
<a href="#">Extruded Ramp</a>	Extrude up, translate up 0.02, add landings of length 0.5
<a href="#">Extruded Cordon</a>	Extrude up
<a href="#">Extruded Visual</a>	Extrude up
<a href="#">Extruded Generic Geometry</a>	Extrude up
<a href="#">Spanned Floor</a>	Span two edges
<a href="#">Spanned Barrier</a>	Span two edges
<a href="#">Spanned Portal</a>	Span two edges
<a href="#">Spanned Link</a>	Span two edges, translate up 0.02

<b>Spanned Stair</b>	Span two edges, translate up 0.02, add landings of length 0.5
<b>Spanned Escalator</b>	Span two edges, translate up 0.02, add landings of length 0.5
<b>Spanned Ramp</b>	Span two edges, translate up 0.02, add landings of length 0.5
<b>Spanned Generic Geometry</b>	Span two edges
<b>Drawing Layer</b>	Project onto drawing

4.2.4.5.5 From Drawing Regions

The following transformations are applied when generating a new object from drawing layer [regions](#). A description of the transformations can be found in [Generate Options](#).

Target Object	Transformation
<a href="#">Floor</a>	Copy
<a href="#">Barrier</a>	Extrude up
<a href="#">Portal</a>	Copy
<a href="#">Link</a>	Copy
<a href="#">Stair</a>	Stepped VCE, translate up 0.02
<a href="#">Escalator</a>	Smooth VCE, translate up 0.02
<a href="#">Ramp</a>	Smooth VCE, translate up 0.02
<a href="#">Cordon</a>	Copy
<a href="#">Volume</a>	Copy
<a href="#">Generic Geometry</a>	Copy

4.2.4.6 'Copy to' Context Menu

The 'Copy to' context menu is available when right-clicking on a selection of mesh faces or drawing regions. A new object of the specified type is created with an exact copy of the selected geometry.

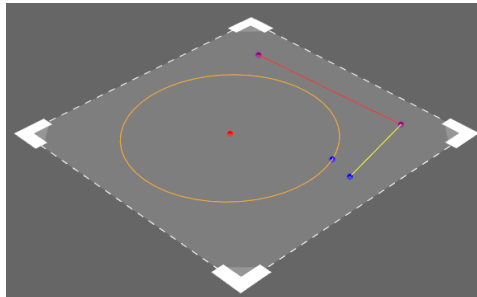
## 4.2.5 Working with Drawings

A drawing layer object can be translated, scaled and rotated much like any other 3D object in the scene. Adding or editing points or lines within a drawing can be done from within the 'Edit Drawing' mode.

### Editing a Drawing

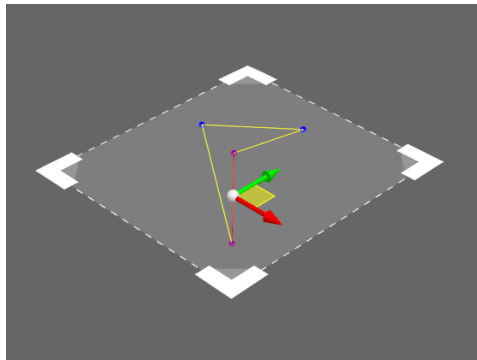
To enter 'Edit Drawing' mode, select a drawing layer object and then use the 'Edit Drawing' mode button in the [tool panel](#) or use the 'D' shortcut key. Lines and points can only be edited in drawing mode.

The points and lines are highlighted in red when selected, much like vertices, edges and faces for other scene objects. The connected lines of a point will additionally be highlighted in orange and the connected points of a line will be highlighted purple. This can help identify points that are conceptually but not physically connected to a line as in the case with the centre point of a circle.



Selecting the centre point of a circle highlights the circle orange. Selecting a straight line turns its end points purple. Unselected points and shapes are blue and yellow respectively.

The [manipulator](#) can be used to transform points and lines in the 2D plane.



Translating drawing elements in the plane

### Regions

Regions are empty spaces within a drawing that are completely enclosed by lines. Regions can be used to generate floor, barrier, or other scene objects. See [Finding Regions](#) for more information.

#### 4.2.5.1 Adding Shapes

To add shapes or lines to a drawing begin by entering 'Edit Drawing' mode (see [Working with Drawings](#)). Lines can only be added when editing a single drawing.

##### Adding Straight Lines

New straight lines can be drawn directly using the [Draw Lines](#) mode. A straight line can be added between two existing points by selecting the points, right clicking on the points, then using the 'Add line' context menu command.

##### Adding Shapes

New shapes can be added when in 'Line' selection mode by right clicking on empty space within the drawing plane and selecting one of the shapes from the 'New Lines' context menu.

Shape	
Circle	Create a single circular line with one centre point and a point on the radius.
Ellipse	Create a single elliptical line with one centre point and then a point on each of the major and minor axis.
Square	Create four straight lines connected to form a square.

#### 4.2.5.2 Drawing Lines

To draw new lines in a drawing layer, select a single drawing and enter 'Draw Lines' mode. 'Draw Lines' mode is accessible through the 'L' hotkey, the drawing object right-click context menu, or through a button in the [tool panel](#) when in 'Edit Drawing' mode.

Note it is not possible to draw lines when multiple drawing layers are selected.

Snapping can be used to trace the faces, edges, or points of other 3D objects or drawings. Created points and lines are projected onto the drawing plane.

Element	Action
Single Line	A) Left click once to create the first point, and left click again to create the second point joined to the first by a line. B) Click and drag anywhere in the drawing plane.
Continuous Line	After drawing the first line segment, hold SHIFT while drawing subsequent lines to create a continuous series of connected line segments.

#### 4.2.5.3 Drawing Commands

The majority of the following operations are available through the right-click context menu. Object commands appear when an object is selected and line/point commands appear when in drawing mode and lines/points are selected. In some cases commands are also available through the [tool panel](#).

Object Command	Effect
Generate -> Merged	Create a new drawing that combines the selected drawings. Do



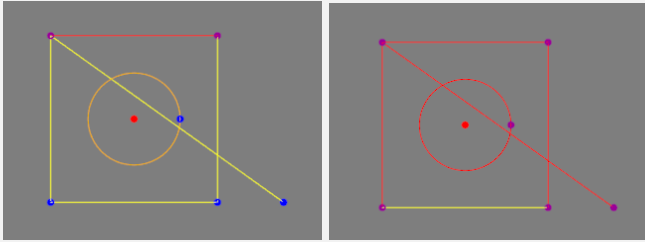
<b>Drawing</b>	<p>not delete the originals.</p> <p>If the drawings are not on the same plane, the drawings will be projected to the plane of the first selected drawing.</p>
<b>Merge</b>	<p>Create a new drawing that combines the selected drawings and delete the original drawings.</p> <p>If the drawings are not on the same plane, the drawings will be projected to the plane of the first selected drawing.</p>
<b>Edit Lines</b>	Changes the selection mode to 'Lines' so that the lines and points of the drawing can be edited.
<b>Find Regions</b>	<p>Combine all selected drawing layers and use the lines to find enclosed areas. These areas are presented as regions that can then be used to generate scene objects (see <a href="#">Finding Regions</a>).</p> <p>Only drawing layers on the same plane are combined together when finding regions.</p>
<b>Simplify</b>	Remove any duplicate lines or unnecessary points in the selected drawings. No points are moved.

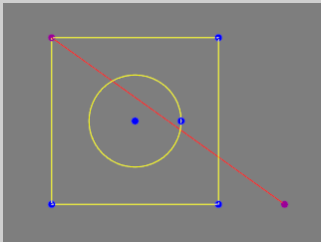
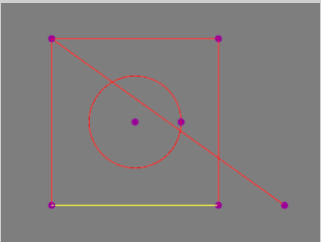
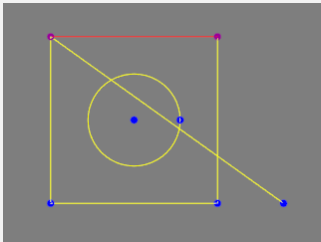
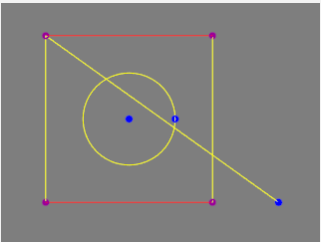
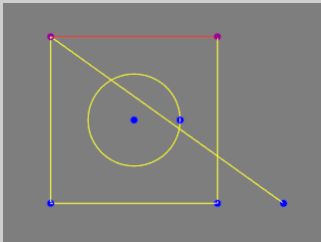
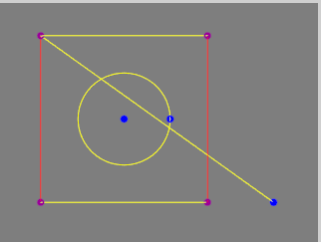
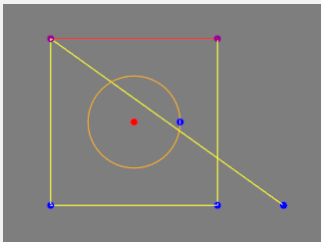
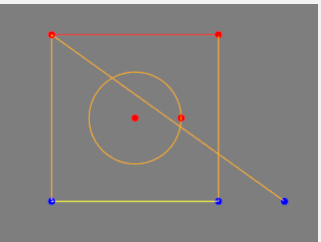
<b>Line/Point Command</b>	<b>Target</b>	<b>Effect</b>
<b>Delete</b>	Points, lines	Delete the selected points and lines. If points are deleted, any connected lines are also deleted. If lines are deleted, any points no longer connected to another line are also deleted.
<b>Discretize</b>	Lines	Split each selected line into a number of straight line segments.
<b>Duplicate</b>	Lines	Create a copy of the lines and their connected points in the same drawing. This is useful for creating repeating patterns.
<b>Extend and Fuse</b>	2 or more straight lines	<p>Extend the selected straight lines until they intersect with each other then fuse them. See: Fuse.</p> <p>Shortcut: 'U'</p>
<b>Explode</b>	Points, lines	Break a point connected to multiple lines into several points, effectively disconnecting the lines. If a point is used multiple times by the same lines, it will also be split. Exploding a line will explode all the points it's connected to.
<b>Extract</b>	Points, lines	Remove the selected points and lines from the current drawings and use them to create new drawing layer objects.
<b>Find Regions</b>	Lines	Combine all selected lines from all selected drawing layers and use them to find enclosed areas. These areas are presented as regions that can be used to generate scene objects (see <a href="#">Finding Regions</a> ).

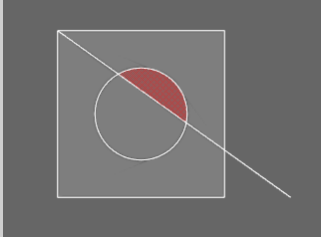
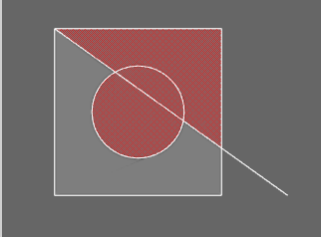
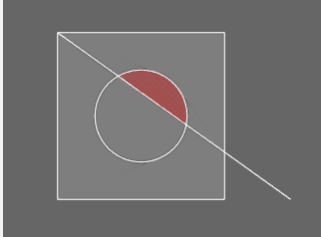
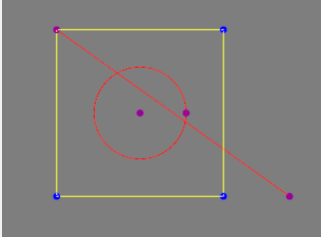
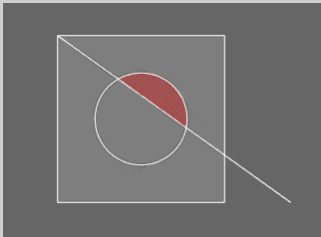
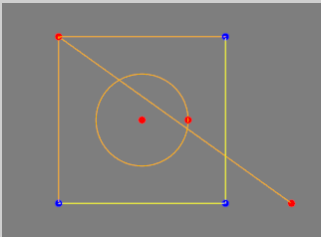
<b>Fuse</b>	2 or more straight lines	Connect lines together where they cross by splitting the lines, adding a new point at the intersection, then joining the new lines together at the point.
<b>Merge</b>	2 or more points, 2 or more straight lines	<p>Merge nearby points into each other. Any lines connected to the selected points will be connected to the merged points.</p> <p>If straight lines are selected, sufficiently "parallel" lines will be merged into a single line. Two straight lines will be merged if they are close to each other and if both of the shorter line's end points are close enough to the longer line projected to infinity.</p> <p>For both merging points and straight lines, a tolerance can be specified. This operation may cause lines to shift around.</p>
<b>Simplify</b>	Points, lines	Remove duplicate lines and unnecessary points. No points are moved.
<b>Split</b>	Straight lines, circles, circle arcs	Splits the lines in two. Circles and circle arcs will be split into circle arcs.

**Selection Commands**

The selection commands can be used to select additional components that are associated in some way with the target component(s). The selection commands can be useful when searching for gaps in enclosed spaces.

Command	Target	Effect
<b>Select: Connected Lines</b>	Points and Lines	<p>Select all lines which are connected to selected points or are connected to a point connected to the selected lines.</p>  <p style="text-align: center;"><b>Before</b>                      <b>After</b></p>
<b>Select: Intersecting Lines</b>	Lines	Select all lines that intersect with the selected lines. Lines that touch at end points count as intersecting. Lines that share a connected point but do not actually cross are not counted.

		 <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Parallel Lines</b>	Single Straight Line	<p>Select all straight lines that are parallel to the selected line.</p>  <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Perpendicular Lines</b>	Single Straight Line	<p>Select all straight lines that are perpendicular to the selected line. The initial line is deselected.</p>  <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Connected Points</b>	Points and Lines	<p>Select all points which are connected to a selected line or are connected to a line connected to a selected point.</p>  <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Connected Regions</b>	Regions	<p>Select all regions which are adjacent to the selected regions. Corner adjacency is not counted.</p>

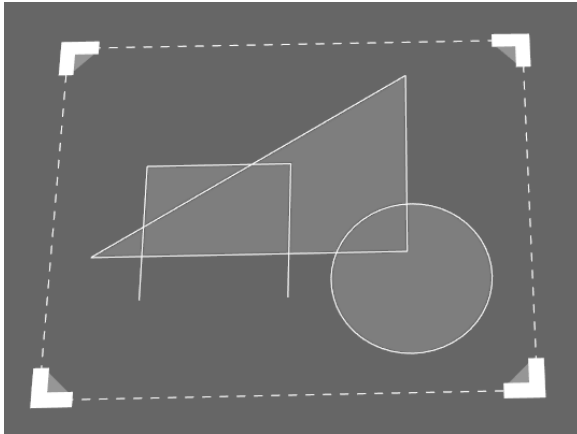
		 <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Connected Lines</b>	Regions	<p>Select all lines which define the selected regions.</p>  <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>
<b>Select: Connected Points</b>	Regions	<p>Select all points connected to lines which define the selected regions.</p>  <p style="text-align: center;"><b>Before</b></p>  <p style="text-align: center;"><b>After</b></p>

#### 4.2.5.4 Finding Regions

Drawing regions are areas in a drawing layer that are fully enclosed by lines. Once found, regions can be selected and used to generate scene objects such as floors, stairs, or barriers. To fill the regions in a drawing select a drawing then enter the "Find Regions" mode from the [tool panel](#) or press the 'R' shortcut key.

Lines do not need to be connected together to enclose a region. Two unrelated lines that happen to intersect can form one corner. However, a region can only be formed when a space is completely enclosed. See [Healing Drawings](#) for techniques on how to identify and close gaps in region boundaries.

Regions can be from all lines in a single drawing, from a subset of lines within a drawing, or from the combined set of lines from multiple drawings.



Six regions enclosed by various lines

#### 4.2.5.5 Healing Drawings

Drawings can be used to create geometry either from the lines within the drawing or the empty spaces (called [regions](#)) enclosed by those lines (see: [Copying and Converting Geometry](#)).

When empty spaces are not completely enclosed, some healing may be required before they can be treated like regions. See: [Drawing Commands](#) for a full list of commands available.

##### Delete Unwanted Detail

Examine each drawing layer and delete any layers that do not contain information useful for generating scene objects. Electrical systems, duct work, labeling, grid lines... anything that does not contribute to the definition of the physical space.

Extra lines within a drawing layer can also add unnecessary complexity to regions, subdividing a space further than is useful. Delete unwanted lines before finding regions.

##### Merge Multiple Drawings

Some drawing operations are only possible when editing a single drawing. If the information needed to define a floor/door/stair is spread over multiple drawing layers, consider merging those layers into a new combined layer. A standard merge will delete the source layers. To preserve the source layers use the right-click context menu and select 'Generate - > Merged Drawing'.

##### Close Gaps

See [Drawing Commands](#) for a full list of commands that can be used to help ensure [regions](#) are well formed. Each of these commands is available through the right-click menu when editing a drawing in the 'Lines' selection mode.

- **Add Line:** Obvious gaps can be filled by selecting two points on either end of the gap and adding a line between them with 'Add Line'.
- **Extend and Fuse:** Select two lines that are almost touching and use 'Extend and Fuse' to extend the lines until they intersect. The two lines will be joined by a new point at the intersection location.
- **Merge Points:** Points that are very close together can be merged into a single point using 'Merge'.

##### Performance Tips

If the 'Find Regions' operation is taking a long time to complete, try selecting only a subset of lines within the drawing layers. Only those lines selected will be used to construct regions. Or, merge multiple drawing layers into one to take advantage of region caching. Regions are cached when operating on a single drawing layer and will persist until the drawing layer is modified.

## 4.2.6 Editing Object Properties

Object properties can be edited using the Properties Window. This dialog is available through the object's right-click menu, or by double-clicking on the object in the scene or list views. Objects which share common properties can have those properties edited in batch all at once using [object multi-edit](#).

Properties Window Toolbar	
<b>File</b>	For some query types, this allows the results to be saved to an image or CSV file. For <a href="#">agent actions</a> , the action graph image can be exported. Additionally, for any object type, this allows the current object(s) to be <a href="#">exported</a> to an XML file for use in another project.
<b>Select</b>	Select the objects currently being edited. This can be useful in combination with the list view, scene view, or when <a href="#">choosing objects</a> .
<b>Multi</b>	Choose which objects are currently being edited (see <a href="#">Editing Multiple Objects</a> for details).
<b>Generate/Evaluate</b>	Used by <a href="#">graph</a> , <a href="#">map</a> , and <a href="#">table</a> queries to calculate results.
<b>Help</b>	Display the help page for the inspected object.
<b>Arrow</b>	For events which use tables to input data, or for <a href="#">table</a> and <a href="#">graph</a> queries which generate data, the arrow will show or hide the data.

### Object Header

The header consists of a colour swatch, object name, and object indicator. This header is visible for all object types regardless of the properties.

Object Header	
<b>Colour Swatch</b>	The object colour can be modified by clicking on the colour swatch. Left click to bring up the colour chooser. Right click to pick a specific colour or reset to the object's default colour. See <a href="#">Working with Colours</a> .
<b>Name</b>	Valid object names must begin with a letter then be followed by any combination of letters, numbers, underscores, or dashes.
<b>Object Indicator</b>	The object indicator displays the status of the object. If all properties in the object are valid the indicator will be green. Red indicates that at least one property is invalid. Right-clicking on the indicator can be used to reset all object properties to their default values.

### Properties

Below the name field is a list of properties for the particular object being inspected. Beside each property is an indicator. The indicator changes colour depending on the state of the property: ● green for valid, ■ red for invalid.

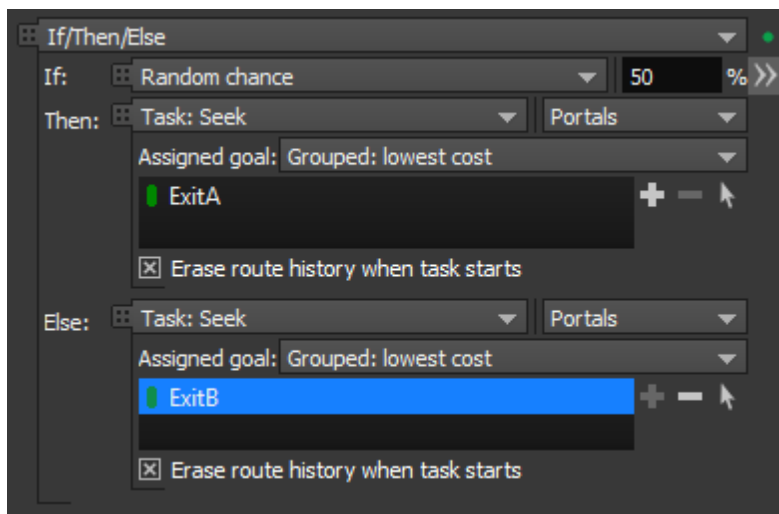
It is possible to reset a property to its default value, copy the value, paste a value, or set the value from another source object, all through the right-click menu of the indicator.

Property values can be copied within an object or even between objects by dragging the indicator of the source property onto the destination property.

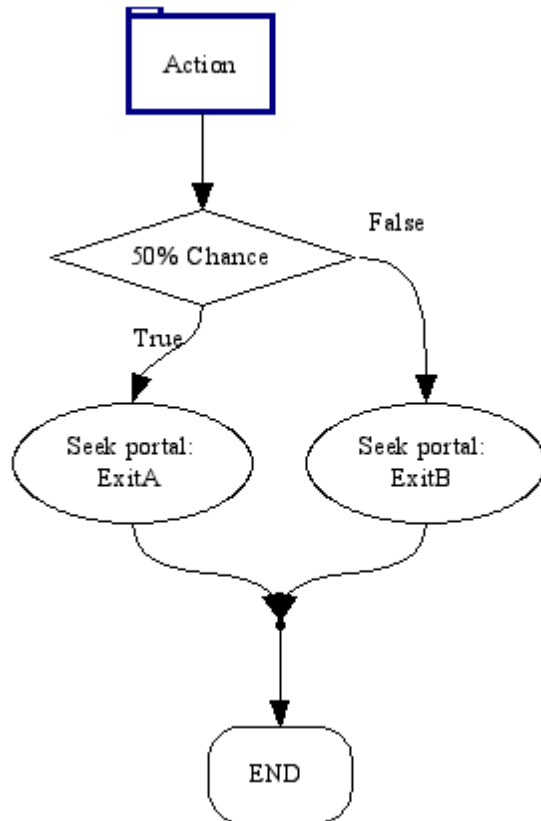
### Action Properties

Many objects can apply actions to agents (see [Where to Use Actions](#)). For action properties, additional information is available by toggling the arrow button below the indicator.

If the action is valid, a flow diagram will be displayed which shows how the action functions. If the action is invalid, a list of errors will be displayed.



An action property. The arrow is on the right, below the indicator.



Flow diagram of the action.

#### 4.2.6.1 Editing Multiple Objects

MassMotion supports editing the properties of many objects simultaneously. Select multiple objects and then use the right-click menu to select 'Properties'. Or inspect the properties of a single object, then use the 'Multi' button at the top of the window to add objects to the current session.

The editor dialog allows editing of any properties that the chosen objects have in common. When editing objects of similar type all properties are available. If multi-editing a floor and link, only shared properties such as the map resolution can be modified. In all cases, setting the value of a property will set that value on all objects.

##### Editing Names

When inspecting the properties of multiple objects, the name entered will be applied to all objects being edited. Numeric suffixes will be added as needed to ensure that all names remain unique. For instance, if three objects are selected, then:





- Entering a name of 'NewName' will result in the objects being renamed to 'NewName', 'NewName1', and 'NewName2'
- Entering a name of 'NewName1' will result in the objects being renamed to 'NewName1', 'NewName2', and 'NewName3'
- Entering a name of 'NewName10' will result in the objects being renamed to 'NewName10', 'NewName11', and 'NewName12'

##### Property Indicators

The small coloured property indicators to the left of each property field are especially important when



editing multiple objects. Different colours and shapes of the property indicator indicate different states:

Indicator appearance	Meaning
 Green circle	All edited objects have the same valid value for this property.
 Yellow triangle	All edited objects have valid values for this property but they are not all the same.
 Red square	All edited objects have invalid values for this property.
 Split yellow/red square	Some edited objects have invalid values for this property and some have valid values.

Right-clicking on the property indicator allows the property value from a specific object to be copied to all edited objects. 'Set from' brings up a dialog allowing any one of the currently-edited objects to be selected; that object's value for the current property will then be copied to all other objects. 'Set from any' works the same way but allows any object in the current project to be chosen.

#### 4.2.7 Process Chains

A process chain is a series of one or more servers connected together. A process chain has one or more start points (green circles) through which agents may enter the chain, and one or more end points (blue circles) where agents are released from the chain. The most basic process chain (shown in Figure 1-1) consists of a single server with one start point and one end point.

The most simple compound process chain (shown in Figure 1-2) consists of two servers connected together, with a single start point, a single end point, and a single internal connection. Agents following the internal connection from one server to another will only proceed to the connected server when it has the available capacity. If there is no available capacity, agents will wait at the upstream processor, potentially blocking other agents from accessing the upstream processor.

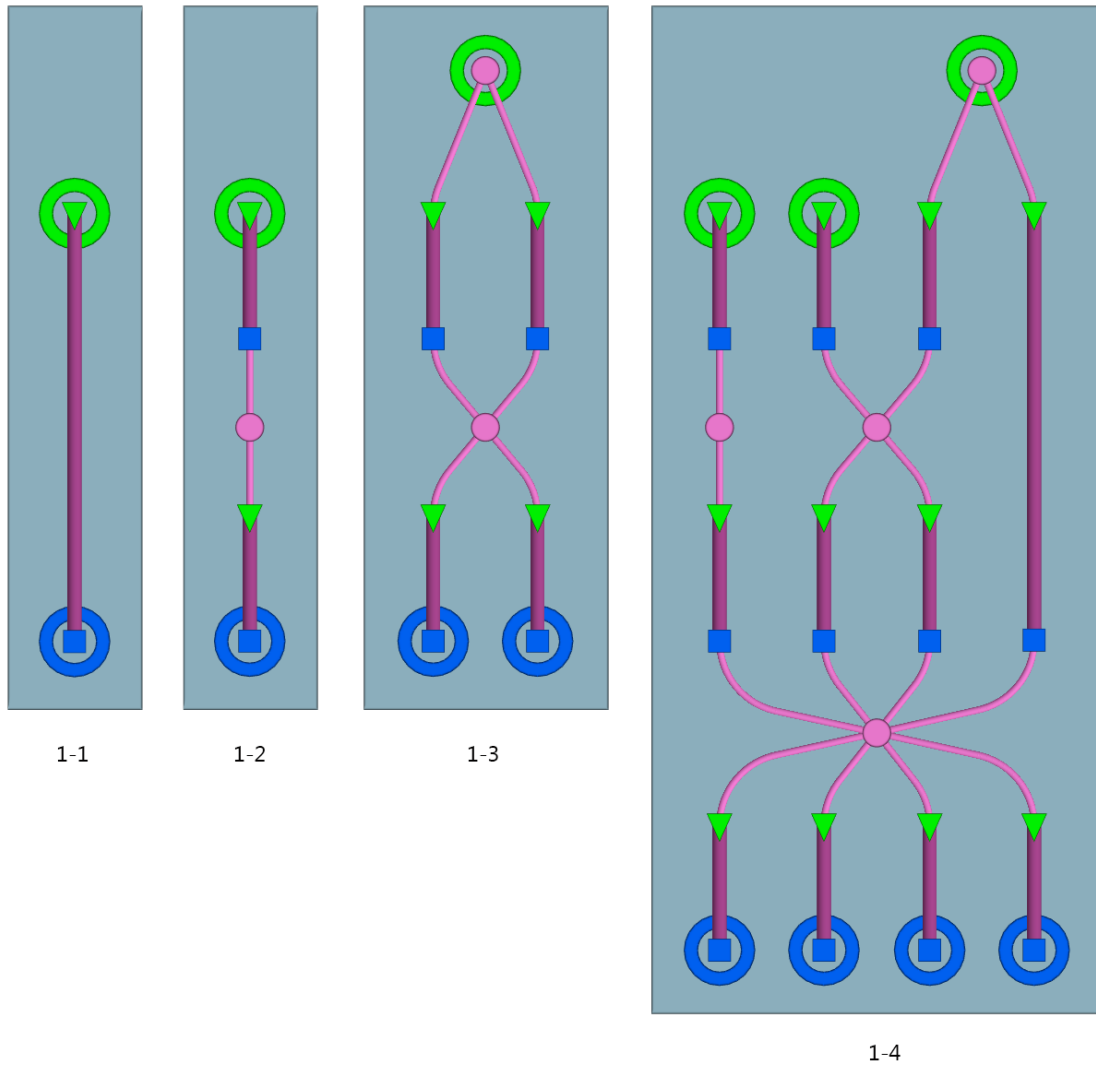


Figure 1

**Servers**

Servers form the core building blocks of a process chain. Each server is composed of an input buffer line (green triangle at entry point and purple path geometry) and a terminating process node (blue square at exit point). Agents walk and queue along the input buffer until they reach the process node and will only leave the server once processing is complete. The amount of time taken to process a single agent can be specified through the server properties. The server can also limit access to specific agents based on token possession, and limit the maximum number of agents permitted in the input buffer. A single server is shown in Figure 1-1. For more information on server construction and properties, see [Servers](#).

**Dispatches**

Dispatch objects are automatically created when server objects are connected to each other. The dispatches are represented by pink discs with pink connector lines indicating which server objects are connected. Upstream servers are referred to as sources and downstream servers as sinks. Figure 1-3 shows how server entry points can be grouped together using a dispatch and also how successive groups of server exit points and entry points can be connected in a many-to-many configuration via a single dispatch. Figure 1-4 shows how multiple levels of servers may be connected using many servers and dispatches to define complex operations. The dispatch objects

control the distribution pattern of agents (random or shortest queue) from sources to sinks. For more information on dispatch properties please see [Dispatches](#).

### **Movement Within a Process Chain**

Agents that enter a server cease to navigate via the [normal crowd movement](#) algorithms and will instead advance along the input buffer in a strictly ordered fashion. When in transit between two internal servers, agents will employ normal crowd movement behavior, attempting to avoid obstacles and each other, but will not engage in any route selection or wayfinding. For this reason all servers in a process chain must be located on the same [floor](#).


### **Measuring Counts and Times in a Process Chain**

Please see [Analysis Objects](#) for information on the various metrics available for measuring the performance of process chains.




#### **4.2.7.1 Editing Process Chains**

To create or edit process chains first construct and arrange the desired [Servers](#).

##### **Process Chain Edit Model**

Once the servers are in the desired configuration they can be connected together by entering the "Process Chain Edit" mode. This can be done by either clicking on the  icon in Model Panel on the right hand side of the main window or by using the hot-key "Q". In edit mode, all server start and end points and all dispatches in the model will be outlined in yellow circles. Yellow circles identify points that can be connected together

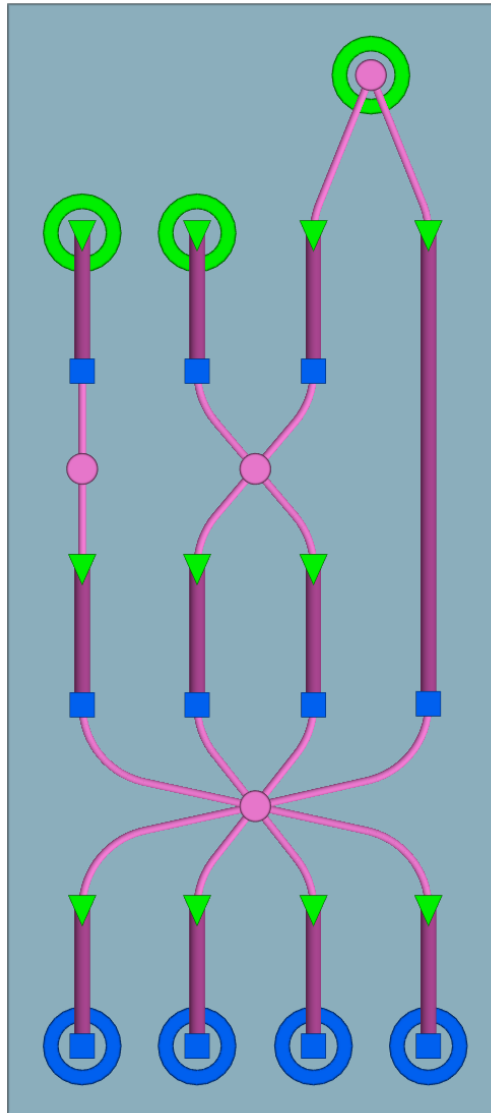
##### **Connecting/Disconnecting Servers**

To connect servers together, select the desired servers and click the  Connect icon in the Model Panel. Dispatches will automatically be created between server start and end points as needed. To disconnect process elements from a process chain, click the  Disconnect icon in the Model Panel. Connecting and disconnecting process chain elements may also be done by right clicking the selected 3D objects and selecting the appropriate item in the pop-up menu. To exit "Process Chains Edit" mode click on the  icon in Model Panel on the right hand side of the main window or press the hot-key "Q".

#### **4.2.7.2 Sending Agents to Process Chains**

Process chain entry points (server or dispatch) are identified by green circles. Agents can be directed to a single entry point using the "Task: Seek process start" [action](#). Once an agent is tasked with seeking a process chain entry it will navigate the scene using the standard route choice algorithms until it reaches the floor with the process chain, at which point it will attempt to enter and follow the chain.

If the entrances to servers in the chain are left unconnected, separate actions are required to explicitly send agents to the individual entry servers. If the servers are connected by a single entry dispatch, all agents can be sent to that single dispatch, and then the dispatch will distribute agents across the connected servers. For example, in Figure 1-4 there are three start point entries into the process chain (two servers and one dispatch). Agents can be explicitly directed to any of the three using actions. However, those agents that are directed to the third start point will be evenly distributed across the third and fourth server.



A complex process chain with multiple entry points.

### 4.2.8 Choosing Objects

Authoring frequently involves choosing one or more objects for a particular property or operation.

#### Single Objects

Clicking on the 'mouse cursor' icon beside a single-object field (or on the field itself if it is blank) will bring up a dialog with a list of all valid objects for that field. Alternatively, if the desired object has already been selected elsewhere, the 'crosshair' icon beside the field can be used to choose that object. For instance, a [simulation run](#) could be selected in the list view on the left side of the main window and then the crosshair could be used to choose that simulation run.

#### Multiple Objects

Similar to single objects, clicking on the mouse cursor icon or in an empty multi-object field will bring up a separate selection dialog. The two arrows in this dialog can be used to move valid objects between the **Available** and **Selected** columns. Alternatively, objects that are currently selected can be directly added or removed to or from the chosen objects field using the plus or minus buttons. For instance, one or more links could be selected in the 3D view and then added or removed from a bank

by using the plus and minus buttons.

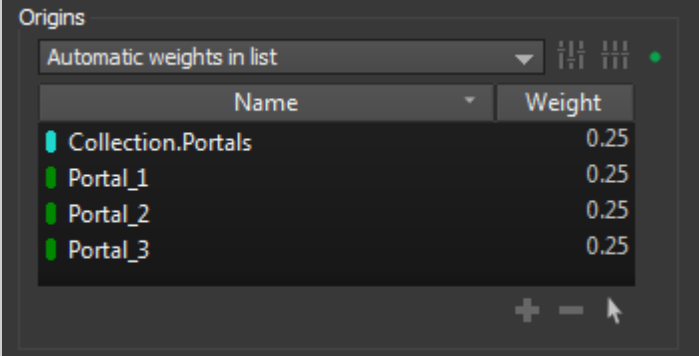
**Object Order**

In some cases, the order of chosen objects is important, such as when defining a [trip](#). In these cases the multi-object field will additionally have up and down arrows that can be used to change the order of selected objects. If arrows are not displayed, then order is not significant.

**Object Weights**

Chosen objects can sometimes have weights assigned to them, such as the origin [portals](#) in a [journey event](#). In the following table, assume that three individual portals and one collection of two portals have been set up. There are then three options for how to assign and use weights:

Weighting Type	Meaning
<p><b>All equally likely</b></p>	<p>All individual objects in the list, and all member objects of any collections in the list, are merged into a single large set and weighted equally. In the example below, each individual portal would have a weight of 0.2 since there are a total of 5 portals.</p> <p><b>Weights:</b> Portal_1 = 0.2, Portal_2 = 0.2, Portal_3 = 0.2, Portal_4 = 0.2, Portal_5 = 0.2</p> <div data-bbox="544 904 1246 1263" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p>Objects</p> <p>Ignore weights in list (and collections) <span style="float: right;">        ●</span></p> <ul style="list-style-type: none"> <li>● Portal_4</li> <li>● Portal_5</li> </ul> <p style="text-align: right;">+ - ↗</p> <p style="text-align: center;">Find All Members</p> </div> <p style="text-align: center;"><b>Collection members ignoring weights</b></p> <div data-bbox="544 1301 1246 1659" style="border: 1px solid black; padding: 5px;"> <p>Origins</p> <p>Ignore weights in list (and collections) <span style="float: right;">        ●</span></p> <ul style="list-style-type: none"> <li>● Collection.Portals</li> <li>● Portal_1</li> <li>● Portal_2</li> <li>● Portal_3</li> </ul> <p style="text-align: right;">+ - ↗</p> </div> <p style="text-align: center;"><b>Journey origins ignoring weights</b></p>
<p><b>Automatic weights</b></p>	<p>Each item in the list is weighted equally; in the example below the three top-level portals and the collection would each have a weight of 0.25. For any item that is a collection, the collection's weight is then divided between the collection's individual member items based on each item's weighting within the collection. Again in the example below, the two portals inside the collection would both have a weight of 0.125 (half of 0.25 each).</p>

Weighting Type	Meaning
	<p><b>Weights:</b> Portal_1 = 0.25, Portal_2 = 0.25, Portal_3 = 0.25, Portal_4 = 0.125, Portal_5 = 0.125</p>  <p style="text-align: center;"><b>Collection members with automatic weights</b></p>  <p style="text-align: center;"><b>Journey origins with automatic weights</b></p>
<p><b>Manual weights</b></p>	<p>Each item in the list is assigned a manual weight; the two 'equalizer' buttons can be used to quickly set all weights equal or normalize them (force the sum to 1 while keeping the same relative proportions). For any item that is a collection, the manual weight assigned to the collection is then divided between the collection's individual member items based on each member's weighting within the collection.</p> <p>In the example below, the three top-level portals would keep their weights of 0.2 each while the two portals within the collection would end up with weights of 0.1 and 0.3 (one-quarter and three-quarters of 0.4 respectively).</p> <p><b>Weights:</b> Portal_1 = 0.2, Portal_2 = 0.2, Portal_3 = 0.2, Portal_4 = 0.1, Portal_5 = 0.3</p>

Weighting Type	Meaning																
	<div data-bbox="544 264 1246 622"> <p>Objects</p> <p>Manual weights in list</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Weight ^</th> </tr> </thead> <tbody> <tr> <td>Portal_4</td> <td>0.25</td> </tr> <tr> <td>Portal_5</td> <td>0.75</td> </tr> </tbody> </table> <p>Find All Members</p> <p><b>Collection members with manual weights</b></p> </div> <div data-bbox="544 658 1246 1016"> <p>Origins</p> <p>Manual weights in list</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Weight</th> </tr> </thead> <tbody> <tr> <td>Collection.Portals</td> <td>0.4</td> </tr> <tr> <td>Portal_1</td> <td>0.2</td> </tr> <tr> <td>Portal_2</td> <td>0.2</td> </tr> <tr> <td>Portal_3</td> <td>0.2</td> </tr> </tbody> </table> <p><b>Journey origins with manual weights</b></p> </div>	Name	Weight ^	Portal_4	0.25	Portal_5	0.75	Name	Weight	Collection.Portals	0.4	Portal_1	0.2	Portal_2	0.2	Portal_3	0.2
Name	Weight ^																
Portal_4	0.25																
Portal_5	0.75																
Name	Weight																
Collection.Portals	0.4																
Portal_1	0.2																
Portal_2	0.2																
Portal_3	0.2																

## 4.2.9 Working with Colours

### Individual Colours

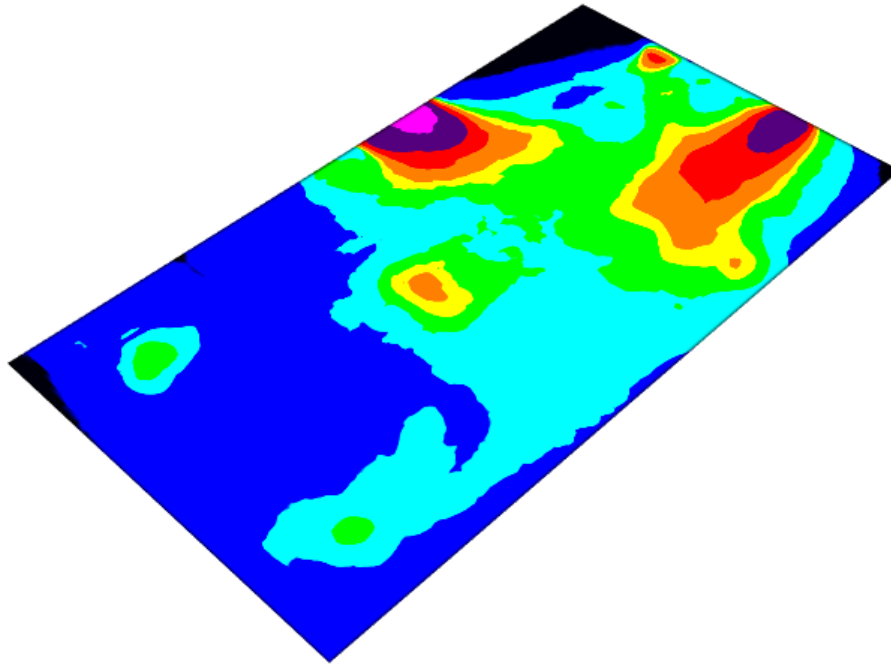
Colour 'swatches' are used throughout MassMotion wherever a colour should be specified (such as the colour of an object in the 3D scene, or the colour of an individual series in a [graph](#)). In all such cases, the colour swatch shows a preview of the colour and can be clicked to bring up a full-featured colour dialog. Alternatively, the swatch can be right-clicked to select from a small number of built-in colours or reset the colour to its default value (e.g., light blue for a [floor](#)).

### Alpha or Transparency

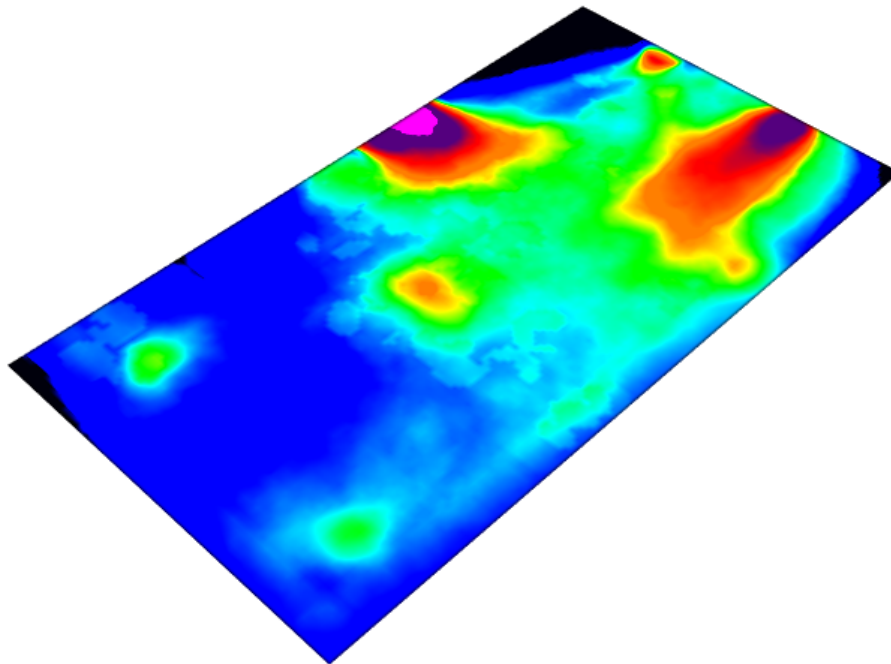
Transparency can be set by changing the 'Alpha channel' in the colour dialog; an alpha channel value of 255 is fully opaque while a value of 0 is fully transparent (invisible).

### Colour Contours and Gradients

It is sometimes necessary to define a mapping between numerical values (such as time or density values) and colours, such as in some [map queries](#). For instance, the Fruin and IATA standards described in [LOS Colour Mapping](#) define colours associated with different ranges of agent density. In MassMotion, custom colour mappings can be defined by specifying a list of colours with cut-off values in between colours. In most cases, these cut-off values and colours can then be visualized as either discrete colour bands with sharp contours, or as smooth gradients.



**Contour colouring**

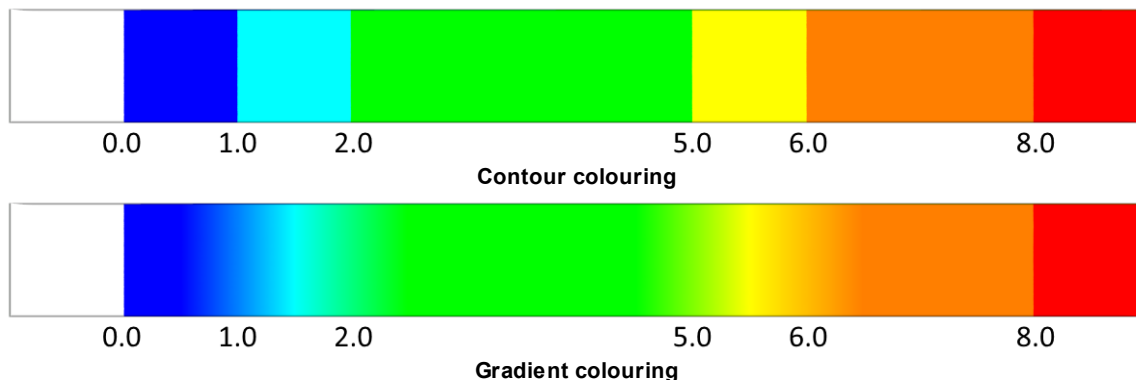


**Gradient colouring**

The number of colours is always one more than the number of cut-off values. The first colour is used for all values less than the first cut-off value and the last colour is used for all values greater than the last cut-off value. If sharp contours are used, the second colour is used for all values between the first and second cut-off value, the third colour is used for all values between the second and third cut-off values, and so on. If a gradient is used, colour values are smoothly interpolated across each cut-off value except for the first and last, which remain as sharp cut-offs. For example, the images below show the relationship between numerical value and colour for cut-off values of 0.0, 1.0, 2.0, 5.0, 6.0



and 8.0. In the images above, note that the black and pink contours remain sharp even in the gradient version. (In many cases the lower sharp cut-off will not be visible since it typically has an associated cut-off value of 0.0, and often no values are less than or equal to 0.0.)



The gradient behaviour is designed so that it is obvious (via a sharp edge) when a value falls outside of the expected range. To avoid the sharp cut-off at each end of a gradient, the recommended method is typically:

- At the lower end, make the first two colours the same. In the above example, blue could be used for both 'less than or equal to 0.0' and 'between 0.0 and 1.0'. Note that this means it will no longer be possible to tell the difference between a small value and 'no data'.
- At the upper end, add an extra cut-off value and associated colour; the cut-off value chosen should be one that you do not expect to ever reach. In the above example, a cut-off value of 12.0 could be added with an associated colour of bright pink. This will cause the orange and red to blend together over the cut-off value of 8.0; however, if the measured value (density, distance, time, cost etc.) ever exceeds 12.0, a sharp cut-off to bright pink will be visible, perhaps indicating a dangerously high population density or unacceptable time to exit during an evacuation.

The repeated-colour technique can also be used at the upper end, but the technique recommended above has the advantage of clearly indicating when the simulation has exceeded expected bounds.

#### 4.2.10 Working with Time

Working with MassMotion often involves specifying individual times, durations, or time ranges, such as the time a particular event should fire or the time range over which a group of agents should enter the simulation. Times and durations can either be entered explicitly in the form HH:MM:SS or using a variety of shortcuts as shown in the table below:

Entered text	Resulting time
0	00:00:00
5	00:00:05
10s	00:00:10
2m	00:02:00
90s	00:01:30
1.5h	01:30:00
12:15	12:15:00

Entered text	Resulting time
0:1:5	00:01:05
-30m	-00:30:00 (useful for specifying offsets, see below)

Note that:

- Combined shortcuts such as 1h30m are not supported but can be entered as 90m or 1.5h instead.
- A single number is interpreted as a count in seconds, but a time such as 1:30 is interpreted as hours and minutes.

**Time Ranges**

Where a simple time range is required, it can be specified as:

Description	Meaning
All available	The entire simulation
All before	From the beginning of the simulation to the specified time
All after	From the specified time to the end of the simulation
Specified interval	From the specified start time, for the specified duration

**Time References**

In several places, times can be entered as either absolute or relative to another time. These can be specified as:

Description	Meaning
Absolute	A specific time in HH:MM:SS form. This is relative to midnight (00:00:00) on the start day of the simulation. For instance, if the start time of the simulation (from the <a href="#">project settings</a> ) is set to 08:30:00, an absolute time of 00:30:00 is actually before the start of the simulation.
Event start	A specified offset from the start of a specified event (negative offset values can be used).
Simulation start	A specified offset from the start time of the simulation.
Simulation end	A specified offset from the end time of the simulation (use negative offset values to indicate times before the end of the simulation).

**Time Reference Ranges**

In some places, time ranges can be specified using time references (as above) instead of simple times. In these cases the following options are possible:

Description	Meaning
All simulation	The entire simulation
Between start and end	Starting from one specified time reference, continuing until a second specified time reference
Over duration	Starting from one specified time reference, continuing for a specified duration
Scheduled	Specified using a series of intervals. Starting from the beginning of an interval and continuing for the duration of the interval. Repeating for each interval. Interval start times are relative to the starting time reference.

### 4.2.11 Triggering Events

Events can be used to create agents, apply actions to agents, or modify the state of scene objects. An event that is engaged in any of these activities is said to be **active**. An event can become active multiple times during a simulation.

Events can become active at a specified time for a specified interval, or they can be set to become active in response to changing conditions in the scene. Once an event becomes active, it will remain active until done or turned off. An event cannot restart once active. An event can become active again immediately after ceasing to be active.

[Triggers](#) are the mechanism by which events can be turned on and off. A trigger will 'fire' when a certain condition is met. The most basic triggers are time triggers which fire at a specified time. A more complicated trigger is the area population trigger which will fire continuously whenever the population in an area is above or below a target threshold. Triggers can be combined together using logic to create complex combinations of conditions.

#### Simple Start Trigger

Some events like the [Journey](#) will become active whenever the starting trigger fires. The journey will then create agents over the specified duration. If the starting trigger fires again while the journey is active, that firing is ignored. Once the journey has completed creating all agents, it will cease to be active. The next time the starting trigger fires the journey will again become active and create the specified number of agents.

#### Trigger Range

Events like the [Gate Access](#) will specify an active range using a trigger range. The trigger range will specify a starting trigger and an end condition. The gate event will open/close the gates when the start trigger fires, leave the gates open/closed while active, then return the gates to their original state when the end condition is met. While active, the start trigger is ignored. Once the end condition has been met and the event is no longer active, the start trigger can again fire and cause the event to become active.

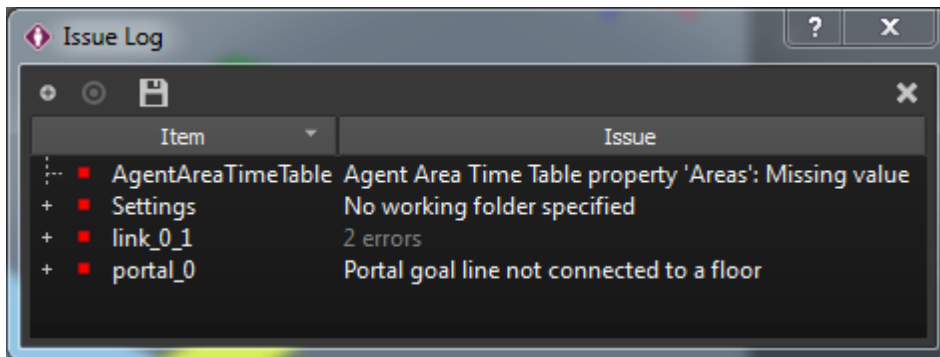
Trigger Range	
All	The event will become active at the start of the simulation and remain active until

Trigger Range	
simulation	the end of the simulation.
Between start and end	The event will become active when the start trigger fires. Once active, the event will remain active until the end trigger fires. Any firings by the start trigger while active, or by the end trigger while inactive are ignored.
Over duration	The event will become active when the start trigger fires and remain active for the specified interval.
Scheduled	The event will become active according to the table of intervals. The scheduled times are with respect to the firing of the start trigger. The event is considered active from the firing of the start trigger to the completion of the final interval. Any firing of the start trigger while the event is active is ignored.
Whenever	The event will be active any time the condition is satisfied, and will not be active when the condition is not satisfied.

### 4.2.12 Issue Window

The issue window lists the results of a [validation](#) or [audit](#). In the case of validation the window shows potential errors in any object or the project settings. The issue window can be accessed from the bottom right of the [main window](#), listing the results of the last validation.

A similar issue log panel is available at the end of a simulation, listing any issues which arose during the course of the simulation.



Issues are grouped by the objects they are related to, and individual issues for each object and further details can be seen by expanding listed items. All issues can be expanded at once using the "Toggle expansion" button.

When an issue or audit result references one or more objects, those objects can be highlighted in the [list](#) and [scene](#) views using the "Select subject" button. The contents of the issue window can be saved to an external CSV file.

### 4.2.13 Keyboard and Mouse Controls

Project Controls	
<b>Ctrl-A</b>	Select all from view. Select visible objects in the <a href="#">3D scene view</a> or the current

Project Controls	
	members of the <a href="#">list view</a> .
<b>Ctrl-Z</b>	Undo
<b>Ctrl-Y</b>	Redo
<b>Ctrl-S</b>	Save

Camera Manipulation Controls	
<b>Pan</b>	<p>Press and hold the 'S' key, then press and hold the left mouse button while moving the mouse.</p> <p>Alternative: Hold the middle mouse button while moving the mouse.</p>
<b>Rotate</b>	<p>Press and hold the 'S' key, then press and hold the right mouse button while moving the mouse. Note rotations are done around the current focus object. To change the focus object, select an object and press the 'F' key.</p> <p>Alternative: Hold the shift key and middle mouse button while moving the mouse.</p>
<b>Zoom</b>	<p>Press and hold the 'S' key, then press and hold the middle mouse button while moving the mouse.</p> <p>Alternative 1: Hold the control key and middle mouse button while moving the mouse.</p> <p>Alternative 2: Scroll the mouse wheel.</p>
<b>A - Show All</b>	Use the 'A' key to move the camera so that all visible objects and agents are framed within in the view.
<b>F - Focus</b>	Use the 'F' key to centre and focus the view on the currently selected object(s). The selection object(s) will become the new focus for any camera rotations.

<a href="#">Playback</a> Controls	
<b>Right Arrow</b>	Advance one frame in playback.
<b>Left Arrow</b>	Rewind one frame in playback.
<b>Up Arrow</b>	Play/pause simulation playback.
<b>Down Arrow</b>	Pause simulation playback.

Object Selection Controls	
<b>Left Click</b>	Select an object or agent in the scene. Click and drag to "box select" multiple

Object Selection Controls	
	objects. Dragging right will only select objects fully enclosed by the box, dragging left will select any object within or overlapped by the box.
<b>Ctrl Key + Left Click</b>	Hold this key while selecting objects to toggle (add/remove) objects in the current selection.
<b>Shift Key + Left Click</b>	Hold this key to add objects to the selection
<b>E</b>	While an object is selected, switch to edge selection mode. Only edges of selected objects can be selected.
<b>T</b>	While an object is selected, switch to vertex selection mode. Only vertices of selected objects can be selected.
<b>Y</b>	While an object is selected, switch to face selection mode. Only faces of selected objects can be selected.
<b>Space</b>	Return to object selection mode. Any <a href="#">manipulations</a> will also be stopped.
<b>Escape</b>	Cancel or clear the current operation. Repeatedly pressing escape will walk back through active modes until in object select mode with no objects selected.

Mode Controls	
<b>M</b>	Open the measure tool. See <a href="#">Model Panel</a> tools for more information.
<b>I</b>	Open the tracing tool. See <a href="#">Tracing New Objects</a> for more information.
<b>Q</b>	Enter or exit process chain edit mode for editing the connections between servers.
<b>D</b>	While a drawing layer is selected, switch to drawing mode. Only points and lines of the selected drawing layer can be selected.
<b>R</b>	While a drawing layer is selected, switch to drawing mode and fill all regions of the drawing layer. If already in drawing mode with drawing lines selected, only fill regions using the selected lines.

Transform Controls	
<b>X, C, V, B</b>	Start scaling, rotating, translation, or snap manipulations respectively. See <a href="#">Editing Geometry</a> or <a href="#">Transforming Objects</a> for more information.
<b>Z</b>	Start manipulator snap for moving the transform frame of reference. See <a href="#">Editing Geometry</a> or <a href="#">Changing the Reference Frame</a> for more information.

Tool Controls	
<b>Right Click</b>	Open a context specific menu which can be used to modify objects, view their properties, find related objects or create new objects in the <a href="#">scene view</a> . Object

Tool Controls	
	components have a separate context menu specific to the type of component.
<b>Ctrl-D</b>	Duplicate the currently selected object. The new object will be selected.
<b>G</b>	Starts a growing operation when a vertex, edge or face is selected. See <a href="#">Editing Geometry</a> or <a href="#">Growing Objects</a> for more information.
<b>N</b>	Generate a new object of a specific type from the current selection using the current <a href="#">generation options</a> .
<b>O</b>	In trace mode, close the traced polygon by adding a new line between the previous line and the starting point (see <a href="#">Tracing New Objects</a> ).
<b>K</b>	Open the slice (Knife) tool. See <a href="#">Editing Geometry</a> for more information.
<b>U</b>	Apply "Extend and Fuse" to selected lines in a <a href="#">Drawing Layer</a> . See <a href="#">Drawing Commands</a> for more information.
<b>Delete</b>	Delete selected objects or components.

#### 4.2.14 Application Preferences

MassMotion provides a number of preferences that apply to all projects. These preferences can be accessed under Preferences in the main window's Edit menu.

General	
<b>Scene View: Background</b>	Background colour to use when displaying scenes.
<b>Scene View: Near clip distance</b>	Minimum distance from the camera at which objects are visible. Decrease to avoid hiding near objects. Choosing a value that is too small could cause objects that are close together to appear as if they overlap one another.
<b>Scene View: Far clip distance</b>	Maximum distance from the camera at which objects are visible. Increase to avoid hiding far away objects in very large scene. Choosing a value that is too large could cause objects that are close together to appear as if they overlap one another.
<b>File Import/Export: List separator</b>	Specify the default separator character to use when importing and exporting csv files. This will affect <a href="#">Timetable</a> files, query data export, and <a href="#">agent position export</a> .
<b>Recent File List: Max count</b>	The maximum number of entries displayed in "Open recent" under the "File" menu.
<b>Auto Save: Enable</b>	When enabled, the current project will be saved to a temporary folder at regular intervals. See <a href="#">auto save</a> for more information.
<b>Auto Save: Interval</b>	Specify the interval between auto saves.

Editing	
<b>Undo: Stack size</b>	Set the maximum number of undo steps recorded. The larger the number the larger the memory footprint taken by the undo system. When modifying the geometry of large models each undo operation can hold a significant copy of the model in memory. Reducing the stack size can help reduce the amount of RAM required to edit a large project.
<b>Generate from: Reference geometry</b>	Specify the default action to take after generating objects from reference geometry.
<b>Generate from: Faces</b>	Specify the default action to take after generating objects from faces.
<b>Generate from: Edges</b>	Specify the default action to take after generating objects from edges.
<b>Generate from: Drawings</b>	Specify the default action to take after generating objects from drawing regions or lines.

Movie/Image	
<b>Appearance: Resolution</b>	Default resolution to use for movie/image export.
<b>Appearance: Background</b>	Default background colour to use for movie/image export.
<b>Movie: Quality</b>	Default quality setting to use for movie/image export.
<b>Movie: Frame rate</b>	Default frame rate to use for movie/image export.
<b>Overlay: Text</b>	Default overlay text colour to use for movie/image export.
<b>Overlay: Population count</b>	Whether population count should be displayed by default during movie/ image export.
<b>Overlay: Time</b>	Whether current time should be displayed by default during movie/image export.
<b>Overlay: Map legend</b>	Whether the map legend (if any) should be displayed by default during movie/image export.
<b>Overlay: Map title</b>	Whether the map title (if any) should be displayed by default during movie/ image export.
<b>Overlay: Reference axes</b>	Whether the set of reference axes should be displayed by default during movie/image export.



### 4.2.15 Fixing Face Normals

Each mesh face has a front side and a back side typically identified by a face normal vector. MassMotion does not make any distinction between front and back, but other 3D packages might. If exported geometry does not appear correctly in 3rd party software, it could be that the face normals need to be corrected.

To correct face normals, enable 'Darken back faces' from the Geometry section of the Visibility Options in the [3D Scene View](#). The normals can then be adjusted by right clicking on a selected object or right clicking on one or more selected faces.

From the Geometry submenu of the object context menu:

Object Context Menu	
<b>Automatically fix normals</b>	Attempt to guess at the correct orientation of face normals by assuming surface objects will have normals facing up, and volume based objects will have normals facing outwards. This option should provide a quick way to correct 95% of the normals in a legacy project.
<b>Align normals up</b>	Set all normals to be pointed up so that the top surfaces are considered the front.
<b>Align normals out from centroid</b>	Set all normals to be pointed out from the object centroid so that front faces are on the outside of the object.
<b>Flip normals</b>	Swap all front and back faces so that normals are opposite to their current direction.

From the Normals submenu of the face context menu:

Face Context Menu	
<b>Align up</b>	Set all normals to be pointed up.
<b>Align out</b>	Set all normals to be pointed out from the centroid of the selected faces.
<b>Flip</b>	Swap all front and back faces so that normals are opposite to their current direction.

## 4.3 Objects

Almost all data in a project is stored in the form of objects. Objects can be classified as:

- [Scene Objects](#): physical objects such as floors which have a concrete presence in the scene.
- [Activities](#): create agents, modify agents, or control operations in the scene.
- [Analysis](#): queries and related objects that analyze the results of a simulation.
- [Bookmarks](#): used to save and restore scene viewpoints, which objects are shown/hidden, and current playback time.

### Properties

Each object contains a list of properties. These properties can be modified using the [property](#)

[window](#) which is available through the object's right-click menu, or by double-clicking on the object in the scene or list views. Objects which share common properties can have those properties edited all at once using [object multi-edit](#). Some properties can refer to other objects (See [Choosing Objects](#)).

A property can be either valid or invalid based on its value. The validity of a property is displayed by the property indicator beside it. These indicators can be used to reset a property to its default value, or to copy values from one property to another (either by right-clicking or dragging the indicator). In some cases a property will be shown as valid but still fail validation. This occurs when the cause of the problem is only uncovered by checking against other objects or property values.

Common Object Properties	
<b>Selected</b>	All objects can be <a href="#">selected</a> .
<b>Hidden</b>	All scene objects can be hidden. Hidden objects are not drawn in the <a href="#">scene view</a> . They appear in the <a href="#">list view</a> , but with only an outline for the corresponding capsule. Hiding a collection will hide all of its member objects. Hiding a map will remove any textures that it has applied in the scene.
<b>Disabled</b>	Some objects can be disabled. A disabled object will not be included in simulations and will be shown as grey in the scene and displayed with a grey type capsule in the <a href="#">list view</a> .

### 4.3.1 Scene Objects

Scene objects have a physical presence in the scene and are visible in the [3D Scene View](#).

#### Imported Geometry

Scene Object Type	Description
<a href="#">Reference Geometry</a>	Imported geometry from external sources. Reference geometry will come in as either <a href="#">generic geometry</a> (3D mesh), a <a href="#">drawing layer</a> (2D drawings) or a special <a href="#">ifc geometry</a> for IFC files.

#### Basic Objects

Scene Object Type	Description
<a href="#">Floor</a>	Any flat area in which agents may walk.
<a href="#">Barrier</a>	Blocks agent passage.
<a href="#">Portal</a>	Represents entrances into the simulation as well as agent destinations.
<a href="#">Server</a>	Used to model queues and more complex agent behaviour.
<a href="#">Wait Space</a>	An flat area of a floor in which agents may be constrained to wait.

**Connection Objects**

Scene Object Type	Description
<a href="#">Link</a>	A <a href="#">connection object</a> which connects floors horizontally. May represent a doorway or turnstile.
<a href="#">Stair</a>	A vertical <a href="#">connection object</a> which represents a stair.
<a href="#">Escalator</a>	A vertical <a href="#">connection object</a> representing an escalator. Escalators must be unidirectional.
<a href="#">Ramp</a>	A vertical <a href="#">connection object</a> which represents a ramp.
<a href="#">Path</a>	A <a href="#">connection object</a> constructed from a curve segment. Does not need to be horizontal.
<a href="#">Elevator</a>	A vertical object connecting multiple floors via a moving cab.

**Counting Objects**

Scene Object Type	Description
<a href="#">Cordon</a>	Boundary planes in space that agents can cross.
<a href="#">Volume</a>	Areas in space that may contain agents.

**Decorative Objects**

Scene Object Type	Description
<a href="#">Visual</a>	Cosmetic objects that enhance the look of the scene but have no impact on the simulation.

**4.3.1.1 Reference Geometry**

Reference geometry objects are created by [importing](#) content from external files. Standard 3D elements are imported as [generic geometry](#), 2D elements are imported as [drawing layer](#) objects, and the contents of an IFC file are imported as [IFC geometry](#).

These objects have no impact on simulation but they can be used to create MassMotion objects for simulation (see [Generating Objects from Geometry](#) for more information).

Imported reference geometry is automatically placed inside a [transform collection](#) which can be used to scale, rotate, and translate the imported geometry.

**Reference Geometry Properties**

<b>Lock Geometry</b>	While enabled, the object's geometry cannot be directly manipulated or edited. It is still possible to delete a locked object or apply transformations using the associated <a href="#">Transform</a> object.
<b>Transform</b>	Upon import, all objects from the same file are grouped in a <a href="#">transform collection</a> which allows them to be translated, rotated and scaled together.

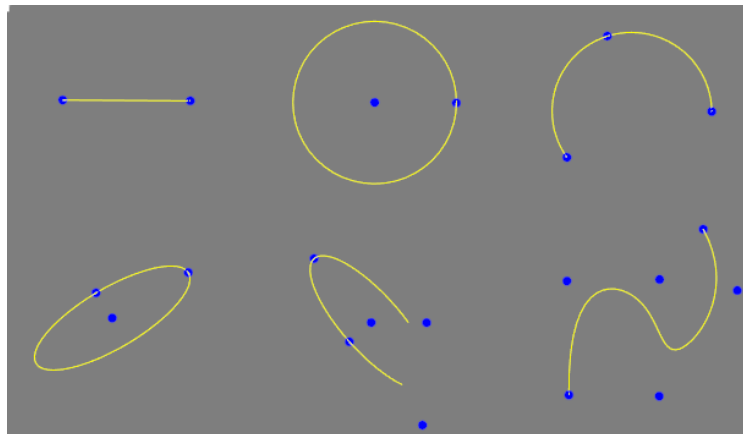
4.3.1.1.1 Drawing Layer

Drawing layers are a [reference geometry](#) which represents 2D elements on a common plane. Drawings consist of a series of points and lines. Lines can be connected together into line chains. Areas completely enclosed by lines can be filled to form regions. Lines and regions can then be used to quickly create other [scene objects](#) via the right click menu. This can be done in "Edit Drawing" mode by extruding the lines from the drawing plane, or in "Find Region" mode by extruding enclosed [regions](#).

It is possible to translate, rotate and scale the entire drawing in 3D space, however, editing drawings components must be done on the 2D plane. See: [working with drawings](#).

**Lines**

Drawings are comprised of points and lines. Lines are connected to and defined by a certain number of points. It is possible for a point to be connected to multiple lines or to the same line multiple times. To separate lines which share a common point, or a point that is connected to many lines, use the [explode command](#).



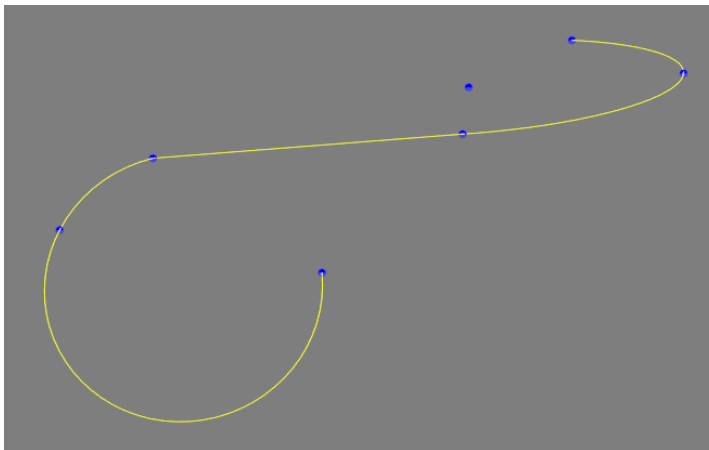
**Straight Line, Circle, Circle Arc**  
**Ellipse, Ellipse Arc, Spline**

Line Type	Description
<b>Straight Line</b>	A straight line between two points.
<b>Circle</b>	A circle defined by a centre point and a point on the circle.
<b>Circle Arc</b>	A circle arc defined by the arc end points and a third point on the arc.
<b>Ellipse</b>	An ellipse defined by a centre point and two points on the ellipse.  Given a centre O, and two points A and B, the ellipse is defined as any point where $P = (A-O) \cos(t) + (B-O) \sin(t)$ for any t.

	Note that A and B therefore do not define the semi-major and semi-minor axes.
<b>Ellipse Arc</b>	<p>An ellipse arc defined by a centre point, two points on the ellipse (but not necessarily on the arc), and two points defining the start and end angles of the arc.</p> <p>The points defining the ellipse define the ellipse arc similarly to the points of the full ellipse.</p> <p>It is possible for none of the points to be on the line they define.</p>
<b>Spline</b>	<p>A spline can be defined by any number of points, but generally at least 4. Splines cannot be created within MassMotion, only imported.</p> <p>It is possible for none of the points to be on the line they define.</p>

### Line Chains

Lines, circle arcs and some ellipse arcs that have common end points are considered a line chain and are treated specially for some [editing commands](#) (eg. collapse to straight line).



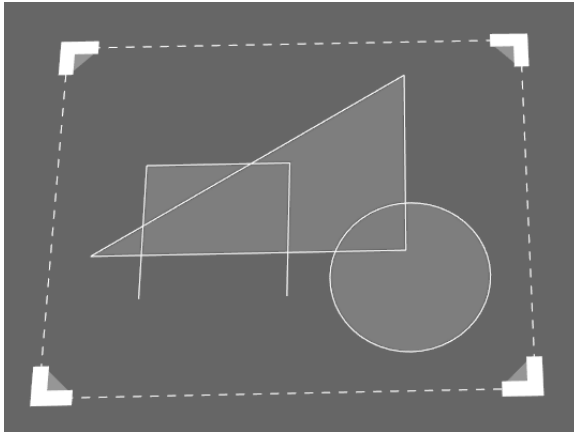
**A line chain comprised of a circle arc, a straight line and an ellipse arc.**

Ellipse arcs can only be part of a line chain if their start and end angle points are directly on the line itself. If working with ellipse arcs, it may be easier to discretize them to straight line segments.

A single line (straight line, circle arc, or valid ellipse arc) is considered to be a line chain as well. While collapsing a single straight line to a line will do nothing, collapsing a circle arc to a line will replace it with a straight line connecting its end points.

### Regions

[Drawing Regions](#) are a special type of component generated by filling areas that are completely enclosed by lines. Regions can be accessed via region mode in the [tool panel](#) or via the shortcut key 'R'. Regions can be used to [generate scene objects](#).



Six regions enclosed by various shapes

#### 4.3.1.1.2 Generic Geometry

Generic geometry is mesh based [reference geometry](#) that has no special type or characteristic. It cannot be create as a new object but is usually produced by [importing geometry](#) from an external file. It can also be generated by converting existing objects into reference geometry through the right-click 'Convert' context menu.

Generic geometry can be used to create scene objects for use in simulation and analysis. See [generating objects from geometry](#) for more information.

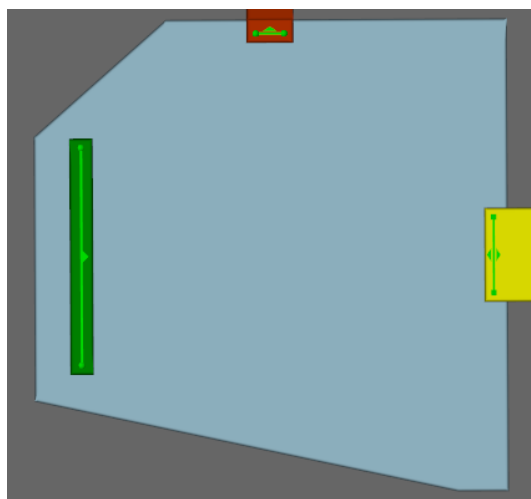
#### 4.3.1.1.3 IFC Geometry

IFC geometry objects preserve the IFC type information taken from the source IFC file. This type information can be used to automatically convert IFC geometry into a suitable MassMotion [scene object](#). See [generating from IFC](#) for more information.

### 4.3.1.2 Floor

Floor objects are contiguous polygon mesh surfaces that define the extent of a "walkable" area. An agent's awareness of other agents, barriers, and route options is based on the floor on which they are standing. Most other scene objects must be connected to floor objects (portals, links, stairs, ramps, escalators).

The default new floor is a 10m x 10m flat square.



A floor

**Restrictions on Geometry**

- Floors may have thickness, although only the top surface will be considered during simulation.
- The floor surface must not overlap itself (the entire top surface of the floor must be visible from above).
- The floor surface should be continuous (a single floor should not have two or more disconnected parts).
- The floor may have holes.
- Floors must have dimensions of at least 2 agent widths by 2 agent widths (by default 1m x 1m).
- There is no maximum size for floors, however users should consider subdividing large (>10,000m<sup>2</sup>) floors into logical areas.
- The floor surface can contain elevation changes but those changes are not considered for [Agent Movement](#), speed calculations, or [Agent Navigation](#).

If connected objects are completely separated by [barriers](#) on the floor, it is more appropriate to separate the floor into two separate floors. Agents attempting to traverse a floor between two links completely separated by a barrier will be removed from the simulation with an error.

**Impact on Agent Speed**

By default, agent speed is unaffected by floor traversal. Speed can be capped at a specific value by enabling the 'Limit Speed' property. See [Agent Profile](#) for information on agent speed.

**Properties**

General Tab	
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of <a href="#">surface maps</a> . Smaller values describe walkable space and barrier edges more accurately and can be necessary in confined spaces. Smaller values also increase memory and processing requirements during simulation initialization and execution.

**Agent Behaviour Tab**

<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the floor.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the surface, avoiding obstacles and other agents.</p> <p><b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors.</p>
<b>Ignore barriers</b>	Agents will walk through obstacles when traversing the surface. Obstacles are not included in the surface map.
<b>Ignore neighbours</b>	Agents do not see other agents and will walk through them, making no attempt to avoid neighbors. Agents will take the shortest path to their goal. Density is calculated as if each agent was in isolation.
<b>Body radius</b>	If enabled the body radius for all agents on the surface will be changed to the specified value (m). This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the floor will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.
<b>Direction Bias</b>	If enabled, the direction bias of agents on the surface will be changed to the specified value. This can be useful for corners or narrow spaces where the natural system wide direction bias is locally inappropriate. See <a href="#">agent profile</a> for more information on Direction Bias.
<b>Speed Limit</b>	Enables a maximum threshold for agent speed (m/s). Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed density</b>	If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a> .

Actions Tab	
<b>On enter</b>	The action will be applied to all agents as they enter the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>On exit</b>	The action will be applied to all agents as they leave the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

Collections Tab	
<b>Zones</b>	Zones which the floor is a member of, if any. For more information, see <a href="#">Zones</a> .



### 4.3.1.3 Barrier

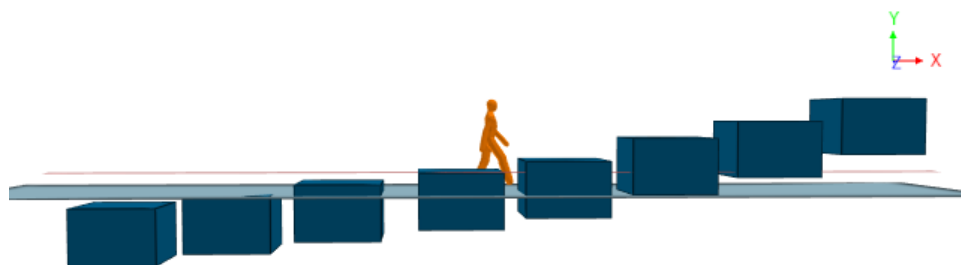
A barrier describes a region through which agents cannot pass. Barriers are typically used to represent columns, walls, furniture, or anything else that can impede agent movement. If a barrier will not impact agent movement (escalator handrails, trees beside a walking path, etc.) then it should be converted to a [visual](#) object so as not to impact simulation performance.

The default new barrier is a 1m x 1m x 1m cube which can be then edited to any size or shape.

#### Barrier geometry interaction with walkable geometry

The blocked region on a floor is determined using the volume of intersection between the floor and the barrier. Those portions of the barrier that do not intersect with the floor are ignored. Those portions of the barrier that are more than 40cm above the floor are ignored.

Intersection tests are performed using the bounding box of the floor which is formed using the highest and lowest points on the floor (including the underside). If barrier walls or columns on a floor intersect with the underside of the floor above, they will impact agent movement on that floor. To avoid this, ensure that barriers for a lower floor terminate just below the upper floor.



1m tall barriers arranged at various heights relative to a floor. The red line marks the 0.4m cutoff. Measuring distance from floor to barrier top is -0.2, 0.0, 0.2, 0.4, 0.6, 1.0, 1.3, 1.7.



The resulting obstacle map shows the first and last barriers are below and above the cutoff. The 5th barrier has top and bottom beyond the cutoff so only the walls are included.

#### Notes on barrier geometry

- Barrier objects can be 3D volumes or 2D planes.
- Barriers can have a large impact on the time it takes to compile a simulation. Barriers that contain a large number of faces can take a long time to project onto the floor obstacle maps. Similarly, if an obstacle contains faces over a large area, then many floors will have to consider the barrier when generating their maps.

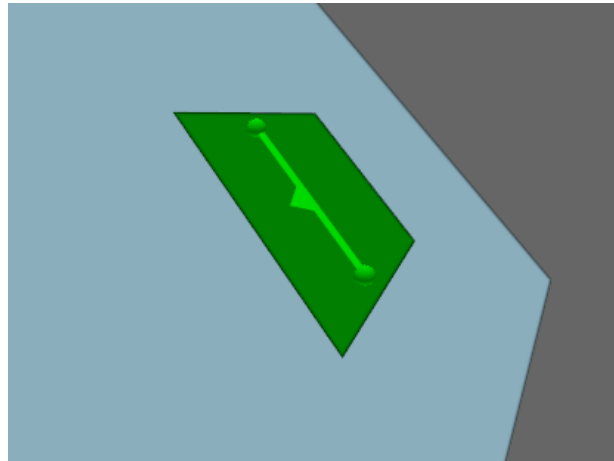
- Barriers must not completely block off links or portals on the same floor. Doing so may result in a large number of agents being removed from the simulation. If dividing the floor into unconnected areas is necessary, use separate floors for each area instead.

**Properties**

There are no properties for objects of this type.

**4.3.1.4 Portal**

Portals act as gateways in the scene. [Events](#) use portals to indicate where to place newly created agents. Events and [actions](#) use portals to specify target destinations.



A portal

**Restrictions on Geometry**

- Portals must be flat.
- Portals must be approximately rectangular or trapezoidal.

Portals must be positioned between 0.01m and 0.2m above a floor and no closer than 0.2m to a floor edge. A portal can only connect to one floor.

**Properties**

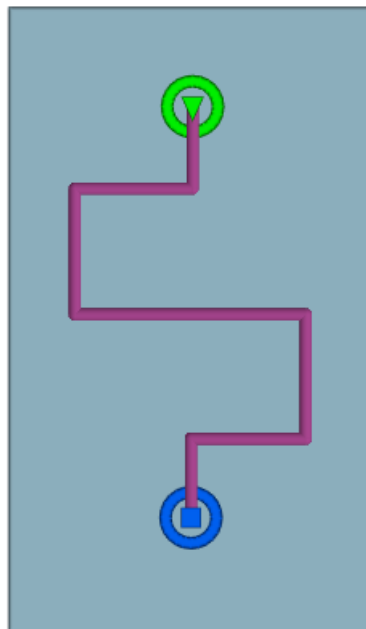
General Tab	
<b>Network: Type</b>	<p>The role the portal will play in the network.</p> <p><b>Entrance and Destination:</b> Agents may enter and exit the simulation at this portal as well as use it as a waypoint.</p> <p><b>Entrance Only:</b> Agents may enter the simulation at this portal, but may not use it as a destination or waypoint.</p>
<b>Agent placement: Distribute</b>	<p>Defines how newly created agents are placed in the simulation environment. Agents will only appear in valid unobstructed regions of the portal or floor, but may appear on top of existing agents in areas of high density.</p> <p><b>Along Spawn Line:</b> Agents appear randomly distributed along the portal goal line.</p> <p><b>Inside Portal:</b> Agents appear randomly distributed inside the planar rectangular boundary of the portal geometry.</p> <p><b>On Floor:</b> Agents appear randomly distributed inside the planar, rectangular</p>

	boundary of the floor beneath the portal.
<b>Agent placement: Start angle</b>	Measured in degrees. Defines the orientation of the agents when they enter the simulation environment. This angle is visually represented by a triangle decoration on the portal's spawn line.

Actions Tab	
<b>Entered simulation</b>	The action will be applied to all agents as they enter the simulation through the portal. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Arrived at waypoint</b>	The action will be applied to agents that are seeking the portal as they arrive at the portal. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

#### 4.3.1.5 Server

A server is used for process modeling and is the basic building block of process chains. Each server has a conceptual entry point (green triangle) through which agents enter, and an exit point (blue square) through which agents leave the server. If the server has a process time other than zero, agents will be processed and held at the exit point for the specified duration. Server ports can be grouped together or connected to other ports to form process chains. For more information on process modeling and process chains see the introduction to [process chains](#).



Server object

To create a server use the "Server" button on the Scene tab of the main application window or right click in a 3D view and select Create-> Scene-> Server. The default new server is a 2m horizontal line segment. This line segment can be extended by growing the end vertices, or splitting the edge into multiple pieces. See [editing geometry](#) for more information.

**How Agents Use Servers**

Agents can be directed to one or more servers through the 'Seek process chain' task given as an [agent action](#). When directed to a process chain agents will seek out the start of the chain. Once an agent has found the start it will be dispatched to one of the entry servers in the chain. The agent then moves through the chain passing from server to server as capacity becomes available. Once the agent has reached the end of the process chain the 'Seek process chain' task is complete. See [server operations](#) for a description of how an agent will approach and be processed by a server.

**Access**

Whether or not a server is available to an agent depends on a number of factors including the server entry state, available server capacity, and whether or not the server is set to require or prefer particular types of agents. See [controlling agent access](#) for more information.

**Notes on server geometry**

- A server must contain a single curve object (even when the approach type is set to 'Ignore Line').
- All points on the curve must be just *above* the same floor object.
- Server ports can only be grouped with or connected to other servers that are on the same floor.

**Impact on Agent Speed**

By default, agent speed is unaffected by server traversal. Agents will slow down to follow an agent that is immediately in front. In highly congested areas, boarding rates onto the server may be artificially limited by speed density constraints. Setting the speed density to 'Unconstrained' can remove these constraints and result in higher boarding rates.

**Properties**

General Tab	
<b>Approach</b>	<p>Determine how agents will move towards the server target (see below).</p> <p><b>Standard walk to target:</b> Use the traversal type of the floor to approach the server.</p> <p><b>Teleport to target:</b> Instantly teleport to the server.</p> <p><b>Virtual:</b> Join the server from the agent's current position. The agent will queue at the server and be processed by the server all without moving. The agent will be free to move again once released by the server.</p>
<b>Target</b>	<p>Determine which part of the server the agent should approach.</p> <p><b>Server entry:</b> Agents will walk or teleport to the start of the server line.</p> <p><b>End of queue on server:</b> Agents will walk or teleport to the end of the queue on the server. If the server is full agents will walk or teleport to the server entry. If there is no queue agents will walk or teleport to the server exit.</p> <p><b>Server exit:</b> Agents will walk or teleport to the end of the server line.</p>
<b>Processors</b>	<p>Specify the number of agents that can be processed at the same time.</p> <p>This is only available when 'Approach' is set to 'Virtual' or 'Target' is set to 'Server exit' as in all other cases only one agent can physically access the processor at a time.</p>

<b>Capacity</b>	<p>Limit the number of agents that can be using the server at one time. When enabled, dispatches will only send agents to the server when there is available capacity.</p> <p>Available capacity is calculated as:</p> $\text{Available Capacity} = \text{Max Capacity} - \text{Count Approaching} - \text{Count Queuing} - \text{Count Processing}$ <p>Max capacity can be:</p> <p><b>Specified:</b> A fixed capacity count.</p> <p><b>Calculated from line length:</b> The number of agents that will fit on the server line as calculated from line length, agent radius (specified), and average queue spacing.</p>
<b>Contact Time</b>	<p>If enabled, agents will be held in the processing state for the given interval.</p> <p><a href="#">Tests</a> provide a flexible way for varying the processing time. Agents can be given different contact times depending on the time of day, population on the servers, or token possession. Tests are evaluated from top to bottom and an agent will use the distribution corresponding to the first test that evaluates to true. If no tests evaluate to true the fallback distribution is used.</p>

<b>Access</b>	
<b>Entrance default</b>	<p>The server entrance can be opened or closed using a <a href="#">server access</a> event.</p> <p><b>Closed:</b> The server entrance will start closed preventing agents from using the server.</p> <p><b>Open:</b> The server entrance will start open.</p>
<b>Require agents satisfy</b>	<p>When enabled the server will only be available to agents for which the test evaluates to true. Agents for which the test does not evaluate to true will see the entrance as closed.</p>
<b>Prefer agents satisfy</b>	<p>When enabled the server will accept all agents, but given a choice will always choose an agent for which the test evaluates to true.</p>
<b>Exit default</b>	<p>The server exit can be opened or closed using a <a href="#">server access</a> event.</p> <p><b>Closed:</b> The server exit will start closed, holding agents that have been processed until the server exit is opened.</p> <p><b>Open:</b> The server exit will start open, releasing agents after processing when there is downstream capacity.</p>

<b>Agent Behaviour Tab</b>	
<b>Queue Spacing</b>	<p>This determines how closely each agent will follow a previous agent. This can be set as a distribution in metres, see <a href="#">single value distributions</a> for more information.</p>

<b>Speed Density</b>	If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a> .
----------------------	---

<b>Actions Tab</b>	
<b>Done Processing</b>	The specified action is applied to each agent immediately after the agent has been processed but before it leaves the server. If a task is given during this action the agent will abandon the current "Seek process chain" task before it is considered complete. The action is applied before the agent is considered by any server exit event test or downstream server's preferred or required test.
<b>On Release</b>	The specified action is applied to each agent as it leaves the server. If the server is the end of a process chain, the action is applied immediately after the agent has completed the current "Seek process chain" task.

<b>Connections Tab</b>	
<b>Source</b>	If the server is connected to an upstream <a href="#">dispatch</a> object it will be listed here. A source dispatch can be removed, replaced or added here. Servers can only be connected to one source dispatch at a time.
<b>Sink</b>	If the server is connected to a downstream <a href="#">dispatch</a> object it will be listed here. A sink dispatch can be removed, replaced or added here. Servers can only be connected to one sink dispatch at a time.

4.3.1.5.1 Server Operations

An agent can be directed to a process chain through a task given by the 'Seek process chain' [action](#). The agent will seek the floor that contains the process chain, choosing the lowest cost route through the network. Once on the floor the agent will immediately register with the process chain's initial dispatch. This dispatch will send the agent to one of the entry servers. The agent will approach the server, follow the server, be processed by the server, then register with the next dispatch in the chain. The task is completed when the agent has been processed by an end server in the process chain.

If the agent is interrupted and given a new task before reaching the process chain end it will restart at the beginning of the process chain when returning to the process chain task.

**Dispatch**

A [dispatch](#) will assign the agent to a downstream server. In the case where there is only one server the assignment is trivial. When there are multiple servers the method of assignment can be specified through a property of the dispatch object. Once an agent is assigned to a server it registers with that server and is included in the counts for that server.

**Approach**

Agents will only approach a server that has available capacity as defined by the input buffer capacity. If an agent has been dispatched to a server and there is no available capacity, the agent

will stand in place until space becomes available. Once there is capacity, the agent is registered with the input buffer (regardless of whether or not it has reached the server line) and approaches the processor according to the approach type (see the "General" properties table in [server](#)). Once the agent is within close range of the processor it becomes available for processing. If there is no available processor capacity, agents that are ready for processing will stand in place until there is capacity.

### Processing

When the agent begins processing it is removed from the input buffer but is still included in server capacity calculations. The processor may process multiple agents at one time according to the processor capacity. The amount of time spent processing each agent is based on the server's contact time. There is a common contact time distribution that is the default for all agents. Additional distributions can be specified on a per test basis. See the "General" properties table in [server](#) for more information on contact time.

### After Processing

Once processed, an agent is ready to leave the server. What happens next depends on the situation:

<i>Server exit is closed (see <a href="#">server access</a> event)</i>	The agent is held in place until the server exit is opened.
<i>Server is in the middle of a process chain and there is no downstream capacity</i>	The agent is held in place until there is available downstream capacity.
<i>Server is in the middle of a process chain and there is downstream capacity</i>	The agent registers with the next dispatch and awaits assignment to one of the next servers in the chain.
<i>Server is the end of a process chain</i>	The agent finishes the 'Seek Process' task and continues with its next task.

#### 4.3.1.5.2 Controlling Agent Access

##### Opening/Closing the Entrance

Entry to a server can be opened or closed through the use of [server access](#) events. A server with a closed entry is not visible to agents and will be ignored during [dispatch](#). Once an agent has been dispatched to a server it will proceed to that server even if the entry is closed while the agent is in transit.

##### Restricted or Preferential Access

A server can be configured to give exclusive and/or preferential access to specific agents based on the results of one or more tests. Servers that require a specific test will only accept agents for which the test evaluates to true. Servers with a preferred test will accept any agent but when given a choice will always choose an agent for which the test evaluates to true.

If an agent is sent to a group of servers that all require a test that does not evaluate to true for that agent, that agent will be removed from the simulation and the error logged.

If two servers are grouped and only one is set with either a preferred or required test, all agents that

pass the test will be sent to that server. Only once the preferred or required server is at capacity will agents that pass the test be sent to the other server that does not have any restrictions.

*Example:* Use of required and preferred is typical of an airport check-in system, where the economy class and business class status is determined by possession of an economy or business token. The economy and business accumulator queues would require their respective tokens, while the check-in desks would prefer the same tokens. While the business queue would be restricted to agents holding the business token, the business check-in desk would be willing to accept economy token holders in the event that the business queue was empty.

#### 4.3.1.6 Wait Space

Wait spaces define constrained areas on a floor in which agents can wait.

Agents are directed to use a wait space through the [wait style](#) property of whatever it is they are waiting for ([elevators](#), gated [links](#), [stairs](#), [ramps](#), [escalators](#), [paths](#), events, or [actions](#)).

##### Geometry

A wait space can only span a single floor, and must be entirely on that floor. Each wait space is connected to its floor by a multi-segmented goal line. All points of the goal line must be over the wait space. Goal lines can be edited like any other geometry component in vertex or edge selection mode. Additional points can be added using [split](#) or [grow](#). Goal lines act both as the initial target guiding agents to the wait space and as a potential target for the cluster wait style.

##### Choosing a Wait Space

Depending on the wait style, agents will either be assigned a wait space or will chose the lowest cost wait space. In either case, the agent will only consider wait spaces that are on the same floor as the agent. If none of the wait spaces are on the current floor, the agent will use the specified fallback wait style. When choosing a wait space, agents will consider how long it would take to walk to the wait space and the relative congestion on that wait space.

##### Using a Wait Space

Agents will move towards the goal line of the wait space and begin "waiting" as soon as they enter the wait space area. Waiting agents will move or stand still according to the wait style setting of the wait space.

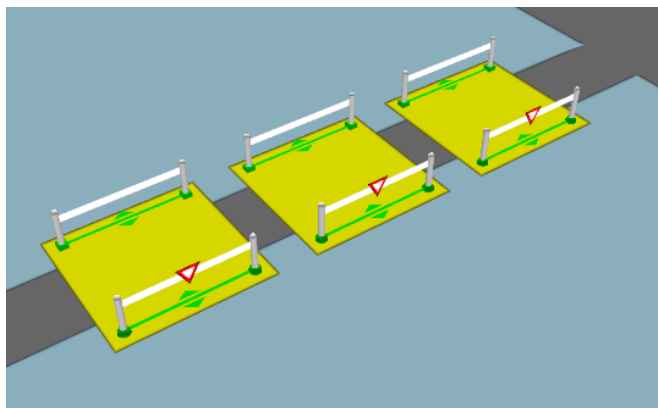
General Tab	
<b>Costs: Distance added</b>	Modifies the perceived distance the agent will have to travel to reach the wait space.
<b>Costs: Crowding multiplier</b>	Modifies the penalty associated with crowding or high density on the wait space.
<b>Wait Style</b>	Sets the behaviour of agents waiting on the wait space. While waiting, agent movement will be constrained by the wait space boundaries. See <a href="#">wait style</a> for a description of the different waiting behaviours.

#### 4.3.1.7 Link

Links are the most basic type of [connection object](#). They act as bridges or doors through which agents may pass from one floor to the other. The default new link is a 1m x 2m flat rectangle.

Links will ignore changes in elevation and do not add any vertical route costs.





A series of gated links with priority flow enabled.

### Restrictions on Geometry

- The link should be very close to flat
- The link should be approximately quadrilateral (a rectangle, trapezoid or parallelogram) although they may consist of a large number of individual triangles.
- Links must be at least 0.4m wide to allow proper generation of goal lines.

The link must be placed such that each goal line is between 0.01m and 0.20m above a floor, at least 0.20m from the edge of the floor. The two connecting floors must be different (a link cannot be used to connect two portions of the same floor).

### Impact on Agent Speed

By default, agent speed is unaffected by link traversal. Speed can be capped at a specific value through enabling the 'Limit Speed' property. See [agent profile](#) for information on agent speed.

### Properties

General Tab	
<b>Direction</b>	Sets the direction in which agents can traverse the link.  <b>Two way:</b> Agents may cross in both directions <b>Unidirectional:</b> Agents may only cross in the specified direction.
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A number greater than 1 will increase the perceived time cost of queuing, making agents more likely to seek alternate options when there is queuing. A number less than 1 will decrease perceived queuing costs making agents more likely to wait in a queue. See <a href="#">agent navigation</a> for more information.
<b>Costs: Opposing flow multiplier</b>	Adjusts the penalty for links with agents moving across the object in the opposite direction. Numbers greater than 1 amplify the perceived cost from significant counter flows. Numbers less than 1 reduce the perceived cost. See <a href="#">agent navigation</a> for more information.
<b>Physical: Map</b>	Determines the resolution of <a href="#">surface maps</a> . Smaller values describe

<b>resolution</b>	walkable space and barrier edges more accurately and can be necessary in confined spaces. Smaller values also increase memory and processing requirements during simulation initialization and execution.
-------------------	---

<b>Access Tab</b>	
<b>Gate: Enable use as gate</b>	Configures the object as a gate allowing it to be opened or closed through the <a href="#">open gate event</a> . Agents cannot pass through a closed gate and will apply a 'cost of waiting' penalty when a route is closed.
<b>Gate: Default state</b>	Determine whether the starting/resting state is open or closed.
<b>Gate: Wait style</b>	Sets the waiting behaviour of agents while the gate is closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Gate: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Gate: Commit to wait</b>	Sets whether agents will commit to the gate once it has been chosen. When checked, agents will wait for this object regardless of changes in route costs elsewhere on the floor.
<b>Limit Flow</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity. The maximum rate can be specified as an absolute rate or a rate based on width.
<b>Priority: Enable priority access</b>	Enables a limited yield system where agents moving in one direction receive preferential access. This is useful for constrained geometry where the majority of the flow is in one direction, or for cases such as train doors where alighting passengers often have priority over those boarding.
<b>Priority: Primary direction</b>	<p>Sets the direction in which agents will have primary access. The counter-flow direction will yield.</p> <p><b>One way:</b> Agents moving in the counter-flow direction will not enter the object when there are agents moving in the priority direction with priority access. If <b>Primary will yield</b> is set, agents traveling in the primary direction will yield if they arrive at the object while it is already being used by agents in the counter-flow direction. This is useful for scenarios such as ladders where usage should be restricted to one direction at a time.</p> <p><b>Two way:</b> Agents in either direction can claim priority access. Priority access is maintained for a given direction until there are no more agents crossing in that direction.</p>
<b>Priority: Capture range</b>	Sets the distance in metres from which agents approaching the object can capture priority access.
<b>Priority: Move aside</b>	When enabled, agents in the counter-flow direction will move aside to accommodate the priority flow.

<b>Priority: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Priority: Commit to wait</b>	When enabled, agents that have chosen the object will continue to wait until they have access regardless of changes to costs in other routes on the floor.

<b>Agent Behaviour tab</b>	
<b>Queue: Apply queuing force on approach</b>	When enabled, a 'queuing' force is applied to agents as they approach the link. This force nudges agents towards the area in front of the link which tends to result in trapezoid shaped queues. When disabled, queues can be more of a semicircle.
<b>Delay on enter</b>	<p>Sets how long an agent will be delayed before stepping onto a link. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.</p> <p>If a link is gated and closes before the delay finishes, the agent will wait until the next time the link opens before attempting to step on again.</p>
<b>Delay on exit</b>	Sets how long an agent will be delayed before stepping off a link. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the link.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the surface, avoiding obstacles and other agents.</p> <p><b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors.</p>
<b>Ignore Barriers</b>	Agents will walk through obstacles when traversing the surface. Obstacles are not included in the surface map. This is commonly used for connection objects like stairs, escalators, or ramps.
<b>Ignore neighbours</b>	Agents do not see other agents and will walk through them, making no attempt to avoid neighbors. Agents will take the shortest path to their goal. Density is calculated as if each agent was in isolation.
<b>Body radius</b>	If enabled the body radius for all agents on the surface will be changed to the specified value (m). This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the link will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.
<b>Direction bias</b>	If enabled, the direction bias of agents on the surface will be changed to the specified value. This can be useful for corners or narrow spaces where the natural system wide direction bias is locally inappropriate. See <a href="#">agent profile</a> for more information on direction bias.

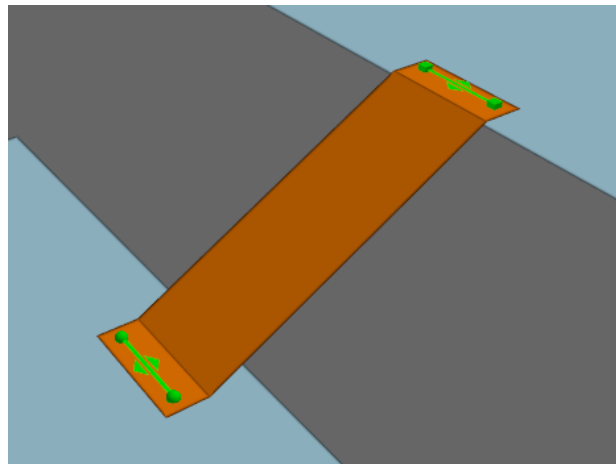
<b>Speed limit</b>	Enables a maximum threshold for agent speed (m/s). Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed density</b>	If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a> .

Actions Tab	
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

Collections Tab	
<b>Bank</b>	The bank which the link is a member of, if any. For more information, see <a href="#">route banks</a> .
<b>Perimeters</b>	Perimeters which the link is a member of, if any. For more information, see <a href="#">perimeters</a> .

**4.3.1.8 Stair**

Stairs are [connection objects](#) used to connect two floors that are at different elevations. This change in elevation results in an additional vertical cost during agent route selection and can have an impact on agent speed during traversal. Stairs are more costly to traverse than escalators and ramps.



A stair

### Restrictions on Geometry

- The stair object itself should not include any handrails or other geometry that extends above the walkable surface. If these are desired, they should be added as separate [barrier](#) or [visual](#) objects.
- The stair must have a flat landing area at each end; this should be at least 0.5m by 0.5m to allow construction of valid goal lines. Landings should be approximately rectangular or trapezoidal.
- The geometry (including the landings) may have thickness, but only the top surface will be considered during the simulation.

In order to run a simulation, the stair landings must be placed such that each goal line is between 0.01m and 0.20m above a floor, at least 0.20m from the edge of the floor. The two floors must be different (a stair cannot be used to connect two portions of the same floor).

### Impact on Agent Speed

By default, agent speed is modified as a function of the stair angle and direction of travel (see table below). Speed can also be capped at a specific value through enabling the 'Limit Speed' properties. See [agent profile](#) for information on agent speed.

Direction of Travel	Angle X (degrees)	Percentage of Natural Speed
Up	$0 < X < 27$	42.6
Up	$27 \leq X \leq 32$	Interpolated between 42.6 and 37.8
Up	$32 < X$	37.8
Down	$0 < X < 27$	57.4
Down	$27 \leq X \leq 32$	Interpolated between 57.4 and 49.8
Down	$32 < X$	49.8

### Properties

General Tab	
<b>Direction</b>	Sets the direction in which agents can traverse the stair.

	<p><b>Two way:</b> Agents may cross in both directions</p> <p><b>Unidirectional:</b> Agents may only cross in the specified direction.</p>
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A number greater than 1 will increase the perceived time cost of queuing, making agents more likely to seek alternate options when there is queuing. A number less than 1 will decrease perceived queuing costs making agents more likely to wait in a queue. See <a href="#">agent navigation</a> for more information.
<b>Costs: Opposing flow multiplier</b>	Adjusts the penalty for links with agents moving across the object in the opposite direction. Numbers greater than 1 amplify the perceived cost from significant counter flows. Numbers less than 1 reduce the perceived cost. See <a href="#">agent navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of <a href="#">surface maps</a> . Smaller values describe walkable space and barrier edges more accurately and can be necessary in confined spaces. Smaller values also increase memory and processing requirements during simulation initialization and execution.
<b>Physical: Rise angle</b>	The angle of incline measured from the horizontal. If automatically generated, the angle is measured using the horizontal run and vertical rise between goal lines. The magnitude of the rise angle can have an impact on agent speed (see table above).

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate allowing it to be opened or closed through the <a href="#">open gate event</a> . Agents cannot pass through a closed gate and will apply a 'cost of waiting' penalty when a route is closed.
<b>Gate: Default state</b>	Determine whether the starting/resting state is open or closed.
<b>Gate: Wait style</b>	Sets the waiting behaviour of agents while the gate is closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Gate: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Gate: Commit to wait</b>	Sets whether agents will commit to the gate once it has been chosen. When checked agents will wait for this object regardless of changes in route costs elsewhere on the floor.
<b>Limit Flow</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity. The maximum rate can be specified as an absolute rate or a rate based on width.

<b>Priority: Enable priority access</b>	Enables a limited yield system where agents moving in one direction receive preferential access. This is useful for constrained geometry where the majority of the flow is in one direction, or for cases such as train doors where alighting passengers often have priority over those boarding.
<b>Priority: Primary direction</b>	<p>Sets the direction in which agents will have primary access. The counter-flow direction will yield.</p> <p><b>One way (Ball to Box or Box to Ball):</b> Agents moving in the counter-flow direction will not enter the object when there are agents moving in the priority direction with priority access. If <b>Primary will yield</b> is set, agents traveling in the primary direction will yield if they arrive at the object while it is already being used by agents in the counter-flow direction. This is useful for scenarios such as ladders where usage should be restricted to one direction at a time.</p> <p><b>Two way:</b> Agents in either direction can claim priority access. Priority access is maintained for a given direction until there are no more agents crossing in that direction.</p>
<b>Priority: Capture range</b>	Sets the distance in metres from which agents approaching the object can capture priority access.
<b>Priority: Move aside</b>	When enabled, agents in the counter-flow direction will move aside to accommodate the priority flow.
<b>Priority: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Priority: Commit to wait</b>	When enabled, agents that have chosen the object will continue to wait until they have access regardless of changes to costs in other routes on the floor.

<b>Agent Behaviour tab</b>	
<b>Queue: Apply queuing force on approach</b>	When enabled, a 'queuing' force is applied to agents as they approach the stair. This force nudges agents towards the area in front of the stair which tends to result in trapezoid shaped queues. When disabled, queues can be more of a semicircle.
<b>Delay on enter</b>	<p>Sets how long an agent will be delayed before stepping onto an stair. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.</p> <p>If a stair is gated and closes before the delay finishes, the agent will wait until the next time the stair opens before attempting to step on again.</p>
<b>Delay on exit</b>	Sets how long an agent will be delayed before stepping off a stair. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the stair.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the</p>

	<p>surface, avoiding obstacles and other agents.  <b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors.</p>
<b>Ignore barriers</b>	<p>Agents will walk through obstacles when traversing the surface. Obstacles are not included in the surface map. This is commonly used for connection objects like stairs, escalators, or ramps.</p>
<b>Ignore neighbours</b>	<p>Agents do not see other agents and will walk through them, making no attempt to avoid neighbors. Agents will take the shortest path to their goal. Density is calculated as if each agent was in isolation.</p>
<b>Body radius</b>	<p>If enabled the body radius for all agents on the surface will be changed to the specified value (m). This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the stair will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.</p>
<b>Direction bias</b>	<p>If enabled, the direction bias of agents on the surface will be changed to the specified value. This can be useful for corners or narrow spaces where the natural system wide direction bias is locally inappropriate. See <a href="#">agent profile</a> for more information on Direction Bias.</p>
<b>Speed limit</b>	<p>Enables a maximum threshold for agent speed (m/s). Agents traveling above the specified maximum will have their speed reduced to the maximum.</p>
<b>Speed density</b>	<p>If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a>.</p>

Actions Tab	
<b>Ball: On enter</b>	<p>The action will be applied to all agents as they enter the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>
<b>Ball: On exit</b>	<p>The action will be applied to all agents as they leave the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>
<b>Ball: On enter</b>	<p>The action will be applied to all agents as they enter the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>
<b>Ball: On exit</b>	<p>The action will be applied to all agents as they leave the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>

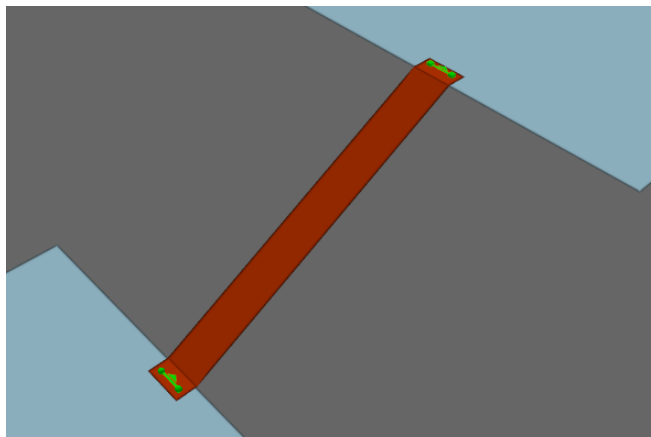


Collections Tab	
<b>Bank</b>	The bank which the stair is a member of, if any. For more information, see <a href="#">route banks</a> .
<b>Perimeters</b>	Perimeters which the stair is a member of, if any. For more information, see <a href="#">perimeters</a> .

#### 4.3.1.9 Escalator

Escalators are a vertical [connection object](#). Unlike other connection objects, escalators must be unidirectional. They are most typically used to model real world escalators, but can be made fully horizontal to simulate moving walkways. The default new escalator is 4m tall, 7m long, and 1.15m wide, with 0.5m landings at each end.

A change in elevation results in an additional vertical cost during agent route selection, however, escalators are less costly to traverse than [stairs](#) or [ramps](#).



An upwards escalator.

#### Restrictions on Geometry

- The escalator object itself should not include any handrails or other geometry that extends above the walkable surface. If these are desired, they should be added as separate [barrier](#) or, wherever possible, [visual](#) objects.
- The width of the escalator should include the distance between the handrails (for an escalator with 1m wide steps, this defaults to 1.15m).
- The escalator must have a flat landing area at each end; this should be at least 0.5m by 0.5m to allow construction of valid goal lines. Landings should be flat and approximately rectangular or trapezoidal.
- The geometry (including the landings) may have thickness, but only the top surface will be considered during the simulation.

In order to run a simulation, the escalator landings must be placed such that each goal line is between 0.01m and 0.20m above a floor, at least 0.20m from the edge of the floor. The two floors must be different (an escalator cannot be used to connect two portions of the same floor).

#### Impact on Agent Speed

Agent speed is modified to be exactly equal to the tread speed as specified in the escalator properties.

**Properties**

<b>General Tab</b>	
<b>Direction</b>	Defines the direction of the escalator. Escalators are always unidirectional (ie. either <b>Ball to Box</b> or <b>Box to Ball</b> ).
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A number greater than 1 will increase the perceived time cost of queuing, making agents more likely to seek alternate options when there is queuing. A number less than 1 will decrease perceived queuing costs making agents more likely to wait in a queue. See <a href="#">agent navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of <a href="#">surface maps</a> . Smaller values describe walkable space and barrier edges more accurately and can be necessary in confined spaces. Smaller values also increase memory and processing requirements during simulation initialization and execution.
<b>Physical: Rise angle</b>	The angle of incline measured from the horizontal. If automatically generated, the angle is measured using the horizontal run and vertical rise between goal lines.
<b>Tread Speed</b>	Defines the speed of the escalator along the incline, measured in metres per second.

<b>Access Tab</b>	
<b>Gate: Enable use as gate</b>	Configures the object as a gate allowing it to be opened or closed through the <a href="#">open gate event</a> . Agents cannot pass through a closed gate and will apply a 'cost of waiting' penalty when a route is closed.
<b>Gate: Default state</b>	Determine whether the starting/resting state is open or closed.
<b>Gate: Wait style</b>	Sets the waiting behaviour of agents while the gate is closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Gate: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Gate: Commit to wait</b>	Sets whether agents will commit to the gate once it has been chosen. When checked agents will wait for this object regardless of changes in route costs elsewhere on the floor.
<b>Limit Flow: Cap flow of entering agents</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity.

<b>Limit Flow: Rate type</b>	<p>Sets whether the flow rate is calculated as people/minute or people/minute/metre.</p> <p>If using people/minute/metre, the effective width (general tab property) is used.</p>
<b>Limit Flow: Max rate</b>	Sets the maximum flow rate. Flow rate will be measured according to the rate type shown above.

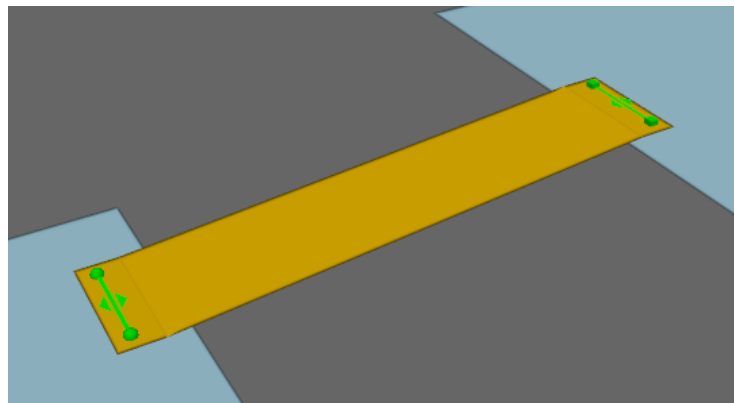
<b>Agent Behaviour tab</b>	
<b>Queue: Apply queuing force on approach</b>	When enabled, a 'queuing' force is applied to agents as they approach the escalator. This force nudges agents towards the area in front of the link which tends to result in trapezoid shaped queues. When disabled, queues tend to be more of a semicircle.
<b>Delay on enter</b>	<p>Sets how long an agent will be delayed before stepping onto an escalator. This can be set as a distribution in seconds. See <a href="#">single value distributions</a> for more information.</p> <p>If an escalator is gated and closes before the delay finishes, the agent will wait until the next time the escalator opens before attempting to step on again.</p>
<b>Delay on exit</b>	Sets how long an agent will be delayed before stepping off an escalator. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the floor.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the surface, avoiding obstacles and other agents.</p> <p><b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors.</p>
<b>Ignore Barriers</b>	Agents will walk through obstacles when traversing the surface. Obstacles are not included in the surface map. This is commonly used for connection objects like stairs, escalators, or ramps.
<b>Ignore neighbours</b>	Agents do not see other agents and will walk through them, making no attempt to avoid neighbors. Agents will take the shortest path to their goal. Density is calculated as if each agent was in isolation.
<b>Body Radius: Set agent radius</b>	If enabled the body radius for all agents on the surface will be changed to the specified value. This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the escalator will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.

Actions Tab	
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

Collections Tab	
<b>Bank</b>	The bank which the escalator is a member of, if any. For more information, see <a href="#">route banks</a> .
<b>Perimeters</b>	Perimeters which the escalator is a member of, if any. For more information, see <a href="#">perimeters</a> .

#### 4.3.1.10 Ramp

Ramps are [connection objects](#) that represent inclined surfaces and are used to connect two floors that are at different elevations. This change in elevation results in an additional vertical cost during agent route selection and can have an impact on agent speed during traversal. Ramps are less costly to traverse than stairs, but more costly than escalators.



A ramp

#### Restrictions on Geometry

- The ramp object itself should not include any handrails or other geometry that extends above the walkable surface. If these are desired, they should be added as separate [barrier](#) or [visual](#) objects.
- The ramp must have a flat landing area at each end; this should be at least 0.5m by 0.5m to allow construction of valid goal lines. Landings should be approximately rectangular or trapezoidal.

- The geometry (including the landings) may have thickness, but only the top surface will be considered during the simulation.

In order to run a simulation, the ramp landings must be placed such that each goal line is between 0.01m and 0.20m above a floor, at least 0.20m from the edge of the floor. The two floors must be different (a ramp cannot be used to connect two portions of the same floor).

### Impact on Agent Speed

By default, agent speed is modified as a function of the ramp angle and direction of travel (see table below). Speed can also be capped at a specific value through enabling the 'Limit Speed' properties. See [agent profile](#) for information on agent speed.

Direction of Travel	Angle X (degrees)	Percentage of Natural Speed
Up	$0 < X < 5$	100
Up	$5 \leq X < 10$	88.5
Up	$10 \leq X \leq 20$	Interpolated between 88.5 and 75
Up	$20 < X$	75
Down	Any	100

### Properties

General Tab	
<b>Direction</b>	Sets the direction in which agents can traverse the ramp.  <b>Two way:</b> Agents may cross in both directions <b>Unidirectional:</b> Agents may only cross in the specified direction.
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A number greater than 1 will increase the perceived time cost of queuing, making agents more likely to seek alternate options when there is queuing. A number less than 1 will decrease perceived queuing costs making agents more likely to wait in a queue. See <a href="#">agent navigation</a> for more information.
<b>Costs: Opposing flow multiplier</b>	Adjusts the penalty for links with agents moving across the object in the opposite direction. Numbers greater than 1 amplify the perceived cost from significant counter flows. Numbers less than 1 reduce the perceived cost. See <a href="#">agent navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of <a href="#">surface maps</a> . Smaller values describe walkable space and barrier edges more accurately and can be necessary in confined spaces. Smaller values also increase memory and processing

	requirements during simulation initialization and execution.
<b>Physical: Rise angle</b>	The angle of incline measured from the horizontal. If automatically generated, the angle is measured using the horizontal run and vertical rise between goal lines. The magnitude of the rise angle can have an impact on agent speed (see table above).

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate allowing it to be opened or closed through the <a href="#">open gate event</a> . Agents cannot pass through a closed gate and will apply a 'cost of waiting' penalty when a route is closed.
<b>Gate: Default state</b>	Determine whether the starting/resting state is open or closed.
<b>Gate: Wait style</b>	Sets the waiting behaviour of agents while the gate is closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Gate: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Gate: Commit to wait</b>	Sets whether agents will commit to the gate once it has been chosen. When checked agents will wait for this object regardless of changes in route costs elsewhere on the floor.
<b>Limit flow</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity. The maximum rate can be specified as an absolute rate or a rate based on width.
<b>Priority: Enable priority access</b>	Enables a limited yield system where agents moving in one direction receive preferential access. This is useful for constrained geometry where the majority of the flow is in one direction, or for cases such as train doors where alighting passengers often have priority over those boarding.
<b>Priority: Primary direction</b>	<p>Sets the direction in which agents will have primary access. The counter-flow direction will yield.</p> <p><b>One way (Ball to Box or Box to Ball):</b> Agents moving in the counter-flow direction will not enter the object when there are agents moving in the priority direction with priority access. If <b>Primary will yield</b> is set, agents traveling in the primary direction will yield if they arrive at the object while it is already being used by agents in the counter-flow direction. This is useful for scenarios such as ladders where usage should be restricted to one direction at a time.</p> <p><b>Two way:</b> Agents in either direction can claim priority access. Priority access is maintained for a given direction until there are no more agents crossing in that direction.</p>
<b>Priority: Capture range</b>	Sets the distance in metres from which agents approaching the object can capture priority access.

<b>Priority: Move aside</b>	When enabled, agents in the counter-flow direction will move aside to accommodate the priority flow.
<b>Priority: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Priority: Commit to wait</b>	When enabled, agents that have chosen the object will continue to wait until they have access regardless of changes to costs in other routes on the floor.

<b>Agent Behaviour tab</b>	
<b>Queue: Apply queuing force on approach</b>	When enabled, a 'queuing' force is applied to agents as they approach the ramp. This force nudges agents towards the area in front of the link which tends to result in trapezoid shaped queues. When disabled, queues can be more of a semicircle.
<b>Delay on enter</b>	<p>Sets how long an agent will be delayed before stepping onto an ramp. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.</p> <p>If a ramp is gated and closes before the delay finishes, the agent will wait until the next time the ramp opens before attempting to step on again.</p>
<b>Delay on exit</b>	Sets how long an agent will be delayed before stepping off a ramp. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the ramp.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the surface, avoiding obstacles and other agents.</p> <p><b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors.</p>
<b>Ignore barriers</b>	Agents will walk through obstacles when traversing the surface. Obstacles are not included in the surface map. This is commonly used for connection objects like stairs, escalators, or ramps.
<b>Ignore neighbours</b>	Agents do not see other agents and will walk through them, making no attempt to avoid neighbors. Agents will take the shortest path to their goal. Density is calculated as if each agent was in isolation.
<b>Body radius</b>	If enabled the body radius for all agents on the surface will be changed to the specified value (m). This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the ramp will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.

<b>Direction bias</b>	If enabled, the direction bias of agents on the surface will be changed to the specified value. This can be useful for corners or narrow spaces where the natural system wide direction bias is locally inappropriate. See <a href="#">agent profile</a> for more information on Direction Bias.
<b>Speed limit</b>	Enables a maximum threshold for agent speed (m/s). Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed density</b>	If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a> .

Actions Tab	
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

Collections Tab	
<b>Bank</b>	The bank which the ramp is a member of, if any. For more information, see <a href="#">route banks</a> .
<b>Perimeters</b>	Perimeters which the ramp is a member of, if any. For more information, see <a href="#">perimeters</a> .

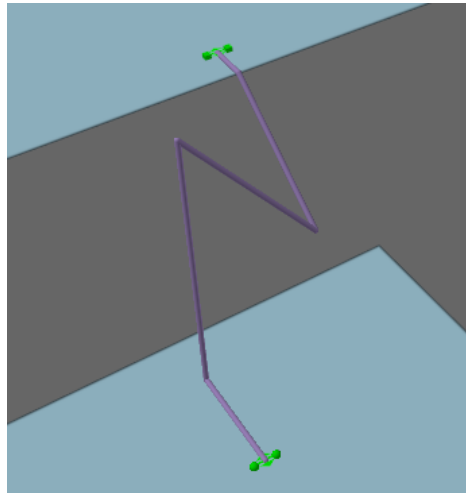
#### 4.3.1.11 Path

Paths are [connection objects](#) consisting of a simple curve. Agents can enter the curve at one end, follow along the curve in single file, and exit at the far end. While on a curve agents will either move at their desired speed or follow immediately behind the agent in front. Agents on the curve will ignore agents not following the curve or on the same curve but moving in the opposite direction. Agents not on the curve will attempt to avoid agents on the curve.

Agents on a path will see each other as cylinders with a radius equal to their body radius. Agents following behind another agent will maintain a "Queue Spacing" (set in Agent Behaviour) between the surfaces of the cylinders.



The default new path is a 2m horizontal line segment. This line segment can be extended by growing the end vertices, or splitting the edge into multiple pieces. See [working with lines](#) for more information on editing paths.



A path

#### Restrictions on Geometry:

- A path must contain a single curve object.
- There are no restrictions on the path described by the curve provided that it is continuous. Curves can be horizontal or vertical and can intersect with other objects or themselves.
- Each end point of the curve must be positioned just *above* a separate floor.
- The final line segment at either end of the path should be roughly horizontal so as to produce valid goal lines.

#### Impact on Agent Speed

By default, agent speed is unaffected by path traversal. Agents will slow down to follow an agent that is immediately in front. Speed can be capped at a specific value through enabling the 'Limit Speed' property. See [agent profile](#) for information on agent speed. In highly congested areas, boarding rates may be artificially limited by speed density constraints. Setting the speed density to 'Unconstrained' can remove these constraints and result in higher boarding rates.

#### Properties

General Tab	
<b>Direction</b>	Sets the direction in which agents can traverse the path.  <b>Two way:</b> Agents may cross in both directions <b>Unidirectional:</b> Agents may only cross in the specified direction.
<b>Costs: Distance added</b>	Adds a distance penalty to all routes that pass through the object. When choosing a route, agents will perceive routes that include objects with a positive distance penalty as longer than they actually are. Negative distance penalty values make routes appear shorter. Distances are measured in metres. See <a href="#">agent navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A number greater than 1 will increase the perceived time cost of queuing, making agents more likely to seek

	alternate options when there is queuing. A number less than 1 will decrease perceived queuing costs making agents more likely to wait in a queue. See <a href="#">agent navigation</a> for more information.
<b>Costs: Opposing flow multiplier</b>	Adjusts the penalty for links with agents moving across the object in the opposite direction. Numbers greater than 1 amplify the perceived cost from significant counter flows. Numbers less than 1 reduce the perceived cost. See <a href="#">agent navigation</a> for more information.

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate allowing it to be opened or closed through the <a href="#">open gate event</a> . Agents cannot pass through a closed gate and will apply a 'cost of waiting' penalty when a route is closed.
<b>Gate: Default state</b>	Determine whether the starting/resting state is open or closed.
<b>Gate: Wait style</b>	Sets the waiting behaviour of agents while the gate is closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Gate: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Gate: Commit to wait</b>	Sets whether agents will commit to the gate once it has been chosen. When checked agents will wait for this object regardless of changes in route costs elsewhere on the floor.
<b>Limit flow</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity. The maximum rate can be specified as an absolute rate or a rate based on width.
<b>Priority: Enable priority access</b>	Enables a limited yield system where agents moving in one direction receive preferential access. This is useful for constrained geometry where the majority of the flow is in one direction, or for cases such as train doors where alighting passengers often have priority over those boarding.
<b>Priority: Primary direction</b>	<p>Sets the direction in which agents will have primary access. The counter-flow direction will yield.</p> <p><b>One way:</b> Agents moving in the counter-flow direction will not enter the object when there are agents moving in the priority direction with priority access. If <b>Primary will yield</b> is set, agents traveling in the primary direction will yield if they arrive at the object while it is already being used by agents in the counter-flow direction. This is useful for scenarios such as ladders where usage should be restricted to one direction at a time.</p> <p><b>Two way:</b> Agents in either direction can claim priority access. Priority access is maintained for a given direction until there are no more agents crossing in that direction.</p>

<b>Priority: Capture range</b>	Sets the distance in metres from which agents approaching the object can capture priority access.
<b>Priority: Move aside</b>	When enabled, agents in the counter-flow direction will move aside to accommodate the priority flow.
<b>Priority: Cost of waiting</b>	Sets the penalty (in seconds) for agents waiting. If this penalty is too high agents will seek alternative routes to their goal. See <a href="#">agent navigation</a> for more information.
<b>Priority: Commit to wait</b>	When enabled, agents that have chosen the object will continue to wait until they have access regardless of changes to costs in other routes on the floor.

<b>Agent Behaviour Tab</b>	
<b>Queue spacing</b>	This determines how closely each agent will follow a previous agent. This can be set as a distribution in metres, see <a href="#">single value distributions</a> for more information.
<b>Queue: Apply queuing force on approach</b>	When enabled, a 'queuing' force is applied to agents as they approach the path. This force nudges agents towards the area in front of the link which tends to result in trapezoid shaped queues. When disabled, queues can be more of a semicircle.
<b>Delay on enter</b>	<p>Sets how long an agent will be delayed before stepping onto a path. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.</p> <p>If a path is gated and closes before the delay finishes, the agent will wait until the next time the path opens before attempting to step on again.</p>
<b>Delay on exit</b>	Sets how long an agent will be delayed before stepping off an path. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Traversal type:</b>	<p>Defines the manner in which agents will cross the path.</p> <p><b>Standard walk:</b> Agents will walk at their desired speed across the surface, avoiding obstacles and other agents.</p> <p><b>Teleport:</b> Agents will jump instantly across the surface to their intended goal, ignoring obstacles and other neighbors. Route distance calculations will ignore the internal geometry of the path and measure the straight line distance between the two path end points.</p>
<b>Body radius</b>	If enabled the body radius for all agents on the surface will be changed to the specified value (m). This can be useful in areas where crowding conditions are governed by geometric or behavioural conditions outside of normal open space standards. Agents approaching the path will slowly (over a range of 2-3m) transition to the new radius to avoid instantaneous changes. This setting should be used with caution as a) it applies to all agents regardless of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.

	Agents on a path will see each other as cylinders with a radius equal to their body radius and will maintain a space between each other set by their "Queue Spacing".
<b>Speed limit</b>	Enables a maximum threshold for agent speed (m/s). Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed density</b>	If enabled, agents will use the specified speed density relationship rather than the default provided by their profile. For more information on speed density, see the 'Movement' property in <a href="#">agent profile</a> .

Actions Tab	
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the ball side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On enter</b>	The action will be applied to all agents as they enter the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .
<b>Ball: On exit</b>	The action will be applied to all agents as they leave the box side of the object. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

Collections Tab	
<b>Bank</b>	The bank which the path is a member of, if any. For more information, see <a href="#">route banks</a> .
<b>Perimeters</b>	Perimeters which the path is a member of, if any. For more information, see <a href="#">perimeters</a> .

#### 4.3.1.12 Elevator

Elevators connect two or more floors vertically in the scene. Each connected floor is represented as a stop. There is a single cab or car which carries agents between stops, moving up and down within the elevator shaft.

Stops must be specified in the elevator properties. Doors are positioned automatically where the elevator touches the specified floors. Doors are designated as either ball or box doors based on the side of the elevator to which they connect. Elevator sides are identified by small ball and box icons at the top and bottom of the elevator shaft.

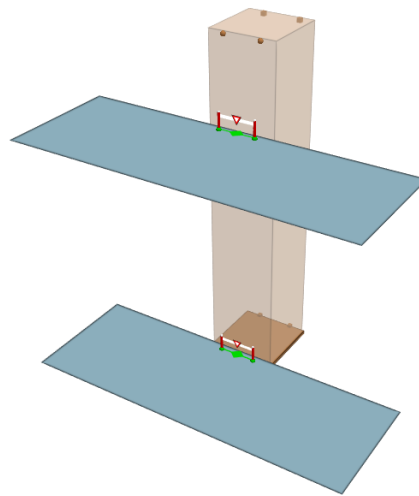
[Elevator behaviour](#) can be controlled in isolation, or in coordination with other elevators by inclusion in an [elevator bank](#). Connected stops may or may not be used depending on the operational

behaviour of the elevator.

### Route Cost

Agents consider elevators as they would any other scene object when evaluating the best route to their goal. The cost of using an elevator is based on a number of factors:

- **Vertical Travel:** The time it will take the elevator cab to move between two floors
- **Queue:** The time it will take for agents ahead in the queue to enter the elevator (assuming infinite capacity)
- **Cycle count:** The number of cycles expected before the agent can board (based on queue size and elevator capacity).
- **Cycle duration:** The expected time it will take the elevator to complete one full cycle/trip
- **Access cost when false:** A penalty applied to the elevator when the access test is enabled and evaluates to false.



An elevator

### Restrictions on Geometry:

- An elevator must be a rectilinear prism.
- The elevator must be touching (but not overlapping) each floor where the door for that stop should appear.
- A single floor can only be used for one stop/door (the same floor cannot be used on both sides of the elevator).

### Properties

General Tab	
<b>Costs: Distance added</b>	This distance is applied to the measured distance between any two stops, increasing or decreasing the expected travel time used to determine the vertical cost of taking the elevator. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information on how agents use distance in cost calculations.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A higher number will increase the perceived time cost of queuing, which makes agents more likely to select alternative routes. This multiplier is independent of the cycle time and only affects how long it would take for queuing agents to move into the elevator if the elevator remained at the stop and had infinite capacity. See <a href="#">Agent</a>

	<a href="#">Navigation</a> for more information.
<b>Costs: Cycle duration</b>	The expected average time it will take the elevator to complete one full cycle. Agents use this to predict how long they will have to wait for a departing full elevator to return to the same stop. The duration can be specified manually, or calculated automatically. If automatic, the duration will be based on the behaviour type, the listed stops, the cab speed, and door timings. Membership in a bank reduces the automatic cycle duration as the predicted wait time is for the next bank member to arrive at the stop.
<b>Transpose doors</b>	Rotate the sides used for ball/box stops by 90 degrees.
<b>Direction</b>	<b>All stops two-way:</b> All listed stops can be used to both board and alight the elevator (depending on operational behaviour). <b>Specify stops by direction:</b> Stops are listed explicitly by direction.
<b>Ball Side</b>	Floors that act as stops on the ball side of the elevator.
<b>Box Side</b>	Floors that act as stops on the box side of the elevator.

<b>Cab Tab</b>	
<b>Capacity</b>	<p>The maximum number of agents permitted in the elevator cab. This can be set as a number, or calculated based on area.</p> <p><b>Number of agents:</b> <i>Available Capacity = Max capacity - Elevator population</i></p> <p><b>Area of cab or Custom area:</b> <i>Available Capacity = ( Max area - occupied area ) / average agent area</i></p> <p>Occupied area is the sum of the area of all agents in the elevator and includes any variations in individual agent size. The average area is specified by the user and defaults to 0.25m. An agent will always take an accurate measure of how much space is free in an elevator, but will use the default agent size rather than their own to determine whether or not they themselves will fit in that available space.</p>
<b>Door width</b>	The width of the elevator door (m).
<b>Speed</b>	The maximum speed of the elevator cab (m/s).
<b>Acceleration</b>	The maximum acceleration of the elevator cab (m/s <sup>2</sup> ).
<b>Jerk</b>	The constant jerk used when moving the elevator cab (m/s <sup>3</sup> ).
<b>Door Timing: Opening</b>	Duration it takes the doors to open (s).
<b>Door Timing: Open Unused</b>	Duration doors remain open if no agent crosses the threshold (s).
<b>Door Timing: Open After Use</b>	Duration doors remain open after an agent crosses the threshold (s).

<b>Door Timing: Closing</b>	Duration it takes the doors to close (s).
-----------------------------	---

Operation Tab	
<b>Elevator Bank</b>	The bank to which the elevator belongs. The bank will control elevator behaviour and general operation. If no bank is specified this elevator will control its own movement.
<b>Rest Floor</b>	The starting position of the elevator cab, and the floor to which it will return if specified in the behaviour properties.
<b>Behaviour Type</b>	The control scheme governing <a href="#">Elevator Behaviour</a> : <b>Call/Answer:</b> The elevator responds to up/down button presses at origin stops. <b>Ordered list:</b> The elevator moves continuously between the specified stops in order. <b>Sabbath:</b> The elevator moves continuously between all connected stops in order. <b>Shuttle:</b> The elevator moves between a source floor and the destinations until the source floor is empty.

Access Tab	
<b>Waiting: Style</b>	Sets the waiting behaviour of agents while the elevator doors are closed. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Flow In</b>	When enabled, the flow of entering agents is not permitted to exceed the specified rate. If demand exceeds capacity, agents are held at the goal line until there is available capacity.
<b>Flow Out</b>	The flow of agents exiting the elevator is not permitted to exceed the specified rate.
<b>Access by Test</b>	When enabled, agents can only use the elevator if the test evaluates to true. If enabled and the test evaluates to false, agents will always perceive the elevator as closed regardless of where it is. They will not attempt to call the elevator and a penalty will be applied to any routes using the elevator. When part of an elevator bank, all elevators must use the same test.
<b>Cost when false</b>	A time penalty added to route costing when the elevator access test evaluates to false.

#### 4.3.1.12.1 Elevator Behaviour

An [elevator](#) can behave in different ways depending on the chosen control scheme. When an elevator is part of an [elevator bank](#), control is determined by the bank.

##### Call/Answer

The elevator will remain idle until called to a stop by an agent at a floor. Agents making a call must indicate whether they wish to travel up or down. Once called, the elevator will move to the call stop,

allow the agent to board, then take the agent to the desired destination floor. If multiple calls are received, the elevator will handle the calls in a sorted order such that it will finish all calls in the same direction before turning around.

Call/Answer	
<b>Return to rest floor</b>	When there are no calls outstanding and the elevator is empty, return to the rest floor. An elevator can be interrupted by a call when returning to the rest floor.
<b>Leave doors open</b>	When empty and idle, leave doors open.

**Ordered list**

The elevator will move through the specified stops, servicing each in order regardless of whether agents are present.

Ordered List	
<b>Visit</b>	The order in which floors are serviced: <b>In specified order:</b> The floors are visited in the order in which they are listed. <b>Bottom to top:</b> The floors are visited from the lowest to the highest. <b>Top to bottom:</b> The floors are visited from the highest to the lowest.
<b>Stop at Floors</b>	The floors at which the elevator will stop. Stops not listed here will not be serviced by the elevator.
<b>Repeat</b>	<b>Bounce:</b> Move through stops in order, then in reverse order. <b>Loop:</b> Move through stops in order, then return directly to the first and start again.

**Sabbath**

The elevator will move through all connected stops, servicing each in order regardless of whether agents are present. The elevator will service all stops in one direction then turn and service the same stops in the opposite direction.

**Shuttle**

The elevator will remain idle until called to one of the source floors by an agent. Agents making a call must indicate whether they wish to travel up or down. Once called, the elevator will move to the stop, allow the agent to board, then take the agent to one of the destination floors. Source floors are sorted by priority. The elevator will ignore calls from lower priority source floors while servicing or responding to a call from a higher priority source floor.

Shuttle	
<b>Return to rest floor</b>	When there are no calls outstanding and the elevator is empty, return to the rest floor. An elevator can be interrupted by a call when returning to the rest floor.
<b>Leave doors open</b>	When empty and idle, leave doors open.
<b>Visit</b>	The order in which source floors are prioritized: <b>In specified order:</b> The order in which they are listed.



	<b>Bottom to top:</b> From the lowest to the highest. <b>Top to bottom:</b> From the highest to the lowest.
<b>Stop at Floors</b>	The floors at which the elevator will respond to calls for boarding.
<b>Safe Floors</b>	The destination floors to which the elevator can take agents. Agents will not be able to board at destination floors.

#### 4.3.1.13 Cordon

Cordon objects count the number of agents crossing a plane. Cordons can be used by [triggers](#) during a simulation, or as [transitions](#) during [analysis](#).

A cordon will by default count agents crossing in both directions. A cordon can be configured to only count crossings in a single direction by changing the 'Direction' property.

##### Notes on geometry

- Must be a single polygon mesh object or a collection of polygon mesh objects. There are no other restrictions on shape or orientation; cordons can be horizontal, vertical, curved or even closed volumes (in which case agents will be counted when passing into or out of the volume).
- Agents will be considered 'at' or 'crossing' the cordon when the centre point between their feet crosses any surface of the cordon object.

##### Properties

General Tab	
<b>Direction</b>	<b>Bidirectional:</b> Crossings will be counted for agents moving in any direction. <b>Unidirectional:</b> Only crossings in the designated direction will be counted (direction indicated by arrows in the scene view).

#### 4.3.1.14 Volume

Volume objects are used to identify when agents are inside an arbitrary area. Volume objects do not need to be rectilinear or even continuous. Agents are considered inside a volume if they are inside any of the component parts of a volume.

Agents on floors with traversal type set to "Teleport" will not be counted in volumes.

##### Notes on geometry

- Must be a single polygon mesh object or a collection of polygon mesh objects.
- Agents will be considered 'in' the volume if the centre point between their feet is within any of the volumes defined by the meshes.

##### Properties

There are no properties for objects of this type.

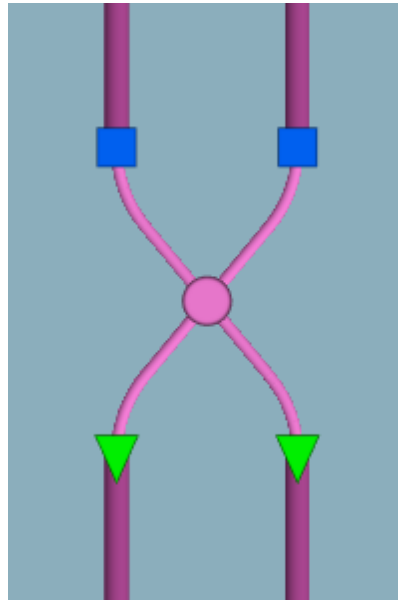
#### 4.3.1.15 Visual

Visual objects can provide context to an environment or enhance the look of a scene. Visual objects are not used during simulation or analysis and have no functional impact on a project.

### 4.3.1.16 Dispatch

Dispatch objects connect servers together into process chains and distribute agents across server inputs.

Dispatch objects are automatically created by connecting server entry and exit points together in the "Edit process chains" mode. Please refer to the overview of [process chains](#) for more information on how to connect server objects. **The location of dispatch objects has no effect on the movement of agents.**



A dispatch connecting two sources and two sinks

Dispatch Properties	
<b>Assign Type</b>	<p>Defines the method used to distribute agents across downstream servers:</p> <p><b>Largest available capacity:</b> Choose the server with the most space. If all servers have limited capacity this is the max capacity minus the current population. If any server is not set to limited capacity its capacity is assumed to be infinite and it will always be chosen.</p> <p><b>Smallest cost:</b> Choose the server for which processing will incur the lowest cost. Cost is calculated using the time it will take to walk to the server, the average time it will take to be processed by the server, and any time that will be spent waiting for agents already on the server. Processing time is taken from the average of the default contact time distribution (test distributions are ignored).</p> <p><b>Smallest distance:</b> Choose the server that is closest to the agent.</p> <p><b>Smallest queue:</b> Choose the server with the smallest queue.</p> <p><b>Randomly:</b> Choose the server randomly.</p> <p>Note that preferred and required test restrictions for server access will be respected for all assignment methods.</p>

<b>Sources</b>	The list of connected upstream servers which feed into this dispatch. Sources can be added or removed using the multi-object chooser in the Dispatch properties page or via the connect and disconnect functions in the "Edit process chains" mode in the main application window.
<b>Sinks</b>	The list of connected downstream servers to which agents will be dispatched. Sinks can be added or removed using the multi-object chooser in the Dispatch properties page or via the connect and disconnect functions in the "Edit process chains" mode in the main application window.

### 4.3.2 Collections

A collection is an object which references a set of member objects. Some collections have an impact on simulation execution. All collections can be used as simple containers for controlling scene object visibility in bulk or referencing a set of objects all at once. A collection cannot be a member of another collection.

Members of a collection can be selected using "Find -> Collection Members" in the right click menu or the "Find all members" button in the collection properties.

#### Types of Collections

Collection Type	Description
<a href="#">Collection</a>	A collection of any type of object (except other collections).
<a href="#">Elevator Bank</a>	A collection of elevator objects that service similar stops in a coordinated manner.
<a href="#">Perimeter</a>	A collection of connection objects encircling a conceptual area of interest. Agents cannot cross the same perimeter twice.
<a href="#">Route Bank</a>	A collection of connection objects which connect the same two objects. Banks help agents navigate clusters of similar and closely placed connections.
<a href="#">Transform</a>	A collection of <a href="#">reference geometry</a> objects which were imported from the same file. These are created automatically by importing geometry.
<a href="#">Zone</a>	A collection of walkable objects which define a conceptual area in the simulation. Connected links and portals can be automatically included.

#### Collections as parameters

A collection can be chosen in most places where multiple objects are expected (such as the gated [links](#) for an [open gate event](#)). Choosing a collection is similar to individually choosing its various members. When a collection contains members that do not make sense for the given choice, those members are ignored. For instance, a [general collection](#) called "EastWing" might include a combination of links, floors and portals. Using that collection within an open gate event would only make use of the gated links, while all other members would be ignored.

The type of objects used from a collection will be indicated by appending a suffix to the collection

name. For example, a [map query](#) choosing a [general collection](#) called "EastWing" will refer to the collection as "EastWing.Surfaces" in its properties.

**List of collection suffixes**

Suffix	Description
<b>Areas</b>	<a href="#">Area</a> objects: walkable objects ( <a href="#">floors</a> , <a href="#">links</a> , etc.) or <a href="#">volumes</a> .
<b>Connections</b>	<a href="#">Connection objects</a> (ie. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">escalators</a> , <a href="#">ramps</a> and <a href="#">paths</a> ).
<b>Cordons</b>	<a href="#">Cordons</a>
<b>ExitPortals</b>	<a href="#">Portals</a> which are set to be destinations.
<b>Gates</b>	<a href="#">Links</a> and other connection objects which have gated access.
<b>Portals</b>	<a href="#">Portals</a>
<b>Profiles</b>	<a href="#">Profiles</a>
<b>SceneObjects</b>	<a href="#">Scene Objects</a>
<b>Servers</b>	<a href="#">Servers</a>
<b>SimRuns</b>	<a href="#">Simulation Runs</a>
<b>Surfaces</b>	Paintable surfaces, used for <a href="#">map queries</a> . Is usually only walkable objects, but the <a href="#">vision time map</a> can also paint barriers.
<b>TripPoints</b>	Possible destinations for a <a href="#">trip</a> (ie. <a href="#">areas</a> , <a href="#">cordons</a> , <a href="#">portals</a> and servers). When used in a <a href="#">trip</a> , reaching any of the members will count as fulfilling that stage of the trip.
<b>Tokens</b>	<a href="#">Tokens</a>
<b>Volumes</b>	<a href="#">Volumes</a>
<b>Walkables</b>	Walkable objects ( <a href="#">floors</a> , <a href="#">links</a> , etc.).

**4.3.2.1 Collection**

A collection is a pure container that serves no purpose other than the grouping of member objects. Any non-collection object can be a member. Objects can belong to more than one collection.

Members can be given weights which determine the likelihood a member will be chosen when the collection is used as an input to another object.

Collection Properties	
<b>Name</b>	The name of the collection.

<b>Objects</b>	Collection members. Any object which is not itself a type of <a href="#">collections</a> can be a member. Members can be given associated weights. See <a href="#">choosing objects</a> for more details.
----------------	---

#### 4.3.2.2 Elevator Bank

An elevator bank is a collection of [elevator](#) objects. The bank assumes control of member elevators and coordinates behaviour to ensure members function as a team. See [elevator behaviour](#) for the different ways in which elevators can be controlled.

During navigation, agents will treat members of an elevator bank as roughly equal in cost and will take the first elevator that becomes available.

Elevator banks should only contain elevators that are close in proximity and service similar floors.

General Tab	
<b>Elevators</b>	The <a href="#">elevator</a> objects controlled by the elevator bank.
<b>Behaviour Type</b>	<p>The control scheme governing <a href="#">elevator behaviour</a>. The bank behavior will replace any behaviour settings on the individual elevators.</p> <p>Bank behaviours include:</p> <p><b>Call/Answer:</b> The elevators respond to up/down button presses at origin stops.</p> <p><b>Sabbath:</b> The elevators move continuously between all connected stops in order.</p> <p><b>Shuttle:</b> The elevators move between a source floor and the destinations until the source floor is empty.</p>

Access Tab	
<b>Access by Test</b>	<p>When enabled, agents can only use the elevators if the test evaluates to true. If enabled and the test evaluates to false, agents will always perceive the elevator as closed regardless of where it is, they will not attempt to call the elevator, and a penalty will be applied to any routes using the elevator.</p> <p>The bank test will replace any specific member elevator test.</p>
<b>Cost when false</b>	A time penalty added to route costing when the elevator access test evaluates to false.

Routing Tab	
<b>Bypass Route</b>	A list of objects, all connected together, which provide an alternate route between the floors serviced by the elevator. Elevator stop floors cannot be part of the bypass route. This is typically used to identify a stair column or set of floors and escalators which an agent might consider when the elevator is busy. The purpose of adding the objects as a bypass route is to ensure that once an agent chooses to take the bypass route, it stays in the bypass route and does not return to the elevator at another floor.

### 4.3.2.3 Perimeter

Perimeters are collections of connection objects which reduce the number of available routes through a scene.

There are two common situations where this can be useful:

1. When the number of possible route permutations is so large that simulation initialization requires hours to complete. This usually occurs when there is a very large number of connections in the scene.
2. When certain regions of the simulation should not be used as shortcuts by agents 'just passing through' (such as with fare paid zones in a transit station).

Perimeters can consist of [connection objects](#) (ie. [links](#), [stairs](#), [ramps](#), [escalators](#) and [paths](#)). When a route involves crossing a link that is a member of a perimeter, it is said to be a route which crosses that perimeter. A route will be considered invalid (and so not available to agents) if it crosses the same perimeter more than once.

Connection objects maybe be part of several different perimeters.

#### Impact on available routes

Given a region of floors completely enclosed by members of a perimeter, access across the boundary to that region will be permitted only to those agents either originating inside the region or seeking a destination within the region. Agents wanting to simply pass through the region will be forced to go around. Figure 1 demonstrates the effect of perimeters on available routes through a network.

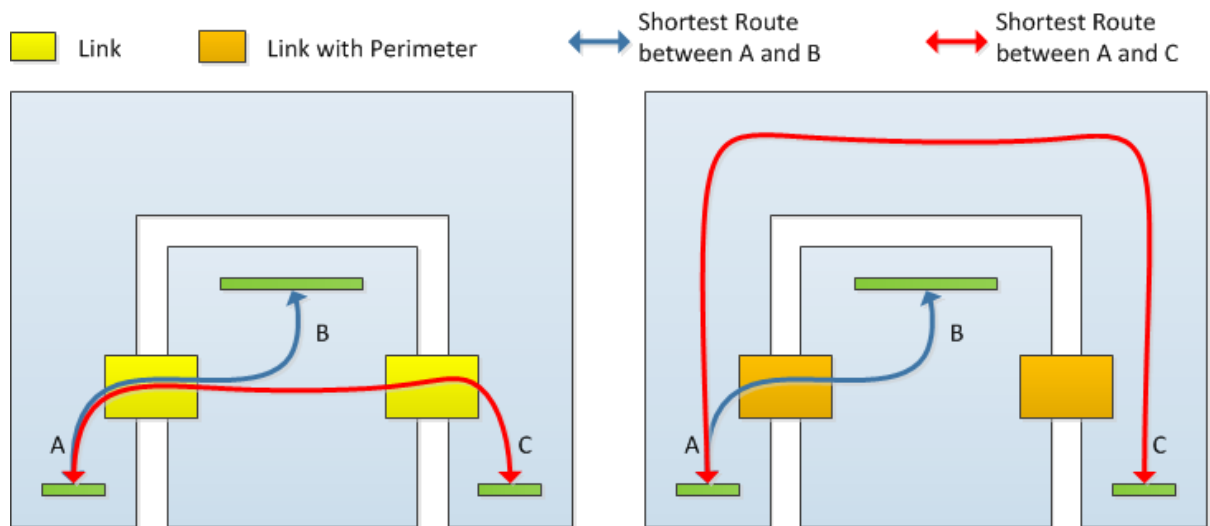
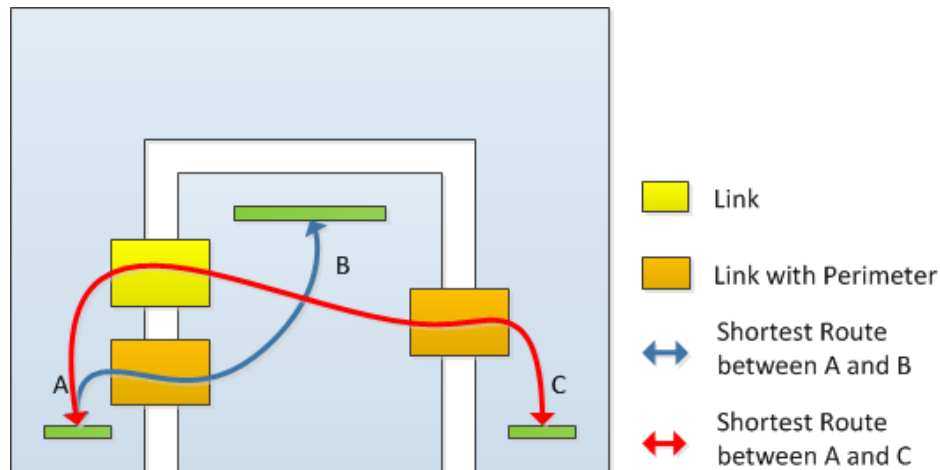


Figure 1: Example floor plan with and without a perimeter.

Care must be taken when constructing perimeters. It is important that all links that can be used to access an area are included in the perimeter. If there is a single link left out, the perimeter is said to have a hole, and will not be effective. The impact of a hole on a perimeter is shown in Figure 2.



**Figure 2: Example floor plan with incomplete perimeter.**

Connection objects in series within a route should not be part of the same perimeter. For example, given a flight of stairs with two stairs connected by a simple floor landing, if both stairs are part of the same perimeter, the flight of stairs will become unavailable for use.

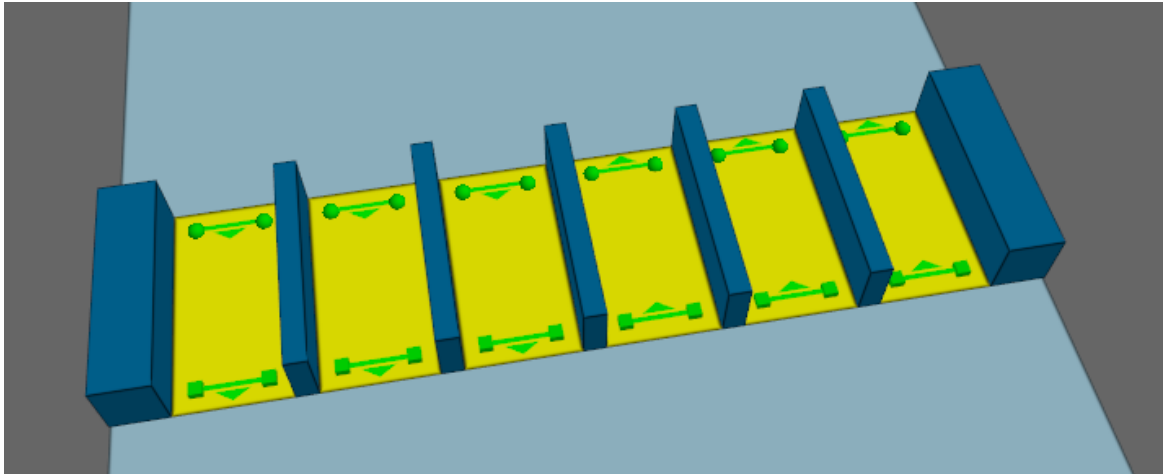
Perimeter Properties	
<b>Name</b>	The name of the perimeter.
<b>Objects</b>	Members of the perimeter. These must be <a href="#">connection objects</a> (i.e. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">escalators</a> , <a href="#">ramps</a> and <a href="#">paths</a> ).

#### 4.3.2.4 Route Bank

A route bank is a collection that contains [connection objects](#) such as [links](#), [stairs](#), [escalators](#), [ramps](#) and [paths](#) that are close together and connect the same two floors. A route bank groups the similar connections together and makes them appear as one choice to agents navigating the space.

During navigation, agents will attempt to choose the most convenient connection based on a number of factors including congestion, distance to the connection (near distance), and distance from the connection to the goal (downstream distance). In many cases this results in unequal distribution of agents along a set of very similar connections due to alignment of the incoming flow, unequal widths of links, or small (but largely irrelevant) differences in downstream distance.

When connections are banked, agents ignore downstream distance and focus exclusively on the remaining cost factors. This results in agents being better spread out over a large set of connections and improves flow across the bank.



**A row of turnstiles. Two banks should be created, one for the three links on the left, and another for the three links on the right.**

While the most common use of banks is for turnstiles or fare gate arrays, banks can also be useful for sets of vertical connections, such as stairs and escalators, where the objects are right beside one another. If both stairs and escalators are members of the same bank, agents will still prefer the escalators regardless of the bank.

**Bank Properties**

Properties	
<b>Name</b>	The name of the bank.
<b>Objects</b>	Members of the bank. These must be <a href="#">connection objects</a> (ie. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">escalators</a> , <a href="#">ramps</a> and <a href="#">paths</a> ).

**4.3.2.5 Transform**

Transform collections can be used to move, scale, or rotate member objects all at once. Transforms are automatically generated when importing geometry and will contain all imported [reference geometry](#) or drawing layers from a single file. In the case of reference geometry the transform also contains texture information that is shared amongst members.

Adding or removing a member will not change the member's current size, position, or orientation. Deleting a transform will delete all associated member objects from the scene.

Reference Model Properties	
<b>Bounding Box: Centre</b>	The centroid of all member geometry in the transform.
<b>Bounding Box: Size</b>	The maximum extents of member geometry.
<b>Transform: Scale</b>	Apply the scale factor to all members, altering their size.  This scale can assist with converting between different unit systems. If the member geometry was originally modeled in inches, specify 'From inches'



	in the combo box, and a scale factor will automatically be calculated to convert the geometry from inches to the native MassMotion units of metres.
<b>Transform: Rotation</b>	Apply a rotation to all members. The rotation is applied after the scale.
<b>Transform: Translation</b>	Apply a translation to all members. The translation is applied after scale and rotation.
<b>Transform: Centre</b>	Sets a translation such that the bounding box centre is at the origin.
<b>Transform: Reset</b>	Does not move members. Return the scale to 1 and rotation and translation to (0,0,0) but keep all members at their current scale, rotation and translation.

#### 4.3.2.6 Zone

A zone is a collection of objects that define a conceptual area in the simulation. Zones are specified using a set of primary members, but then will automatically include additional secondary members based on the "Type" property setting as described below. The areas in a zone do not need to be connected to one another. Objects may be members of any number of zones.

All members, both explicitly and automatically chosen, can be found by using the "Find All Members" button, or by right clicking the zone in the [list view](#) and using "Find: Collection Members".

Zones are required for use in [evacuation events](#).

Zone Properties	
<b>Type</b>	<p>Determines the manner in which members are specified.</p> <ul style="list-style-type: none"> <li>• <b>Chosen objects:</b> Allows all walkable objects to be chosen as members. Automatically adds portals connected to member floors.</li> <li>• <b>Chosen floors and interior connections:</b> Allows only floors to be selected as members. Automatically adds portals connected to member floors as well as connection objects between any two member floors.</li> <li>• <b>Chosen floors and all connections:</b> Allows only floors to be selected as members. Automatically adds portals and connection objects touching any member floor.</li> </ul>
<b>Members</b>	<p>The explicitly chosen members of the zone. These may be <a href="#">floors</a>, <a href="#">links</a>, <a href="#">stairs</a>, <a href="#">escalators</a>, <a href="#">ramps</a> and <a href="#">paths</a> depending on the Type. Both the explicitly and implicitly chosen members of a zone may be selected by right-clicking on the zone in the <a href="#">list view</a> and selecting Find-&gt;Collection Members.</p>

### 4.3.3 Activities

Activity objects have to do with creating agents, defining agent characteristics and behaviour, and controlling elements of the scene.

Object Type	Description
<a href="#">Profile</a>	Used by events to define agent characteristics.
<a href="#">Avatar</a>	An optional editable physical representation of an agent.
<a href="#">Network</a>	An optional definition of the network objects available to the agent during navigation.
<a href="#">Token</a>	Identifying markers held by agents.
<a href="#">Events</a>	A class of objects for creating and controlling agents or modifying the scene.
<a href="#">Action</a>	An operation that can be applied to an agent to modify its properties or assign it new tasks.
<a href="#">Tally</a>	An object for storing, modifying, combining, or testing measured or abstract values
<a href="#">Test</a>	A condition which can evaluate the scene or an agent and return true or false.
<a href="#">Time</a>	A virtual event that can be used by <a href="#">reference times</a> in other events and can help with coordinating the timing of multiple events.
<a href="#">Trigger</a>	Used to control when events become active.

#### 4.3.3.1 Profile

Every agent is created with a set of unique properties that define the agent's physical characteristics and personality. The range of possible values for each agent is defined by the profile used to create that agent. See [single value distributions](#) for more information on distributions.

Events which create agents will have a "Profile" property which determines which profile will be used. Most events can also specify a [collection](#) of profiles, with manual weights specifying the likelihood a particular member profile will be assigned.

#### Profile Properties

The general properties of the agent define radius, speed, and movement characteristics.

General Tab	
<b>Profile</b>	Select from a number of preset values for agent radius and speed [1][2].
<b>Radius</b>	<p>The size of each agent. Given the default value of 0.25m, each agent, measured from one shoulder to the other, will be 0.5m across.</p> <p><b>Note:</b> care should be taken with this value. The simulation has been tuned to a value of 0.25m. While there are no theoretical limits, it is recommended that the practical body radius remain between 0.15m and</p>

	0.4m.
<b>Speed</b>	Each agent is assigned a desired speed according to the specified distribution. The agent will attempt to maintain this speed when moving freely in flat open space. See <a href="#">agent movement</a> for information on other factors affecting agent speed.
<b>Movement (Speed Density)</b>	Controls the relationship between density and speed:  <b>Fruin commuter:</b> Constrain agent speed based on density, resulting in flows tuned to match the data in John Fruin's Pedestrian Planning & Design [2]. <b>Unconstrained:</b> Do not constrain agent speed by density. This will result in potentially unrealistically high flow rates.
<b>Avatar (optional)</b>	Defines the physical appearance of agents produced by this profile. If no avatar is specified the default appearance is used. By default each scene view displays agents using animated bipeds rather than static avatars. See <a href="#">avatar</a> for information on viewing the static avatars during playback.
<b>Direction Bias</b>	When faced with an opposing flow, agents will tend to drift either left or right depending on local customs. The direction bias indicates the direction of drift. In <a href="#">agent movement</a> the collision avoidance, drift, and corner forces are all affected by the direction bias. Available options are: Left Strong, Left Weak, None, Right Weak, Right Strong.

The route choice properties determine the network available to the agent and how the agent will weight different cost factors as they navigate through the network.

Route Choice Tab	
<b>Accessibility Network</b>	Determines the routes available to the agent. By default an agent can use any object but access can be restricted by specifying a <a href="#">network</a> object.
<b>Route Cost Weights</b>	<p>Each weight distribution is used to determine the importance an agent will place on the given cost type. The greater the variation in a particular weight distribution, the greater the variation in decisions made by a given population. Please see <a href="#">agent navigation</a> for a description of the cost components.</p> <p><b>Horizontal :</b> Applied to the horizontal distance cost component. The horizontal distance is the sum of the distances across all objects along the route to the goal.</p> <p><b>Vertical:</b> Applied to the vertical distance component. The vertical distance is the sum of weighted vertical displacements across all objects along the route to the goal. Weights applied to the vertical displacements are based on object type.</p> <p><b>Queue:</b> Applied to the queue cost component. Queue costs apply only to queues that form for objects leading off of the current floor.</p>

	<p><b>Processing:</b> Applied to the closed penalty cost component. This includes all penalties from any gate or priority access 'wait when closed' penalties, delays, or controlled flow restrictions on objects leading off of the current floor.</p>
--	---

Actions Tab	
<b>On birth</b>	The action will be applied to all agents created with the profile as they enter the simulation. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

[1] London Underground’s Station Modelling with Legion Best Practice Guide 2 | 3 July 2009 – section 4.3

[2] Fruin, John J. Pedestrian Planning & Design, Revised Edition Chapt. 4, Elevator World, 1987 - Chapter 3

**4.3.3.2 Avatar**

Geometry that can be used to represent different populations of agents during simulation playback. The avatar has no functional impact on simulation execution and is for visualization purposes only.

Avatar geometry can be edited using a special scene viewer which is accessible through the avatar object properties window. All of the tools available for [editing scene geometry](#) can also be used when editing avatars. It is important that the avatar centroid remain at the origin or it will not accurately reflect actual agent movement during playback.

**Using Avatars**

An agent is assigned a particular avatar through the agent's [profile](#). If no avatar is specified, the agent will use a default representation.

**Viewing Avatars During Playback**

By default MassMotion displays agents using the built in animated biped avatar. To view user specified static avatars, change the agent type in the 3D scene view's display menu.

**4.3.3.3 Network**

A network object defines the options available to an agent as the agent navigates through the scene.

The default, or 'World' network contains all objects in the scene. Custom networks can be created to exclude certain objects or object types. For example, mobility impaired agents might be simulated by using a network that excludes all stairs.

Each agent is aware of a single network at a time. The network completely defines the agent's awareness of the world. Objects that are not in the network do not exist as far as the agent is concerned. Networks can be assigned at birth through the agent's [profile](#). Networks can also be

changed over the course of the simulation through the use of actions.

### Properties

Agents	
<b>Membership</b>	<p>Controls how the Network object will determine membership.</p> <p><b>All objects:</b> The network will contain all objects in the scene. Using a network with 'All objects' is the same as not using a network.</p> <p><b>By including types and objects:</b> The network will include all floors, all links, and any other type that is checked. If a type is not checked, objects of that type will not be included, unless the object is explicitly included in the object list.</p> <p><b>By including objects:</b> The network will only include the specified objects.</p> <p><b>By excluding objects:</b> The network will include all objects except those specified.</p>

#### 4.3.3.4 Token

Tokens are objects held by agents. An agent can hold only one of a given token, but can hold any number of different tokens. Tokens are used to identify certain agents primarily through the use of [tests](#).

Tokens are given or removed using [agent actions](#).

#### 4.3.3.5 Events

Events are objects which can create agents, modify agents, or modify elements in the scene. Events become active at specific times or in response to certain [trigger](#) based conditions in the scene.

### Creating Agents

Object Type	Description
<a href="#">Journey</a>	An event to create agents moving from one or more origins to one or more destinations.
<a href="#">Circulate</a>	An event to create agents who move between "circulation" portals.
<a href="#">Vehicle</a>	An event that simulates the regular arrival and departure of a "vehicle" such as a train.
<a href="#">Evacuate</a>	An event used to simulate evacuations. Agents are created and then evacuate the scene in specified ways.
<a href="#">Timetable</a>	An event used to create large groups of agents and control the scene.

**Controlling Agents and the Scene**

Object Type	Description
<a href="#">Broadcast</a>	An event which applies an <a href="#">agent action</a> to change the behaviour of agents.
<a href="#">Gate Access</a>	An event which opens/closes gated <a href="#">connection objects</a> .
<a href="#">Server Access</a>	An event which opens/closes the entrance and/or exit of servers.
<a href="#">Cache Change</a>	An event which can modify the cache value of a <a href="#">tally</a> .

4.3.3.5.1 Journey

Journey events are the most direct way to populate a scene. They create agents at an origin portal and instruct them to seek a destination portal. Once the agent has reached the destination portal they automatically exit the simulation.

The number of agents created by the event can be specified in different ways according to the population source type. Some methods involve creating a set number of agents for each origin or destination, while others distribute a population randomly across origins or destinations. When [collections](#) are used, the population is distributed randomly across members of the collection according to the collection weighting.

The same portal may be used as both an origin and destination, however, agents that are given the same portal as both origin and destination will exit the simulation immediately after being created.

**Active**

A journey is active over the interval for which it can create agents. This will depend on the population demand type. When using 'Instant' the event is active only for an instant. When using 'Random' or 'Evenly spaced', the event is active for the duration of the specified interval, including those frames when no agents are created. When using a schedule or weighted curve, the event is active from when the start condition is met until the final interval in the table is complete.

**Properties**

Agents	
<b>Timing: Start</b>	The condition for starting the event. See <a href="#">triggering events</a> for more information.
<b>Profile</b>	The <a href="#">profile</a> used to create agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to assign a distribution of profiles amongst the population.
<b>Population: Source</b>	Describes different methods for determining the number of agents. When agents are specified per origin or destination, collections are treated as a single entry with the count distributed over members according to the collection weights.

	<p><b>Count total:</b> Specify a single count of agents</p> <p><b>Count per origin:</b> Specify the number of agents to create at each origin portal (or collection).</p> <p><b>Count per destination:</b> Specify the number of agents that will be assigned to each destination portal (or collection).</p> <p><b>Origin table:</b> Specify a custom count of agents for each origin portal (or collection).</p> <p><b>Destination table:</b> Specify a custom count of agents to be assigned each destination portal (or collection).</p> <p><b>Origin destination matrix:</b> Specify custom counts for each origin/destination pair.</p> <p><b>Schedule:</b> Specify a sequence of interval/population pairs. The first interval begins when the event becomes active.</p>
<b>Population: Arrival</b>	<p>Describes how created agents are distributed over time.</p> <p><b>Evenly spaced:</b> The specified number of agents will arrive at a constant rate over the duration.</p> <p><b>Instant:</b> The specified number of agents will arrive all at once at the event start time.</p> <p><b>Random:</b> The specified number of agents will arrive according to the specified distribution over the duration. See <a href="#">duration distributions</a> for information on arrival distributions.</p> <p><b>Table:</b> A custom distribution can be described by a series of intervals and weights. Each row contains the duration of an interval and the fraction of the total agents to create over that interval. The first interval starts at the event start. Subsequent intervals begin when the previous interval ends. Agents are created according to a uniform distribution within each interval. Fractional values are automatically normalized.</p>
<b>Origins</b>	<p>The <a href="#">portals</a> at which agents will be created. Portals can be weighted to alter the distribution of agents across the set. See <a href="#">choosing objects</a> for more details.</p>
<b>Destinations</b>	<p>The <a href="#">portals</a> from which the agent will be given its initial goal.</p> <p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal which can be reached using the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p>

### Colours Tab

<b>Colours</b>	<p><b>Event colour:</b> Agents are assigned the same colour as the event object.</p> <p><b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour.</p> <p><b>Darken event colour:</b> Agents are assigned a darker version of the event object colour.</p> <p><b>Rule based:</b> Agents are assigned a colour according to the colouring rules. Working from top to bottom, agents will use the first colour for which the condition</p>
----------------	--

	<p>evaluates to true.  <b>Specified colour:</b> Agents are assigned the specified colour.</p>
--	---

Actions Tab	
<b>On birth</b>	<p>The action will be applied to all agents created by the event as they enter the simulation. Any tasks given by the action will be executed before the agent seeks its assigned destination. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>

**Collections**

[Collections](#) can be used in the "Origins" and "Destination" properties. The collections can be weighted, changing the distribution of agents going from/to various portals.

4.3.3.5.2 Circulate

Circulate events create agents that enter the simulation from one of the origin [portals](#), move between several circulation portals, then leave the simulation at one of the destination portals.

The same portal may be used as an origin, circulation and destination portal. Agents may visit the same circulation portal more than once.

With circulate events, the 'initial exit refers to the destination portal specified in the Agents tab rather than the first circulation portal chosen by the agent.

**Active**

A circulate event is active over the interval for which it can create agents. This will depend on the population demand type. When using 'Instant' the event is active only for an instant. When using 'Random' or 'Evenly spaced', the event is active for the duration of the specified interval, including those frames when no agent is created. When using a schedule or weighted curve, the event is active from when the start condition is met until the final interval in the table is complete. Once the event has finished creating agents it is no longer active, even if the created agents are continuing to circulate.

**Properties**

Agents Tab	
<b>Timing: Start Time</b>	<p>The condition for starting the event. See <a href="#">triggering events</a> for more information.</p>
<b>Population : Profile</b>	<p>The <a href="#">profile</a> used to create agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to assign a distribution of profiles amongst the population.</p>
<b>Population : Source</b>	<p>Describes different methods for determining the number of agents to create. When agents are specified per origin or destination, collections are treated as a single entry with the count distributed over members according to the collection weights.</p> <p><b>Count total:</b> Specify a single count of agents  <b>Count per origin:</b> Specify the number of agents to create at each origin portal (or collection).</p>



	<p><b>Count per destination:</b> Specify the number of agents that will be assigned to each destination portal (or collection).</p> <p><b>Origin table:</b> Specify a custom count of agents for each origin portal (or collection).</p> <p><b>Destination table:</b> Specify a custom count of agents to be assigned each destination portal (or collection).</p> <p><b>Origin destination matrix:</b> Specify custom counts for each origin/destination pair.</p> <p><b>Schedule:</b> Specify a sequence of interval/population pairs. The first interval begins when the event becomes active.</p>
<b>Population : Demand</b>	<p>The number of agents the event will create and when they will be created relative to the event start time.</p> <p><b>Evenly spaced:</b> The specified number of agents will arrive at a constant rate over the duration.</p> <p><b>Instant:</b> The specified number of agents will arrive all at once at the event start time.</p> <p><b>Random:</b> The specified number of agents will arrive according to the specified distribution over the duration. See <a href="#">duration distributions</a> for information on arrival distributions.</p> <p><b>Table:</b> A custom distribution can be described by a series of intervals and weights. Each row contains the duration of an interval and the fraction of the total agents to create over that interval. The first interval starts at the event start. Subsequent intervals begin when the previous interval ends. Agents are created according to a uniform distribution within each interval. Fractional values are automatically normalized.</p>
<b>Origins</b>	<p>The <a href="#">portals</a> at which agents will be created. Portals can be weighted to alter the distribution of agents across the set. See <a href="#">choosing objects</a> for more details.</p>
<b>Destinations</b>	<p>The <a href="#">portals</a> from which the agent will be given its ultimate goal. The agent will pursue this goal once it has finished circulating. Despite the fact that agents will seek this goal after circulating, it is still referred to as the 'Initial exit' for the purposes of agent actions and tests.</p> <p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal which can be reached using the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p>

Circulate Tab	
<b>Circulate (Type)</b>	<p>Defines the end conditions for agent circulation. When using a time or duration limit, the agent will cease circulating immediately once the limit is reached regardless of whether the agent is currently waiting or in transit between circulation portals.</p> <p><b>For entire simulation:</b> The agent will continue circulating amongst the portals until the simulation is complete.</p> <p><b>For count:</b> The agent will circulate for the specified number of iterations.</p>

	<p><b>For duration:</b> The agent will circulate for the specified duration (measured from the time the agent is created).</p> <p><b>For duration or count:</b> The agent will circulate until the specified duration or for the specified number of iterations (see 'Wait After Count' below).</p> <p><b>Until time:</b> The agent will circulate until the specified simulation time. Agents created after the time will proceed immediately to their destination.</p> <p><b>Until time or count:</b> The agent will circulate until the specified simulation time or for the specified number of iterations (see 'Wait After Count' below).</p>
<b>Duration</b>	The duration for which agents will circulate. The duration is measured separately for each agent, relative to the time at which the agent was created.
<b>Time</b>	The time at which agents will stop circulating.
<b>Count</b>	The number of portals an agent will visit before considering the circulation finished.
<b>Wait After Count</b>	When the count and duration or end time are used together, this determines the agent behaviour when they finish the circulation count before the specified duration limit or end time. If true, agents will continue to wait at their last circulation portal. If false agents will ignore the incomplete duration or end time and proceed to their destination.
<b>Circulation Portals</b>	The <a href="#">portals</a> agents should circulate between. Portals can be weighted to adjust their chances of being chosen when agents are determining the next leg in their circulation. See <a href="#">choosing objects</a> for more details.

Dwell Tab	
<b>Wait at Start</b>	Whether agents should wait at their origin portal before starting to circulate.
<b>Wait Style</b>	Sets the waiting behaviour of agents while at a circulation portal. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Dwell Duration</b>	When an agent reaches a circulation portal, the agent will wait or dwell at that portal for a period of time. The time is randomly generated according to a distribution as specified by the dwell duration rules. Agents will search through the rules from top to bottom and use the distribution corresponding to the first occurrence of their circulation portal (either directly or as a member of a specified collection). If the circulation portal is not contained within the dwell rules, the default distribution is used.

Colours Tab	
<b>Colours</b>	<p><b>Event colour:</b> Agents are assigned the same colour as the event object.</p> <p><b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour.</p> <p><b>Darken event colour:</b> Agents are assigned a darker version of the event object colour.</p> <p><b>Rule based:</b> Agents are assigned a colour according to the colouring rules. Working from top to bottom, agents will use the first colour for which the test evaluates to true.</p>

	<b>Specified colour:</b> Agents are assigned the specified colour.
--	--

Actions Tab	
<b>On birth</b>	The action will be applied to all agents created by the event as they enter the simulation. Any tasks given by the action will be executed before the agents begin circulating. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

### Collections in circulate events

[Collections](#) can be used in the "Origins", "Destination" and "Circulation Portals" properties. The collections can be weighted, changing the distribution of agents going from/to various portals.

#### 4.3.3.5.3 Vehicle

Vehicle events are used to simulate the controlled arrival and departure of agents at periodic intervals. The event is designed around the idea of a train or bus. It provides options for specifying arrival times, creating boarding agents before each arrival, creating alighting/through passengers at each arrival, opening gated links while the vehicle is in station, and controlling access to the vehicle based on a fixed capacity.

#### Arrival Times

Each vehicle event can have multiple arrival times. These times can be episodic through the calculation of random headway intervals or specified explicitly through the use of a schedule.

#### Alighting / Boarding

Alighting agents are created at or just before each vehicle arrival, while boarding agents are typically created over a longer period but are still specified per arrival. Both are optional. Generally the origins for the alighting agents will be the destinations for the boarding agents, but this does not have to be the case.

#### Through Passengers

Through passengers are agents that arrive and depart with each vehicle arrival. They arrive and depart at the same vehicle origin. They can be created virtually for the purposes of capacity calculations, or 'In Scene' as standard agents that appear with the alighting agents and exit when the vehicle departs. Virtual agents have no impact on simulation performance. Through passengers can be specified as a single count which is then distributed across the entire vehicle, or as a function of alighting agents and capacity so as to achieve a target occupancy level per car.

#### Access

If capacity is enabled, boarding agents will be prevented from accessing the car floors when the population of the car floors reaches capacity. This will only work when access to the car floors is via the links chosen as car doors. Capacity calculations are on a per floor basis.

Use of open gates can be restricted to those agents created by the vehicle event or those agents that pass a test. In this case car floors must be specified and connected to the gated car links in order to determine the boarding and alighting directions.

#### Alternative Uses

Not all features of the vehicle event need to be used at once, nor does the event need to be used specifically for vehicles. It can be used simply to open gates at irregular intervals, or to create bursts of agents at controllable intervals.

**Properties**

General Tab	
<b>Timing: Start</b>	The time at which the event will start. If using a periodic arrival type, this will coincide with the first vehicle arrival. If using a scheduled arrival time, this time represents the 00:00:00 time within the schedule.
<b>Timing: Dwell Time</b>	How long vehicles dwell with the vehicle doors open. This is can be specified as a <a href="#">distribution</a> .  If dwell time exceeds headway time, the vehicle doors will remain open until the last vehicle leaves.
<b>Repeat: Type</b>	<b>Periodic:</b> The first arrival will be at the event start. The next arrival will be after first the dwell interval and then the headway interval have elapsed.  <b>Periodic with lookup:</b> Arrival times are periodic, but population counts are determined by looking each arrival time up in a table of time intervals, and using the population values specified for that interval. Time intervals are relative to the event start time.  <b>Scheduled:</b> Arrival times are set explicitly through a table. Scheduled times are relative to the event start time.
<b>Repeat: Count</b>	The number of vehicle arrivals (available only with periodic types).
<b>Repeat: Headway</b>	The time between vehicle arrivals. This can be specified as a <a href="#">distribution</a> (available only with periodic types).
<b>Repeat: Alighter counts</b>	Specify alighting agent counts in the arrival schedule (available only with table based types).
<b>Repeat: Through passenger counts</b>	Specify through passenger counts in the arrival schedule (available only with table based types).

Access Tab	
<b>Car Doors</b>	Gated links and collections of gated links representing the vehicle doors. These doors are opened when the vehicle arrives and are kept open for as long as the vehicle dwells. This property is optional.
<b>Boarding Access</b>	Determine the agents that will be able to use the gate in the boarding direction. Only these agents will perceive the gate as open while the remaining agents will continue to see it as closed. The boarding direction is the direction leading onto a 'Car Floor'.  <b>Open to all:</b> Any agent can use the gate.  <b>Open to boarders:</b> Only agents created by this event can use the gate.  <b>Open if test true:</b> Only agents for which the test is true can use the gate.
<b>Alighting Access</b>	Determine the agents that will be able to use the gate in the alighting

	<p>direction. Only these agents will perceive the gate as open while the remaining agents will continue to see it as closed. The alighting direction is the direction leading off of a 'Car Floor'.</p> <p><b>Open to all:</b> Any agent can use the gate.</p> <p><b>Open to alighters:</b> Only agents created by this event can use the gate.</p> <p><b>Open if test true:</b> Only agents for which the test is true can use the gate.</p>
<b>Enforce capacity limits on car floors</b>	Control access through the car doors onto the car floors based on the set capacity. The capacity control applies to agents in the boarding direction.
<b>Max Occupancy per Car</b>	The maximum number of passengers allowed on a single car floor.
<b>Car Floors</b>	Each floor included directly or through a collection will be assumed to have the specified capacity. Car doors leading onto a car floor will be closed when the capacity is reached.

<b>Alighting Tab</b>	
<b>Create alighting agents</b>	Enable the creation of alighting agents.
<b>Timing: Before arrival</b>	The time before vehicle arrival that alighting agents are created.
<b>Alighters: Profile</b>	The <a href="#">profile</a> used to create alighting agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to alter the distribution of profiles amongst the population.
<b>Alighters: Demand</b>	<p>The number of alighting agents created each time a vehicle arrives. Agents will be created a few seconds before the vehicle arrives and doors open to give them time to marshal in front of the vehicle doors.</p> <p><b>Random:</b> The number of alighting agents to create for each arrival is determined based on the distribution.</p> <p><b>Lookup Table:</b> The number of alighting agents is determined by comparing each arrival time to a table of time intervals and using the population value for that interval.</p> <p><b>Scheduled:</b> The number of alighting agents is taken from the arrival schedule.</p>
<b>Vehicle Origins</b>	The <a href="#">portals</a> at which alighting agents will be created. Portals can be weighted to alter the distribution of agents across the set. See <a href="#">Choosing Objects</a> for more details.
<b>Vehicle Destinations</b>	The <a href="#">portals</a> from which the alighting agent will be given its initial goal.

	<p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal which can be reached using the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p>
--	---

Boarding Tab	
<b>Create boarding agents</b>	Enable the creation of boarding agents.
<b>Timing: Before arrival</b>	The time before vehicle arrival that boarding agents start being created.
<b>Population: Profile</b>	The <a href="#">profile</a> used to create boarding agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to alter the distribution of profiles amongst the population.
<b>Population: Source</b>	<p>Describes different methods for determining the number of agents to create. When agents are specified per origin or destination, collections are treated as a single entry with the count distributed over members according to the collection weights.</p> <p><b>Count total:</b> Specify a single count of agents</p> <p><b>Count per origin:</b> Specify the number of agents to create at each origin portal (or collection).</p> <p><b>Count per destination:</b> Specify the number of agents that will be assigned to each destination portal (or collection).</p> <p><b>Origin table:</b> Specify a custom count of agents for each origin portal (or collection).</p> <p><b>Destination table:</b> Specify a custom count of agents to be assigned each destination portal (or collection).</p> <p><b>Origin destination matrix:</b> Specify custom counts for each origin/destination pair.</p> <p><b>Schedule:</b> Specify a sequence of interval/population pairs. The first interval begins when the event becomes active.</p>
<b>Population: Demand</b>	<p>The number of agents the event will create and when they will be created (relative to the start of boarding arrivals as specified by the 'Before arrival' and vehicle arrival times). If demand duration exceeds headway time, separate distributions of boarding agents will overlap.</p> <p><b>Evenly spaced:</b> The specified number of agents will arrive at a constant rate over the duration.</p> <p><b>Instant:</b> The specified number of agents will arrive all at once at the 'Before arrival' time.</p> <p><b>Random:</b> The specified number of agents will arrive according to the specified distribution over the duration. See <a href="#">agent start distributions</a> for information on arrival distributions.</p>

	<p><b>Table:</b> A custom distribution can be described by a series of intervals and weights. Each row contains the duration of an interval and the fraction of the total agents to create over that interval. The first interval starts at the 'Before arrival' time. Subsequent intervals begin when the previous interval ends. Agents are created according to a uniform distribution within each interval. Fractional values are automatically normalized.</p>
<b>Vehicle Origins</b>	The <a href="#">portals</a> at which boarding agents will be created. Portals can be weighted to alter the distribution of agents across the set. See <a href="#">choosing objects</a> for more details.
<b>Vehicle Destinations</b>	<p>The <a href="#">portals</a> from which the boarding agent will be given its initial goal.</p> <p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal which can be reached using the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p>

Through Tab	
<b>Create through passengers</b>	Enable the creation of through passenger agents. These agents arrive with the alighting agents but remain on the vehicle and depart when the vehicle departs.
<b>Type</b>	<p><b>Virtual:</b> Through passenger numbers are calculated and used to determine car capacity, but no agents are created in the scene.</p> <p><b>In Scene:</b> Through passengers are created as standard agents. They arrive with the alighting agents, wait in place while the vehicle is in the station, then exit the simulation when the vehicle departs.</p>
<b>Before Arrival</b>	The time before vehicle arrival that through agents are created.
<b>Arrival Population: Profile</b>	The <a href="#">profile</a> used to create through passenger agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to alter the distribution of profiles amongst the population.
<b>Arrival Population: Type</b>	<p><b>Calculated to achieve car occupancy:</b> Specify a target occupancy level. The number of through passengers created per car will be calculated as the car's target occupancy minus the number of alighters in that car. This is only available when access capacity is enabled.</p> <p><b>Specified across vehicle:</b> Create a single count of agents per arrival and distribute the agents across the vehicle origin portals.</p>
<b>Cars Occupancy</b>	<b>At full capacity:</b> Each car arrives at full capacity. The number of through passengers in a car will be equal to the capacity minus the number of alighters leaving that car.

	<p><b>At percent capacity:</b> Each car arrives at some percentage of capacity. The number of through passengers in a car will be equal to the total occupancy minus the number of alighters leaving that car.</p>
<b>Vehicle Demand</b>	<p><b>Random:</b> Generate a random count using the specified distribution.</p> <p><b>Lookup Table:</b> The population is determined by comparing each arrival time to a table of time intervals and using the population value for that interval.</p> <p><b>Scheduled:</b> The population is taken from the arrival schedule.</p>
<b>Vehicle Origins</b>	<p>The <a href="#">portals</a> through which passenger agents will be created. When the population type is set to distribute a single count across the vehicle, portals can be weighted to alter the distribution of agents across the set. When using a capacity based demand, the portals are not used for capacity calculations but only to place agents in the scene.</p>

Colours Tab	
<b>Colours</b>	<p>Defines the colour scheme for boarding/alighting/through agents.</p> <p><b>Event colour:</b> Agents are assigned the same colour as the event object.</p> <p><b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour.</p> <p><b>Darken event colour:</b> Agents are assigned a darker version of the event object colour.</p> <p><b>Rule based:</b> Agents are assigned a colour according to the colouring rules. Working from top to bottom, agents will use the first colour for which the condition evaluates to true.</p> <p><b>Specified colour:</b> Agents are assigned the specified colour.</p>

Actions Tab	
<b>On birth</b>	<p>The action will be applied to all boarding/alighting/through agents created by the event as they enter the simulation. Any tasks generated by the action will be executed before the agent seeks its assigned destination. For a description of the order in which actions are applied see <a href="#">action order of execution</a>.</p>

**Collections in vehicle events**

[Collections](#) can be used in the "Vehicle Doors" property. All member links with gates enabled will be opened as if they were directly used.

Collections can also be used in "Origins" and "Destination" properties for boarding, alighting, and through agents. The collections can be weighted, changing the distribution of agents going from/to various portals.



## 4.3.3.5.4 Evacuate

Evacuate events simulate the evacuation of agents from the scene. Agents created by an evacuate event will first wait a specified amount of time, then attempt to exit the simulation through the best of the destination portals.

If [zones](#) have been specified, agents will attempt to evacuate each zone in order before heading to the destination [portals](#). Once a zone has been evacuated, agents may re-enter previously evacuated zones, so generally, subsequent zones should contain prior zones. Agents created outside evacuation zones will simply head to their destination portals.

Only agents created by this event will evacuate the given zones or head to the destination portals.

**Active**

An evacuate event is active over the interval for which it can create agents. This will depend on the population demand type. When using 'Instant' the event is active only for an instant. When using 'Random' or 'Evenly spaced', the event is active for the duration of the specified interval, including those frames when no agents are created. When using a schedule or weighted curve, the event is active from when the start condition is met until the final interval in the table is complete. Once the event has finished creating agents it is no longer active, even if those agents have not yet completed their evacuation.

**Properties**

Agents Tab	
<b>Timing: Start Time</b>	The condition for starting the event. See <a href="#">triggering events</a> for more information.
<b>Population : Profile</b>	The <a href="#">profile</a> used to create agents. If a collection is specified, each agent will be assigned one of the profiles from the collection. A weighted collection can be used to assign a distribution of profiles amongst the population.
<b>Population : Source</b>	<p>Describes different methods for determining the number of agents to create. When agents are specified per origin or destination, collections are treated as a single entry with the count distributed over members according to the collection weights.</p> <p><b>Count total:</b> Specify a single count of agents  <b>Count per origin:</b> Specify the number of agents to create at each origin portal (or collection).  <b>Count per destination:</b> Specify the number of agents that will be assigned to each destination portal (or collection).  <b>Origin table:</b> Specify a custom count of agents for each origin portal (or collection).  <b>Destination table:</b> Specify a custom count of agents to be assigned each destination portal (or collection).  <b>Origin destination matrix:</b> Specify custom counts for each origin/destination pair.  <b>Schedule:</b> Specify a sequence of interval/population pairs. The first interval begins when the event becomes active.</p>
<b>Population : Demand</b>	<p>The number of agents the event will create and when they will be created relative to the event start time.</p> <p><b>Evenly spaced:</b> The specified number of agents will arrive at a constant rate over the duration.  <b>Instant:</b> The specified number of agents will arrive all at once at the event start</p>

	<p>time.</p> <p><b>Random:</b> The specified number of agents will arrive according to the specified distribution over the duration. See <a href="#">duration distributions</a> for information on arrival distributions.</p> <p><b>Table:</b> A custom distribution can be described by a series of intervals and weights. Each row contains the duration of an interval and the fraction of the total agents to create over that interval. The first interval starts at the event start. Subsequent intervals begin when the previous interval ends. Agents are created according to a uniform distribution within each interval. Fractional values are automatically normalized.</p>
<b>Origins</b>	The <a href="#">portals</a> at which agents will be created. Portals can be weighted to alter the distribution of agents across the set. See <a href="#">choosing objects</a> for more details.
<b>Destinations</b>	<p>The <a href="#">portals</a> from which the agent will be given its initial goal. The agent will pursue its initial goal once it has finished with the pre-movement wait and evacuated any specified zones.</p> <p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal which can be reached using the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p>

Evacuation Tab	
<b>Timing: Pre-movement wait</b>	How long agents will wait before beginning to evacuate. This can be set as a distribution in seconds, see <a href="#">single value distributions</a> for more information.
<b>Timing: Wait style</b>	Sets the behaviour of agents during the pre-movement wait. This can include use of <a href="#">wait spaces</a> . See <a href="#">wait style</a> for a description of the available behaviours.
<b>Zones</b>	The <a href="#">zones</a> in order of evacuation. Agents will evacuate zones in order, but are not prevented from re-entering previously evacuated zones.
<b>Exits: Clear Route History</b>	Agents are biased against backtracking across objects they have already traversed. If unchecked, agents that have evacuated all zones will be biased against re-tracing their steps when seeking the exit portals. If route history is cleared, agents forget where they have been and will have no problem backtracking if that is the best way to reach their exit portals.

Colours Tab	
<b>Colours</b>	<p><b>Event colour:</b> Agents are assigned the same colour as the event object.</p> <p><b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour.</p> <p><b>Darken event colour:</b> Agents are assigned a darker version of the event object colour.</p> <p><b>Rule based:</b> Agents are assigned a colour according to the colouring rules.</p>

	Working from top to bottom, agents will use the first colour for which the condition evaluates to true. <b>Specified colour:</b> Agents are assigned the specified colour.
--	---

Actions Tab	
<b>On birth</b>	The action will be applied to all agents created by the event as they enter the simulation. Any tasks given by the action will be executed before the agent begins the pre-movement wait or any evacuation. For a description of the order in which actions are applied see <a href="#">action order of execution</a> .

### Collections in open gate events

[Collections](#) can be used in the "Origins" and "Destinations" properties. Collections can be weighted, changing the distribution of agents created at those portals.

#### 4.3.3.5.5 Timetable

Timetable objects allow for the rapid and potentially automated creation of large numbers of agents and coordinated events. They are suitable for modeling train schedules, flight schedules, bus schedules, university lectures, intersection gate timings, or any number of additional scenarios.

Timetables allow for the batch import of related agent schedules and/or events. Timetables are driven by a series of comma separated (csv) text files. Once imported, the raw data is embedded in the timetable object and will be saved as part of the scene. The raw data is only processed on validation or simulation so there is little overhead associated with the amount of data or the number of entries.

Timetable files can be generated by hand, but will more often be generated by user written scripts or excel macros.

Different input files control different aspects of the timetable. One file specifies the creation of agents while another controls gate open events. It is possible to define time and location dependencies between the different files (so that a gate opens and agents are created at the same time) through the use of internal reference events. Reference events are not visible to scene elements outside of the timetable. Agents created with respect to a particular reference event are forever associated with that reference identifier and can be later targeted by other timetable events. It is also possible to create a very simple timetable that only executes gate events or only controls agent scheduling.

When inspecting a timetable object, the buttons at the top of the window allow for batch import/export/reload of the various timetable files. Each file type has a default name constructed from the name of the timetable and the file type. Default names are listed in the 'File Types' table below.

For information on each of the supported timetable files, see [timetable files](#).

Menu Bar Commands	
<b>File: Import Timetable</b>	Select a folder and import all contained files with the expected default names (see File Types below).
<b>File: Export Timetable</b>	Select a folder and export all file types that contain data. If file names are not

	specified the default file names are used.
<b>File: Export Empty Timetable</b>	Select a folder and export empty versions of all files using the default names (see File Types below).
<b>Reload</b>	Attempt to reload data from each file specified in the window. Entries where no file is specified will be ignored.

<b>General Properties</b>	
<b>Base Path</b>	This is the default location for input files managed by the timetable. If any specific file names are relative they are assumed to be in this path.
<b>Profile</b>	The default profile to use for agents created by the agent schedule file when no profile is specified in the file.

4.3.3.5.5.1 Timetable Files

Timetable files are all field separated (csv) text files. The separation character is by default a comma but can be set through the [application preferences](#). Blank lines in the file are ignored, as are leading and trailing spaces. Any line beginning with a '#' is considered a comment and is skipped.

<b>File Types</b>	
<b>Reference Event</b>	<p>Import, export, or clear the embedded reference event data. A reference event entry specifies a time, duration, and location. These events can be used by other sections of the timetable such as agent schedules, gate events, evacuation events, or action events.</p> <p>Default name: TimetableReferenceEvent.csv.</p> <p>See <a href="#">Timetable Reference Event File</a> for more information.</p>
<b>Schedule</b>	<p>Import, export, or clear the embedded agent schedule data. Each schedule entry creates a specified number of agents with particular origins and destinations.</p> <p>Default name: TimetableSchedule.csv.</p> <p>See <a href="#">Timetable Schedule File</a> for more information.</p>
<b>Location</b>	<p>Import, export, or clear the embedded location data. Each location entry defines a collection of portals with corresponding distributions or membership weights. Locations can be used by reference events, agent schedules, or evacuation events.</p>

	<p>Default name: TimetableLocation.csv.</p> <p>See <a href="#">Timetable Location File</a> for more information.</p>
<b>Curve</b>	<p>Import, export, or clear the embedded curve data. Each curve entry defines an arrival profile or distribution that can be used by the agent schedules.</p> <p>Default name: TimetableCurve.csv.</p> <p>See <a href="#">Timetable Curve File</a> for more information.</p>
<b>Gate</b>	<p>Import, export, or clear the embedded gate data. Each gate entry defines a collection of gate objects and can be referenced by gate events.</p> <p>Default name: TimetableGate.csv.</p> <p>See <a href="#">Timetable Gate File</a> for more information.</p>
<b>Gate Event</b>	<p>Import, export, or clear the embedded gate event data. Each gate event entry will open a collection of gates for a specified interval. The gate event can be used on its own or in combination with the reference event.</p> <p>Default name: TimetableGateEvent.csv.</p> <p>See <a href="#">Timetable Gate Event File</a> for more information.</p>
<b>Action Event</b>	<p>Import, export, or clear the embedded action event data. Each action event entry will apply the specified action to agents over the specified interval.</p> <p>Default name: TimetableActionEvent.csv.</p> <p>See <a href="#">Timetable Action Event File</a> for more information.</p>
<b>Evacuation Events</b>	<p>Import, export, or clear the embedded evacuation event data. Each evacuation event entry will broadcast a single evacuation over the specified interval.</p> <p>Default name: TimetableEvacuationEvent.csv.</p> <p>See <a href="#">Timetable Evacuation Event File</a> for more information.</p>

The timetable reference event file is a comma separated (csv) text file defining one or more reference events. Each row corresponds to a single reference event. These events can be referenced by other input files within the same timetable. Reference events defined within the timetable may not be referenced from outside of the timetable. Reference event names must be unique and cannot be the same as any other objects within the project.

Reference events can refer to anything from an airplane departure to a train arrival to a university lecture. The end time of the event is calculated as Start Time + Duration.

Column Headers	
<b>Reference Event</b>	The name of a reference event. The name must be unique both within the

<b>Name</b>	file and across the entire MassMotion project. Other files within the timetable can refer to a reference event by name.
<b>Start Time</b>	The time associated with the reference event. The value must be either a single number indicating seconds, or a string of the form hh:mm:ss.
<b>Duration (optional)</b>	The duration from arrival to departure. The value must be either a single number indicating seconds, or a string of the form hh:mm:ss. If no value is specified a duration of 0 is assumed.
<b>Location</b>	Either the name of a location group (see <a href="#">Timetable Location File</a> ) or the name of a portal or collection of portals in the project.
<b>Init Action (optional)</b>	The name of an existing action in the project. The action will be applied to any agent created by a schedule that refers to this reference event. The action is applied after the agent has been given any optional tokens from subsequent columns.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token or collection of tokens by name. The token or token collection must exist in the project. Agents created by a schedule that refers to this event will be given the specified tokens. The tokens are given before the Init Action is applied.

### Example

See [Timetable Reference Event Example](#).

The timetable schedule file is a comma separated (csv) text file defining a number of schedules. Each row corresponds to a single schedule and will create a population of agents. The schedule can optionally refer to events in the reference event file or be used independently.

If the From or To fields reference locations, agents are created and sent to those locations. If the From or To fields refer to reference events, the agents are created and sent to the locations defined by those events.

The start time for a schedule is determined based on whether or not reference events were specified in the From or To fields:

- **From location, To location:** The start time is assumed to be midnight on the day at which the simulation begins (so the time offset effectively becomes the start time).
- **From reference event, To location:** The start time is taken from the From reference event start time.
- **From location, To reference event:** The start time is taken from the To reference event start time.
- **From reference event, To reference event:** The start time is taken from the From reference event start time.

Column Headers	
<b>From</b>	The location at which agents should be created. The location can be either the name of an existing portal in the project, the name of a collection of portals in the project, the name of a location from the <a href="#">Timetable Location File</a> or the name of a reference event from the <a href="#">Timetable Reference Event File</a> . If the location is from the location file or a project collection agents will be distributed across the member portals according to the associated distribution. If the location is the name of a reference event, the event location is used.
<b>To</b>	The goal location given to agents created by the schedule. The location can be either the name of an existing portal in the project, the name of a collection of portals in the project, the name of a location from <a href="#">Timetable Location File</a> , or the name of a reference event from the <a href="#">Timetable Reference Event File</a> . If the location is from the location file agents will be either sent to the portals according to the defined distribution, or told to seek the closest portal member depending on the settings in the location file. If the location is a project collection agents will be sent to the portals according to the collection weights. If the location is the name of a reference event, the event location is used.
<b>Population</b>	The number of agents created by the schedule.
<b>Time Offset (optional)</b>	A time value added to the schedule start time. The value must be either a single number indicating seconds, or a string of the form hh:mm:ss. If a reference event has been specified, this time is added to the reference event start time. If no reference event has been specified, this time is taken as the schedule start time, measured relative to midnight (00:00:00). See above for a full description of how the schedule start time is determined.
<b>Curve (optional)</b>	The name of an entry in the <a href="#">Timetable Curve File</a> curve file. The specified population of agents will be created according to the curve distribution. If no curve is specified, all agents are created over 1 second.
<b>Avatar or Colour (optional)</b>	The name of an existing avatar in the the project or a valid colour name. If the text matches the name of an avatar, then agents created by the schedule will be given that avatar. If the text does not match the name of an avatar but corresponds to a recognized colour, then agents will be created with the default avatar and given the specified colour. If the entry is blank the default avatar is used and the agent is coloured grey.  Colors: aqua, blue, green, orange, red, yellow, black, white, grey, lightgrey, darkgrey, etc.. See <a href="#">Timetable Colours</a> for a full list of supported colours.
<b>Profile (optional)</b>	The name of an existing profile or collection of profiles in the project. If a collection is specified each agent is given one of the profiles chosen randomly according to the collection weighting. If no profile is specified, agents will be given the timetable's default profile.
<b>Init Action</b>	The name of an existing action in the project. The action will be applied to

<b>(optional)</b>	any agent created by the schedule. The action is applied after the agent is given any optional tokens from subsequent columns.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token or collection of tokens by name. The token or token collection must exist in the project. Agents created by the schedule entry will be given these tokens. The tokens are given before the Init Action is applied.

**Example**

See [Timetable Schedule Example](#) and [Timetable Reference Event Example](#).

The following colour names can be used in the [Timetable Schedule File](#).

Agent Colours		
amber	FFBF00	
aqua	00FFFF	
azure	007FFF	
beige	F5F5DC	
black	000000	
blue	0000FF	
brown	8B4513	
chartreuse	7FFF00	
darkblue	00007F	
darkgray	404040	
darkgreen	007F00	
darkred	7F0000	
gray	7F7F7F	
green	00FF00	



ivory	FFFFFF0	
lightgray	BFBFBF	
magenta	FF00FF	
mint	BDFCC9	
navy	00007F	
olive	6B8E23	
orange	FF7F00	
pink	FFB5C5	
purple	9B30FF	
red	FF0000	
rose	FF007F	
salmon	FA8072	
teal	008080	
turquoise	40E0D0	
vermilion	E34234	
violet	7F00FF	
wheat	F5DEB3	
white	FFFFFFF	
yellow	FFFF00	

The timetable location file is a comma separated (csv) text file defining a group of portals. The header lists all portals available in the project. Each row corresponds to a different group, with non-zero values indicating that the given portal is to be included in the group. Fractional values are used to indicate percentage allocations within the group. The location group names must be unique both within the file and within the project.

Location groups can be referenced by the [Timetable Reference Event File](#), [Timetable Schedule File](#), or [Timetable Evacuation Event File](#).

Column Headers	
<b>Group Name</b>	The name of the location group. The name must be unique both within the file and across the entire MassMotion project.
<b>Use Closest Goal (optional)</b>	This determines whether agents using this location as their goal will be assigned a specific member portal based on the distribution (if false or blank), or given all portals and told to seek the closest (if true). Use Y or 1 to indicate true. This field is ignored when a location is used as an entrance in a schedule file.
<b>Portal or Collection Names... (variable)</b>	<p>Each location can indicate which portals are to be included in the group by specifying a non-zero value in the corresponding column. Values should be between 0 and 1. All values for a group should sum to 1. When a location group is specified but only a single portal is required (for example when determining where to create an agent) a portal is chosen from the group using the probability distribution defined by the non-zero fractional values.</p> <p>When collection names are used the location will include all portal members of the collection. Non portal members are ignored. Weights for individual portals are calculated by multiplying the collection member weight by the weight specified for the collection in the location file. When a collection does not specify weights, portal members are considered equally weighted within the collection. If the same portal appears multiple times, its weights are summed.</p>

### Example

```

Assume the following collections defined in the project:
1. CollNoWeight: Portal_A, Portal_B
1. CollManualWeight: Portal_A (0.25), Portal_B (0.75)

#
# Sample MyTimetableLocation.csv
#
Group Use ClPortal_Portal_Portal_Portal_CollNoWeight          CollManualWeight

# create a group with A and B and equal entrance and exit distribution
GroupA,          Y,          Y

# create a group with A and B where agents are 3 times more likely to use A than B
GroupA,          0.75, 0.25
    
```

```

# create a group with equal entrance distribution over A, B, D, where when exiting
GroupAY, Y, Y, 0, Y

# create a group with C and D where as entrances agents are twice as likely to pick
GroupCY, , , 0.6666,0.3333

# create a group with members and weightings Portal_A (0.6*0.5=0.3), Portal_B (0.6*
ColEx1 0.4 0.6

# create a group with members and weightings Portal_A (0.8*0.25=0.2), Portal_B (0.8
ColEx2 0.2 0.8

```

Also, see [Timetable Schedule Example](#).

The timetable curve file is a comma separated (csv) text file defining different arrival profiles for use by the timetable schedule. Each row corresponds to a single curve distribution and can be referenced from [Timetable Schedule File](#) by name.

Column Headers	
<b>Curve Name</b>	The name of the curve.
<b>Interval Duration</b>	The length of each step in the curve. The value must be a single number indicating seconds, a single number with a 'h', 'm', or 's' designator, or a string of the form hh:mm:ss. Agents created inside each interval are distributed randomly within the interval.
<b>Values... (variable)</b>	Each subsequent column specifies a single value indicating the percentage of the total population that is to be created within that interval. Use a single value of 1 to indicate that all agents should be created within a single interval. Values of 0.25, 0.5, 0.25 would indicate that a quarter of the agents should be created over the first interval, half over the second, and the remaining quarter over the third interval. Values in a curve must sum to 1.

### Example

```

#
# Sample MyTimetableCurve.csv
#
Curve Name, Interval Duration, Values..

# all agents within a single second
CBurst1, 1, 1

# uniformly random over 10 seconds

```

```

CConstant10,      10,      1

# uniformly random over 120 seconds
CConstant120,    00:02:00,  1

# tapered over 60 seconds
CTapered60,     10,      0.5,  0.3,  0.1,  0.05,  0.025,  0.025

# tapered over 120 seconds
CTapered120,    20,      0.5,  0.3,  0.1,  0.05,  0.025,  0.025

# custom rates over 1 hour
CCustom1H,      0.2h,    0.2,  0.01,  0.45,  0.15,  0.29
    
```

Also, see [Timetable Schedule Example](#).

The timetable gate file is a comma separated (csv) text file defining a group of gates. The header lists all gates available in the project or collections which contain gates. Each row corresponds to a different group, with non-zero values indicating that the given gate is to be included in the group. The gate group names must be unique both within the file and within the project.

Gate groups can be referenced by the [Timetable Gate Event File](#).

Column Headers	
<b>Group Name</b>	The name of the gate group. The name must be unique both within the file and across the entire project.
<b>Gate or Collection Names... (variable)</b>	Each gate group can indicate which gates are to be included in the group by specifying a non-zero value in the corresponding column. If a collection name is specified, the group will contain all gates found in the collection.

### Example

See [Timetable Gates Example](#)

The timetable gate event file is a comma separated (csv) text file defining a number of gate events. Each row corresponds to a single event and will open one or more gates for a specified period of time. The events defined within the timetable have no connection to events defined elsewhere in the project.

The start time for an event depends on whether or not a reference event was specified. When using a reference event, the start time is taken from the reference event. If no reference event is specified, the start time is assumed to be midnight on the day at which the simulation begins (so the time offset effectively becomes the start time).

Column Headers	
<b>Reference Event Name (optional)</b>	The name of a reference event as defined in the <a href="#">Timetable Reference Event File</a> . The field can be left blank to indicate that the event is not associated

	with any reference event.
<b>Time Offset (optional)</b>	Combines with the event start time to define the time at which the event fires. If a reference event has been specified, this time is added to the reference event start time. If no reference event has been specified, this time is taken as the gate event start time, measured relative to midnight (00:00:00). The value must be either a single number indicating seconds, or a string of the form hh:mm:ss. If no value is specified a value of 0 is assumed.
<b>Duration (optional)</b>	The length of time the gate will remain open. If a reference event has been specified, this field can be left blank to indicate that the reference event duration should be used. The value must be either a single number indicating seconds, or a string of the form hh:mm:ss.
<b>Apply only to Reference Event</b>	If a reference event has been specified and a true value (Y or 1) is used, the gate will only open for those agents created by a schedule that uses the specified reference event. The gate will remain closed to all agents not arriving from or heading to the reference event.
<b>Key Token (optional)</b>	The name of an existing token in the project. If specified, the gate will only open for those agents holding the specified token.
<b>Gates... (variable)</b>	Each subsequent column can specify a gate or gate group to be opened by the event. Each entry must refer either to an existing gate object in the project, a project collection which contains gates, or to a gate group defined in the <a href="#">Timetable Gate File</a> .

### Example

See [Timetable Gates Example](#).

The timetable action event file is a comma separated (csv) text file defining a number of action events. Each row corresponds to a single event and will fire the specified action for a specified period of time. The events defined within the timetable have no connection to events defined elsewhere in the project.

The start time for an event depends on whether or not a reference event was specified. When using a reference event the start time is taken from the reference event. If no reference event is specified the start time is assumed to be midnight on the day at which the simulation begins (so the time offset effectively becomes the start time).

When the event fires, the action is applied to all agents in the scene that meet the criteria defined by the event definition. The event then remains active for the specified duration. If a zone is specified, the active event will be applied to all agents that enter the zone. If no zone is specified, the active event will apply to all agents entering the simulation. The event will not be applied to the same agent twice unless a zone is specified and the agent enters then leaves then re-enters the zone.

### Column Headers

<b>Reference Event Name (optional)</b>	The name of a reference event as defined in the <a href="#">Timetable Reference Event File</a> . The field can be left blank to indicate that the event is not associated with any reference event.
<b>Time Offset (optional)</b>	Combines with the event start time to define the time at which the event fires. If a reference event has been specified, this time is added to the reference event start time. If no reference event has been specified, this time is taken as the event start time measured from midnight (00:00:00). The value must be either a single number indicating seconds, or a string of the form hh:mm:ss. If no value is specified a value of 0 is assumed.
<b>Duration</b>	The length of time the event remains active. See above for a description of how the event is applied while active.
<b>Apply only to Reference Event</b>	If a reference event has been specified and a true value (Y or 1) is used, the action will only be applied to those agents created by a schedule that uses the specified reference event.
<b>Key Token (optional)</b>	The name of an existing token in the project. If specified, the action will only be applied to agents holding the specified token.
<b>Target Zone (optional)</b>	The name of an existing zone in the project. If specified, the action will only be applied to agents in the target zone. If new agents enter the zone while the event is active, the action is applied to those agents as they enter the zone. If no zone is specified, the event is applied to all agents in the scene regardless of location.
<b>Action</b>	The name of an existing action in the project. This action will be applied once to all agents currently in the scene that meet the event criteria, then subsequently to all agents that either enter the zone or scene while the event is active.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token or collection of tokens by name. The token or token collection must exist in the project. Each agent that is affected by the action event will also be given the specified tokens. Agents receive the tokens before the event action is applied.

The timetable evacuation event file is a comma separated (csv) text file defining a number of evacuation events. Each row corresponds to a single event and will fire for a specified period of time. The events defined within the timetable have no connection to events defined elsewhere in the project.

The start time for an event depends on whether or not a reference event was specified. When using a reference event the start time is taken from the reference event. If no reference event is specified, the start time is assumed to be midnight on the day at which the simulation begins (so the time offset effectively becomes the start time).

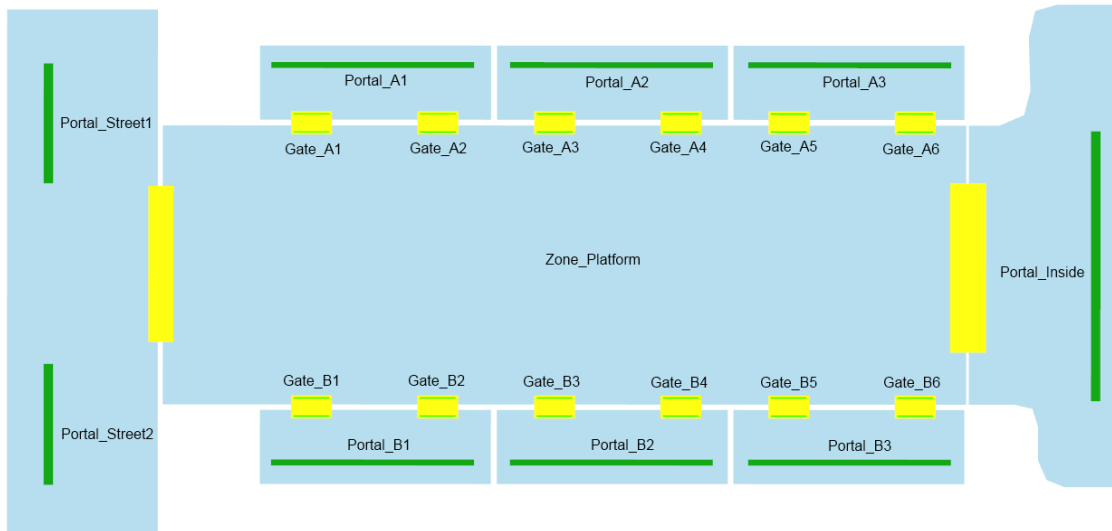
When the event fires, the evacuation command is given to all agents in the scene that meet the criteria defined by the event definition. The event then remains active for the specified duration. If a

zone is specified, the active event will be applied to all agents as they enter the zone. If no zone is specified, the active event will apply to all agents entering the simulation. The event will not be applied to the same agent twice unless a zone is specified and the agent enters then leaves then re-enters the zone.

Column Headers	
<b>Reference Event Name (optional)</b>	The name of a reference event as defined in the <a href="#">Timetable Reference Event File</a> . The field can be left blank to indicate that the event is not associated with any reference event. The reference event can be used to define either the event start time or evacuation location.
<b>Time Offset (optional)</b>	Combines with the event start time to define the time at which the event fires. If a reference event has been specified, this time is added to the reference event start time. If no reference event has been specified, this time is taken as the event start time measured from midnight (00:00:00). The value must be either a single number indicating seconds, or a string of the form hh:mm:ss. If no value is specified, a value of 0 is assumed.
<b>Duration</b>	The length of time the event remains active. See above for a description of how the event is applied while active.
<b>Apply only to Reference Event</b>	If a reference event has been specified and a true value (Y or 1) is used, the evacuation will only be applied to those agents created by a schedule that uses the specified reference event.
<b>Key Token (optional)</b>	The name of an existing token in the project. If specified, the evacuation will only be applied to agents holding the specified token.
<b>Target Zone (optional)</b>	The name of an existing zone in the project. If specified, the evacuation will only be applied to agents in the target zone. If new agents enter the zone while the event is active, the evacuation is applied to those agents as they enter the zone. If no zone is specified the event is applied to all agents in the scene regardless of location.
<b>Pre Movement Distribution (not used)</b>	Not yet supported.
<b>Location... (optional)</b>	The name of an existing portal in the project, the name of a collection of portals in the project, or the name of a location group from the <a href="#">Timetable Location File</a> . If left blank and a reference event has been specified, the location associated with the reference event is used. Agents will be told to seek the location. Once the agent reaches the location it will successfully exit the scene.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token or collection of tokens by name. The token or token collection must exist in the project. Each agent that is affected by the evacuation event will also be given the specified tokens.

4.3.3.5.5.2 Timetable Examples

All timetable examples assume the same general scene which can either be created from scratch using the steps below, or opened pre built from the TimetableExample.mm project in the Examples folder of the MassMotion installation directory.



1. Create a scene with the objects positioned and named as shown above.
2. Create an empty timetable object from the 'Activities' ribbon in the Main Window and name it 'MyTimetable'.
3. Open the MyTimetable properties and use the Export button to export 'Empty Timetable' files to a folder.
4. Edit the generated timetable files as appropriate (see the examples listed below).
5. Reload all file changes back into the timetable object using the 'Reload' button in the timetable's properties window.
6. The project is ready for simulation.

Examples:

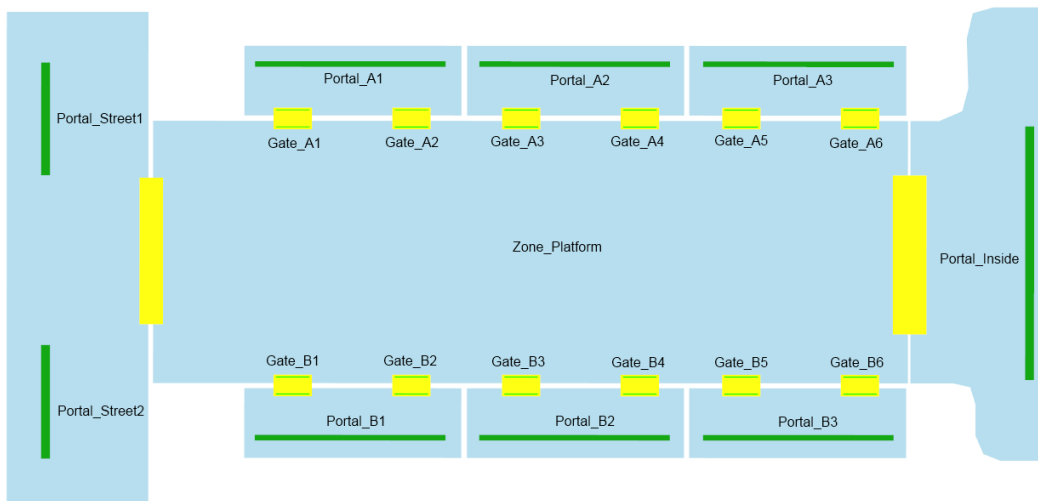
- [Timetable Schedule Example](#)
- [Timetable Gates Example](#)
- [Timetable Reference Event Example](#)

This example includes timetable files for generating a few simple streams of agents moving between the portals as shown below.

**Scene Setup**

Create a scene as described in [Timetable Examples](#) before attempting to edit the timetable files.





**Timetable Setup**

A simple curve file defines a few arrival distributions for use in the schedule file:

#							
#	Sample MyTimetableCurve.csv						
#							
Curve Name,	Interval	Durat	Values..				
#	uniformly random over 30 second						
CConstant30,	30,	1					
#	uniformly random over 120 second						
CConstant120,	00:02:00,	1					
#	tapered over 60 seconds						
CTapered60,	10,	0.5,	0.3,	0.1,	0.05,	0.025,	0.025
#	custom rates over 1 hour						
CCustom1H,	0.2h,	0.2,	0.01,	0.45,	0.15,	0.29	

A simple location file defines groupings of portals for easy reference in the schedule file:

#							
#	Sample MyTimetableLocation.csv						
#							
Group Name	Use Closest	(Portal_A1,	Portal_Portal_Portal_Portal_Portal_Portal_Portal				
#	Track A has uneven arrival distribution of agents across the cars, but departing						

GTrackA,	Y,	0.20,	0.40,	0.40					
# Track B has even arrival distribution and departing agents are even									
GTrackB,	,	,	,	,	Y,	Y,	Y		
# A simple group to to include both streets									
GStreet,	,	,	,	,	,	,	,	Y,	Y

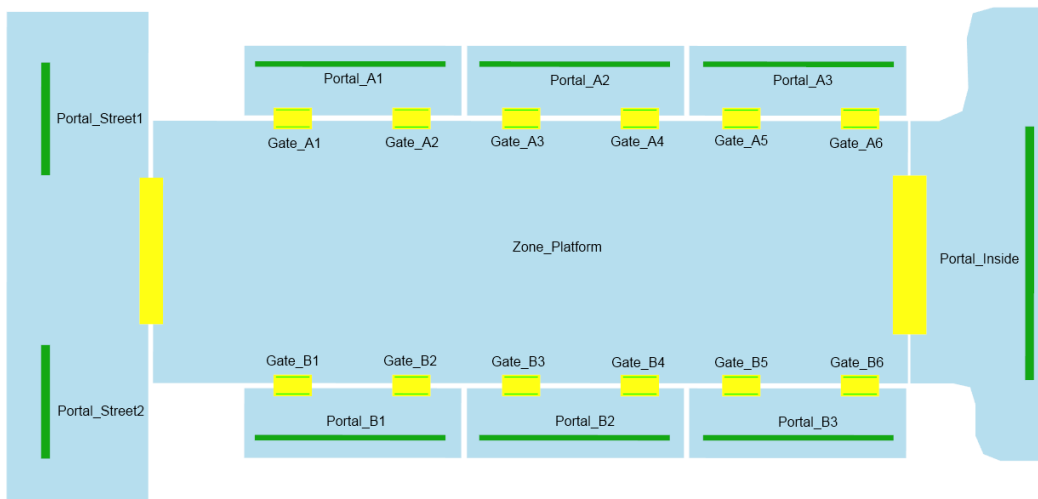
And finally the schedule file for generating agents

#									
# Sample MyTimetableSchedule.csv									
#									
From,	To,	Populat	Time Off	Curve,	Avatar or CProfi	Init Act	Give Tokens...		
# 100 agents at 00:15:00 over 1 hour, from either street portal to Portal_I									
GStreet,	Portal_Inside	100,	00:15:00	CCustom	Red				
# 45 and 23 agents at 00:15:00 over 1 hour, from Portal_Inside to t									
Portal_I	Portal_Street	45,	15m,	CCustom	Blue				
Portal_I	Portal_Street	23,	15m,	CCustom	Blue				
# 60 agents at 00:20:00 over 1 second from portals on Track A to the street, initial									
GTrackA,	GStreet,	60,	00:20:00		Green,		InitArri		
# 100 agents at 27:30:00 over 120 seconds, from TrackA portals to TrackB portals, g									
GTrackA,	GTrackB,	100,	27:30:00	CConsta	Red,	,	,	TNextDay	Ttransfe

This example includes timetable files for opening and closing a series of gates. This example could be used in combination with other timetable files, regular agent schedules, or regular gate events to achieve the desired result.

**Scene Setup**

Create a scene as described in [Timetable Examples](#) before attempting to edit the timetable files.



**Timetable Setup**

A simple gate file defines groupings of links for easy reference in the gate event file:

#														
#	Sample MyTimetableGate.csv													
#														
Group Name	Gate_A1,	Gate_A2,	Gate_A3,	Gate_A4,	Gate_A5,	Gate_A6,	Gate_B1,	Gate_B2,	Gate_B3,	Gate_B4,	Gate_B5,	Gate_B6,		
#	Combine all gates for track A into a single group for easy reference													
Gates_A,	Y,	Y,	Y,	Y,	Y,	Y,								
#	Track B has even arrival distribution and departing													
Gates_B,	,	,	,	,	,	,	Y,	Y,	Y,	Y,	Y,	Y,	Y,	Y,

And finally the gate event file for opening and closing gates.

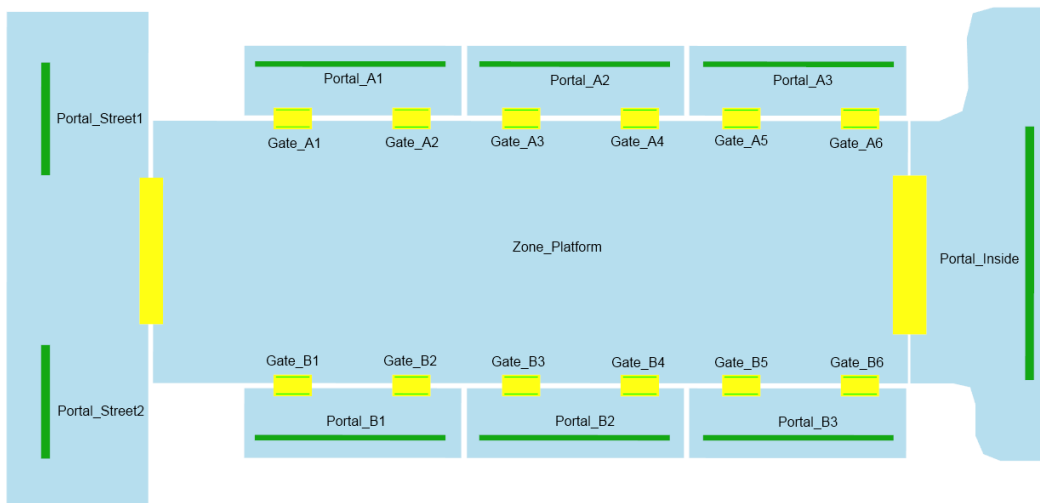
#														
#	Sample MyTimetableGateEvent.csv													
#														
Reference	Time Offset,	Duration,	Apply Only	Key Token,	Gate									
#	Open all gates in the group Gates_A at 00:30:00 for 1 minute, for all agents													
,	00:30:00,	1m,	,	,	Gates_A									
#	Open all gates in the group Gates_B at 00:35:00 for 30 seconds, only for agents h													
,	00:35:00,	30,	,	TArriving,	Gates_B									
#	Open Gate_B4 at 32:30:00 for 1 hour, for all agents													

,	32:30:00,	01:00:00,	,	,	Gate_B4		
---	-----------	-----------	---	---	---------	--	--

This example includes timetable files for simulating a simple train platform using reference events. A continuous stream of departing agents will move from the street and inside portals to both of the platforms. Train arrivals will be represented by reference events. When a train arrives on track A or B, a group of arriving agents will be generated and the corresponding gates opened to allow passage for boarding and alighting passengers. In addition to the train arrivals, a reference event will be used to represent a street festival to which some agents will be

**Scene Setup**

Create a scene as described in [Timetable Examples](#) before attempting to edit the timetable files. In addition to the general steps outlined above, create tokens TTrainA, TTrainB



**Timetable Setup**

A simple curve file defines a few arrival distributions for use in the schedule file:

```
#
# Sample MyTimetableCurve.csv
#
Curve Name,          Interval DuratiValues..

# uniformly random over 1 hour
CConstant1H,        01:00:00,      1
```

A simple location file defines groupings of portals for easy reference in the schedule file:

#									
#	Sample	MyTimetableLocation.csv							
#									
Group Name	Use	Closest	(Portal_A1,	Portal_	Portal_	Portal_	Portal_	Portal_	Portal_

# Track A has uneven arrival distribution of agents across the cars, but departing										
GTrackA,	Y,		0.20,		0.40,		0.40			
# Track B has even arrival distribution and departing agents are even										
GTrackB,	,		,		,		,	Y,	Y,	Y
# A simple group to to include both streets										
GStreet,	,		,		,		,	,	Y,	Y

The reference event file defines each reference event with corresponding time and location. The reference event can then be referenced from other event files or the schedule file.

#										
# Sample MyTimetableReferenceEvent.csv										
#										
Reference	Start Time,		Duration,		Location,		Init Actor	Give Tokens...		
# Train A arrivals at portals in the GTrackA group (could be repeated any number of times)										
Arrive_A1,	14:40:00,		,		GTrackA,		,			
Arrive_A2,	14:55:00		,		GTrackA		,			
# Train B arrivals at portals in GTrackB group (could be repeated any number of times)										
Arrive_B1,	14:45:00,		,		GTrackB,		,			
Arrive_B2,	14:54:00,		,		GTrackB,		,			
# Festival event happening at Portal_Street1										
Festival,	15:00:00,		01:00:00,		Portal_Street1		InitFestival			

The schedule generates agents. Some agents are based on the locations and times of reference events, and others are simply bound from one portal or location to another.

#										
# Sample MyTimetableSchedule.csv										
#										
From,	To,		Populat	Time Off	Curve,	Avatar,	Profi	Init Act	Give Tokens...	
# agents departing on tracks A and B (independent of any reference events)										
GStreet,	GTrackA,	200,	14:15:00	CConstar	Red					
Portal_1	GTrackA,	100	14:15:00	CConstar	Red					
GStreet,	GTrackB,	200,	14:15:00	CConstar	Blue					
Portal_1	GTrackB,	100,	14:15:00	CConstar	Blue					
# agents arriving on trains A1 and A2 (over 1s), with a -5s offset to make										

Arrive_A	GStreet,	20,	-5,	,	Green				
Arrive_A	Portal_Inside	6,	-5,	,	Green				
# agents arriving on trains B1 and B2 (over 1s), with a -5s offset to make sure the									
Arrive_B	GStreet,	25,	-5,	,	White				
Arrive_B	Portal_Inside	10,	-5,	,	White				
# agents going to the festival, starting to appear 45 minutes before it starts									
Portal_I	Festival,	100,	-00:45:	(CConsta	Yellow				
GStreet,	Festival,	57,	-00:45:	(CConsta	Yellow				
# agents arriving on trains and going to the location defined by the Festiv									
Arrive_A	Festival,	3,	-5s,	,	Orange				
Arrive_B	Festival,	6,	-5s,	,	Orange				
Arrive_B	Festival,	1,	-5s,	,	Orange				

The gate file defines groupings of links for easy reference in the gate event file:

#													
#	Sample	MyTimetableGate.c											
#													
Group	N	Gate_A1,	Gate_A2,	Gate_A	Gate_A	Gate_A	Gate_A	Gate_B	Gate_B	Gate_B	Gate_B	Gate_B	Gate_B
# Combine all gates for track A into a single group for easy refer													
Gates_A,	Y,	Y,	Y,	Y,	Y,	Y,							
# Track B has even arrival distribution and departing													
Gates_B,	,	,	,	,	,	,	Y,	Y,	Y,	Y,	Y,	Y,	Y,

The gate event file opens and closes the gates based on train arrival reference events

#													
#	Sample	MyTimetableGateEvent.csv											
#													
Reference	Time Offset,	Duration,	Apply Onl	Key Token,	Gate								
# Open gates briefly for arriving agents only (those generated from the schedule ba													
Arrive_A1	00:00:00,	10,	Y,	,	Gates_A								
# Open gates for all other agents departing on track A													
Arrive_A1	00:00:10,	50,	,	,	Gates_A								
# Similar events for all other arrivals													
Arrive_A2	00:00:00,	10,	Y,	,	Gates_A								
Arrive_A2	00:00:10,	50,	,	,	Gates_A								

Arrive_B1	00:00:00,	10,	Y,	,	Gates_B		
Arrive_B1	00:00:10,	50,	,	,	Gates_B		
Arrive_B2	00:00:00,	10,	Y,	,	Gates_B		
Arrive_B2	00:00:10,	50,	,	,	Gates_B		

#### 4.3.3.5.6 Broadcast

The broadcast event is used to apply the specified action to agents within the simulation. The event can be applied selectively based on whether or not an agent passes a test or is in a specified zone.

When the event becomes active the action is applied to all agents currently in the simulation. While active, the action is applied to all agents as they enter the simulation. The action is generally only applied to a given agent once, even if properties of that agent change while the event is still active.

If a target zone is specified then only agents within the zone receive the action. When using a target zone, agents that enter the zone while the action is active will have the action applied as if it were a zone 'On enter' action. Once inside the zone, the action will not be re-applied to an agent even if properties of that agent change while the event is still active. However, if the agent leaves and then re-enters the zone, the action will be reapplied on re-entry.

If a target test is specified then only agents passing the test will receive the action.

#### Active

When active the broadcast event is applying the action to new agents as they enter the simulation or target zone.

Time	
<b>Active</b>	Conditions under which the event becomes active. For more information on controlling event timing see <a href="#">Triggering Events</a> .
<b>Area of Effect</b>	If set, then only agents inside the specified zone will respond to the event. When the event first becomes active it is applied to all agents in the zone. As the event remains active it is also applied to each agent that enters the zone, for as long as the event remains active.
<b>Apply to Agents</b>	If set, then only agents that pass the specified test will receive the action. The test is used only when the broadcast event becomes active or agents enter an effected zone while the event is active. The action will not be applied to agents for whom the test becomes true while the event is active.
<b>Action</b>	The action that will be applied to agents. The action is applied after any actions related to agent creation, but before any actions received during the frame.

4.3.3.5.7 Gate Access

Gate access events are used to control agent access to various [connection objects](#). When agents encounter a gate that is closed they will either wait until the gate is opened or search for another route to their goal. [Escalators](#), [links](#), [paths](#), [ramps](#) and [stairs](#) must have gating enabled in their "Access" properties before they can be used in a gate event.

The gate event can be used to open or close the target object(s). Access can be specified for both directions or a single direction across the object. Access can also be controlled for a sub population of agents by specifying an agent test.

When multiple events act on the same gate object, the most recently applied event takes precedence.

A [vehicle event](#) is a convenient combination of an open gate event and a [journey event](#).

**Active**

When active the gate access event will be opening/closing the specified gates.

**Properties**

Properties	
<b>Active</b>	Conditions under which the event becomes active. For more information on controlling event timing see <a href="#">triggering events</a> .
<b>Timing: Cycle</b>	If cycling is enabled, the event will alternate between on and off while active. If the event becomes active multiple times in the simulation, it will restart the 'On' timer each time it becomes active.  <b>On:</b> The gate will open/close for this duration after becoming active (or until the event is no longer active). <b>Off:</b> The gate will return to its original state for this duration.
<b>Command</b>	<b>Open:</b> The gate will be opened for as long as the event remains active. If the gate is already opened then there is no change. <b>Close:</b> The gate will be closed for as long as the event remains active. If the gate is already closed then there is no change.
<b>Direction</b>	Determine the direction of travel affected by the event.
<b>Gates</b>	Gated connection objects that will be opened by the event. A connection object may be used by several different open gate events.
<b>Apply to agents</b>	If enabled, the event command will only apply to agents for which the specified test evaluates to true. For example, if the event command is set to close and a 'Has token' test is specified, only agents with the given token will see the gate as closed. .

**Collections in gate events**

[Collections](#) can be used in the "Gates" property. All member links with gates enabled will be opened as if they were directly used.



## 4.3.3.5.8 Server Access

Server access events are used to control agent ingress and/or egress to and from [server](#) objects. The same event can be used to open or close the entry onto a server or the exit from a server.

When the entry to a server is closed the server is unavailable to agents and will be ignored during [dispatch](#).

When the exit of a server is closed processed agents will remain on the server until the exit is opened. Agents held at the server exit can block other agents from being processed and are counted when determining the server's available capacity.

When multiple events act on the same gate server, the most recently applied event takes precedence.

**Active**

When active the server access event will be opening/closing the specified servers.

**Properties**

Properties	
<b>Active</b>	Conditions under which the event becomes active. For more information on controlling event timing see <a href="#">triggering events</a> .
<b>Timing: Cycle</b>	If cycling is enabled, the event will alternate between on and off while active. If the event becomes active multiple times in the simulation, it will restart the 'On' timer each time it becomes active.  <b>On:</b> The gate will open/close for this duration after becoming active (or until the event is no longer active). <b>Off:</b> The gate will return to its original state for this duration.
<b>Command</b>	<b>Open:</b> The server will be opened for as long as the event remains active. If the server is already opened then there is no change. <b>Close:</b> The server will be closed for as long as the event remains active. If the server is already closed then there is no change.
<b>Target</b>	Determine whether the entrance, exit, or both are opened or closed by the event.
<b>Servers</b>	Server objects that will be opened or closed by the event. A server may be used by several different server events.

**Collections in server events**

[Collections](#) can be used in the "Servers" property. All member servers will be opened/closed as if they were directly used.

## 4.3.3.5.9 Cache Change

The cache change event is used to alter the value of one or more cache [tally](#) objects during a simulation.

When the event becomes active, a value is added to the the specified tally objects. This value can be a constant or based on other tally objects. Tally objects can also optionally be reset to their default values before being changed.

Only tally objects set to act as a cache can be changed.

**Active**

When active the cache change event will apply the specified change command to the specified tally objects at the specified frequency.

Time	
<b>Active</b>	Conditions under which the event becomes active. For more information on controlling event timing see <a href="#">triggering events</a> .
<b>Frequency</b>	Determines how often the command will be applied to the tally objects when the event becomes active.  <b>Once when active:</b> Apply the command once each time the event changes from inactive to active.  <b>Periodically when active:</b> Apply the command when the event changes from inactive to active, and then again after each specified interval for as long as the event remains active.
<b>Command: Reset to default</b>	If set, the tally objects will be reset to their default values before any new values are added. If multiple events or actions are resetting and adding to the same tally, all resets are performed before all additions.
<b>Command: Add</b>	The value to be added to the target tally objects. The value will be recalculated each frame in which the command is executed. When multiple events or actions are adding to the same tally, the result will be the sum of all values from each event and action.
<b>Tally Targets</b>	The tally objects targeted by the command. These tally objects must be configured to act as caches.

**4.3.3.6 Tally**

A tally represents a numerical value. The value can be a measure of some quantity in the scene, or an abstract count. Scene quantities might be the population in an area, or the available capacity of a server. Scene quantities always reflect the current state of the scene and change as the scene changes. An abstract count is termed a *cache* and can be used to record user defined values. Cache values can be modified during a simulation through either a [cache change](#) event or an "Add to cache" [action](#).

Tally objects can be useful for tracking/recording values of interest during a simulation. They can also be used to drive simulation operations through the [tally test](#) or [tally trigger](#).

Special Tally	
<b>Cache</b>	An abstract value which can be modified

	through the "Add to cache" <a href="#">action</a> or the <a href="#">cache change</a> event.
<b>Constant</b>	A constant value that will not change.
<b>Named tally</b>	The value calculated through another tally object.

<b>Compound Tally</b>	
<b>Difference</b>	Subtract the value of the second tally from the first (A - B). Tests or triggers can check if this difference is positive or negative to see which of the two tally values is greater.
<b>Scale by</b>	Multiply the value of the tally by the given scale.
<b>Sum of</b>	Add together the values from each of the tallies.

<b>Measured Tally</b>	
<b>Area population</b>	A count of the number of agents in an area. Agents will not be double counted if they appear in multiple areas.
<b>Server queue</b>	The count of agents currently queuing or being processed by a server.
<b>Server queue and approach</b>	The count of agents registered with a server. This includes the queue count as above plus any agents currently moving towards the server.
<b>Server available capacity</b>	The count of spaces left at the server. This entry is only valid for servers with fixed capacity. For servers with unlimited capacity the result is infinite.

#### 4.3.3.7 Actions

An action is an operation applied to an agent. The operation can modify agent properties or assign new [tasks](#). Actions can be applied to agents through [events](#) or as agents transition between objects in the scene (see [where to use actions](#)).

Modifying actions alter the agent immediately as the action is applied. Task giving actions create a new task for the agent to execute in the next frame. When a compound or named action results in multiple tasks, those tasks are collected into a group and executed by the agent in order.

**Special Actions**

Action	
<b>Do nothing</b>	No action will be applied to the agent.
<b>Named action</b>	The action described by the action object will be applied to the agent.

**Compound Actions**

Compound actions combine tests and other actions.

Action	
<b>Choose action from set</b>	<p>Only one action from the specified list will be applied to the agent. The weight beside each action describes the likelihood that action will be chosen. The sum of the likelihood values is always normalized to 100% before the action is applied to an agent.</p> <p>A single action is always chosen, even if that action is 'Do nothing'.</p>
<b>Do tasks until</b>	<p>Creates a do-until task which executes subtasks until a condition is met.</p> <p>Contains a 'do' action and 'until' condition. The 'do' action is applied as normal. Any tasks collected by the 'do' action are executed conditionally by the do-until task. If at any time during task execution the 'until' condition is met, the do-until and all subtasks are canceled and removed from the agent's task stack.</p> <p><b>Wait if done early:</b> When enabled, a wait task is appended to the collected subtasks. If the agent finishes the 'do' tasks before the 'until' condition is met (or the 'do' action did not generate any tasks), the appended wait task will execute until the 'until' condition is met.</p>
<b>If/Then</b>	If the <a href="#">test</a> evaluates to true, the 'Then' action is applied to the agent. Tests are evaluated immediately as the action is applied.
<b>If/Then/Else</b>	If the <a href="#">test</a> evaluates to true, the 'Then' action is applied to the agent, otherwise the 'Else' action is applied. Tests are evaluated immediately as the action is applied.
<b>List of actions</b>	<p>Each action in the list is applied to the agent. If set to 'Given order' the actions are applied from top to bottom. If set to 'Random' the actions are applied in random order.</p> <p>Modifying actions are applied to the agent immediately in the order in which they are applied. The tasks from task giving actions are collected in the order in which the actions are applied and given to the agent as a group of tasks to be executed in that order.</p>

<b>Repeat</b>	<p>Creates a repeat task which applies an action multiple times.</p> <p>Contains a 'repeat' action and an 'until' condition. A repeater task is created which applies the specified action and executes any tasks collected. Once the collected tasks have been completed, the 'until' condition is checked. If the condition is not satisfied at that time, the repeater task begins a new iteration, applying the specified action and executing any collected tasks. The repeater task is complete when an iteration finishes and the 'until' condition is met. Note that the 'until' condition is not checked during task execution but only when the tasks are complete.</p>
---------------	---

### Scene Actions

Scene actions are applied immediately to the scene. In the case of tally related actions, the tally object will not be updated until the next frame.

Action	
<b>Add to cache</b>	Add the specified value to the specified <a href="#">tally</a> objects. The added value is given to the tally objects immediately, but the addition is not evaluated until the end of the frame. Note, negative values can be used for subtraction.

### Modifying Actions

Modifying actions are applied immediately to the agent as the action is executed. Tests performed later in an action will be sensitive to modifications applied earlier in the action.

Action	
<b>Add tokens</b>	Immediately give the specified <a href="#">tokens</a> to the agent. Agents can only hold one copy of a given token, and so any tokens already held by the agent are ignored.
<b>Set avatar</b>	Immediately change the <a href="#">avatar</a> of the agent.
<b>Set color</b>	Immediately change the colour of the agent.
<b>Set network</b>	Immediately change the <a href="#">network</a> used by the agent. If the agent is seeking a goal or evacuating a zone and their current floor is not in the new network the agent will be deleted with an error.
<b>Clear avatar</b>	Immediately return the agent to its original or first assigned <a href="#">avatar</a> . The first avatar assigned to an agent (whether through action or profile) is considered the agent's original avatar. Clear avatar actions will always return an agent to this original avatar. In the case where an agent does not have an avatar or is currently using the first assigned avatar, this action does nothing.
<b>Clear color</b>	Immediately return the agent to its original colouring.
<b>Clear history</b>	Immediately clear the agent's route history. The agent will forget any previous journeys and clear all backtracking penalties associated with routes already traversed.

<b>Clear network</b>	Return the agent to their original <a href="#">network</a> as defined by their birth profile.
<b>Clear tasks</b>	Immediately clear the agent's existing list of tasks. This has no effect on the tasks currently being generated by the action under execution. Newly generated tasks will still be given to the agent after the action has finished executing. If the intention is to clear all tasks including any generated by the current action, 'Clear tasks' can be wrapped in an 'Apply action' action (see below).
<b>Clear tokens</b>	If the agent is currently holding any of the specified <a href="#">tokens</a> , immediately remove them from the agent.

**Task Giving Actions**

Task giving actions create a task that is given to the agent for execution in the next frame. The agent will put aside the task it was previously executing and start on the new task(s). Once all new tasks have finished the agent will return to the original task that was interrupted by the action.

Action	
<b>Apply Action</b>	<p>Wrap the contained action in a task. Rather than being applied immediately, the contained action is applied when the task is executed. For example, consider the follow two actions applied in sequence:</p> <pre>1. Seek portal: Platform 2. Set color: Blue.</pre> <p>The agent is immediately coloured blue and given the 'Seek portal' task.</p> <p>Now consider use of the 'Apply action' action:</p> <pre>1. Seek portal: Platform 2. Apply action( Set color: Blue )</pre> <p>The agent is given two tasks: seek portal and apply action. The agent will execute the tasks in sequence, only changing colour to blue after having reached the Platform portal.</p>
<b>Evacuate zone</b>	<p>The agent will navigate the scene, following the best cost route through the zone to the first object that is outside of the zone. Best cost includes all standard cost components such as horizontal distance and queuing. The task is complete once the agent has reached an object that is not a member of the zone. Agents that are given this task when already outside of the zone will complete the task immediately.</p> <p><b>Erase Route History When Task Starts:</b> If true, the agent will forget all previous journeys and there will be no backtracking penalties for routes traversed before the task began. If false, the agent will be biased against selecting routes traversed while executing previous tasks.</p>
<b>Exit simulation</b>	When executed, the agent will leave the simulation. The exit will be recorded as a 'success'. This is the normal method for removing agents from the simulation after they have completed their journeys/tasks.

<p><b>Follow sign</b></p>	<p>The agent will move towards and transition onto the specified adjacent floor, portal, link, stair, ramp, path, or escalator. It is possible to use an elevator in a 'Follow sign' action but only if the elevator does not require information about a destination stop.</p> <p><b>Grouped: lowest cost:</b> Agents will choose the object with the lowest route cost. Any object not connected to the current floor will be ignored. The agent will continuously re-evaluate its choice as conditions on the floor change. The task is complete once the agent has reached the chosen object.</p> <p><b>Sequence: in order:</b> Agents will navigate from object to object in the order given. The first object must be connected to the current floor. Subsequent objects must be connected to one another in an unbroken chain. Collections can be used to specify a number of options for each 'leg' in the sequence. When a collection is used, agents will choose one object from the collection based on lowest cost. Objects in a collection that are not connected to the current floor are ignored. The task is complete when the agent has reached the final object.</p> <p><b>Single: by chance:</b> Agents will be assigned one of the specified objects by chance. Objects that are not connected to the current floor will be ignored. The task is complete once the agent has reached the assigned object.</p>
<p><b>Seek</b></p>	<p>The agent will navigate the scene, following the best cost route to the specified destination. The task is complete once the agent has reached the destination. Possible destinations include:</p> <p><b>Areas :</b> The agent will navigate the scene, following the best cost route to any of the objects in the identified area. The task is complete once the agent has reached any of the objects in the area. It is not possible to seek a volume. It is possible to seek an elevator, but only if the elevator does not require information about a destination stop.</p> <p><b>Origin:</b> The agent will navigate the scene, following the best cost route to the portal through which the agent entered the simulation. The task is complete once the agent has reached its origin portal.</p> <p><b>Portals:</b> The agent will navigate the scene, following the best cost route to the assigned portal(s). The task is complete once the agent has reached the assigned portal.</p> <p><b>Grouped: lowest cost:</b> Agents are given all portals and instructed to seek the portal with the lowest cost route. Agents will continuously re-evaluate route costs as they navigate the scene and alter their choice as conditions change.</p> <p><b>Single: by chance:</b> The agent will be assigned a single</p>

	<p>portal by chance. Weights can be specified to alter the likelihood of each portal being assigned. See <a href="#">choosing objects</a> for information on how weights are used in portal assignment.</p> <p><b>Process start:</b> The agent will navigate the scene, following the best cost route to the start of the assigned process chain.</p> <p>It is possible to specify a number of process chain entries and have one assigned to the agent based on chance. However, if agents are to be distributed over many servers based on server availability or tokens, then the servers should be connected together into a single dispatch, and the single dispatch specified in the seek process action (see <a href="#">process chains &amp; servers</a>).</p> <p>Once the agent has reached the process chain start, it will enter the chain and continue through the chain until it reaches a chain end point. The task is complete once the agent has reached the chain end point and been processed by the last server.</p> <p>Note, if a 'Seek process start' task is interrupted by another task, agents will return to the beginning of the process chain when resuming the task.</p> <p><b>Erase Route History When Task Starts:</b> If true, the agent will forget all previous journeys and there will be no backtracking penalties for routes traversed before the task began. If false, the agent will be biased against selecting routes traversed while executing previous tasks.</p>
<p><b>Wait</b></p>	<p>The agent will wait in place until the specified condition is met. The behaviour of the agent while waiting is determined by the specified wait style.</p> <p><b>For duration:</b> The agent will wait for the specified number of seconds. The wait timer starts when the agent begins executing the task. The wait timer is paused when the task is interrupted by other tasks and resumes from where it left off when the wait task resumes. The task is complete once the specified number of seconds have elapsed.</p> <p><b>Indefinitely:</b> The agent will wait forever until the task is interrupted.</p> <p><b>Until time:</b> The agent wait until the simulation reaches the specified time. The task is complete once the given time has been reached. The task will be completed in one frame if executed after the specified time.</p>



	<p><b>While test true:</b> The agent will wait until the given test evaluates to true.</p> <p><b>Style:</b> Sets the waiting behaviour of agents for the duration of the wait task. See <a href="#">wait style</a> for a description of the available behaviours.</p>
--	---

#### 4.3.3.7.1 Where to Use Actions

Actions can be applied to agents from a number of different sources. They can be applied at birth via the [event](#) or [profile](#) that creates the agent. They can be applied as agents enter or exit [floors](#), [links,zones](#), or other areas. They can be applied as agents come to the end of a [server](#). Or they can be applied through a general [broadcast](#) event. If an agent receives actions from different sources in the same frame, it is important to understand the [order of action execution](#).

Object	Property	Description
Profile	On birth	Applied to an agent newly created using the profile.
Journey, Circulate, Evacuate, Vehicle, Timetable	On birth	Applied to an agent newly created by the event.
Broadcast	Action	Applied to every agent in the scene at the time specified, or as agents enter the simulation during the time the event is active. In the case where the event targets a zone, when the action is active agents will receive the action when they enter the zone.
Portal	Entered simulation	Applied to an agent when it uses the portal to enter the simulation.
Portal	Arrived at waypoint	Applied to an agent when it reaches a portal that it was seeking.
Floor	On enter	Applied to an agent once it has stepped onto the floor.
Floor	On exit	Applied to an agent once it has stepped off of the floor.
Link, Stair, Ramp, Escalator, Path	Ball: On enter	Applied to an agent once it has stepped onto the link at the ball goal line.

Link, Stair, Ramp, Escalator, Path	Ball: On exit	Applied to an agent once it has stepped off of the link at the ball goal line.
Link, Stair, Ramp, Escalator, Path	Box: On enter	Applied to an agent once it has stepped onto the link at the box goal line.
Link, Stair, Ramp, Escalator, Path	Box: On exit	Applied to an agent once it has stepped off of the link at the box goal line.
Server	Done processing	Applied to an agent once it has been processed by the server but before it has been released. Agents may be held at a server due to a lack of downstream capacity or a closed server exit. See <a href="#">server</a> for more information.
Server	On release	Applied to an agent when it is released from a server.
Zone	On enter	Applied to an agent once it has stepped onto a member of the zone when coming from an object that is not a member of the zone.
Zone	On exit	Applied to an agent once it has stepped onto an object that is not a member of the zone when coming from an object that is a member of the zone.

4.3.3.7.2 Order of Action Execution

The order in which actions are applied is important, as test results in one action could be altered by the results of previous actions (such as giving or removing tokens). The following table describes the order in which actions are applied for various scenarios. In each case, actions are only applied if defined.

Scenario	Order of Actions (if present)
A new agent is created and placed in the scene.	<ol style="list-style-type: none"> <li>1. Creation event 'Internal' (initial goal assignment)</li> <li>2. Profile: 'On birth'</li> <li>3. Creation event 'On birth' (including Timetable)</li> <li>4. Broadcast/Timetable action(s): 'Action'</li> <li>5. Zone(s): 'On enter'</li> <li>6. Floor: 'On enter'</li> <li>7. Portal: 'Entered simulation'</li> </ol>

An agent has transitioned from the ball side of a link onto a floor.	<ol style="list-style-type: none"> <li>1. Broadcast/Timetable action(s): 'Action'</li> <li>2. Link: 'Ball: On exit'</li> <li>3. Zone(s): 'On exit'</li> <li>4. Zone(s): 'On enter'</li> <li>5. Floor: 'On enter'</li> </ol>
An agent reaches its portal destination.	<ol style="list-style-type: none"> <li>1. Broadcast/Timetable action(s): 'Action'</li> <li>2. Portal: 'Arrived at waypoint'</li> </ol>
An agent has finished being processed at a server.	<ol style="list-style-type: none"> <li>1. Broadcast/Timetable action(s): 'Action'</li> <li>2. Server: 'When finished'</li> </ol>

#### 4.3.3.8 Tests

Agent tests operate on a single agent at an instant in time, and return either true or false. Tests are used in [agent actions](#) to customize how the action is applied to an agent and in a number of scene objects for controlling how properties are applied to agents. See [where tests are used](#) for more information.

##### Special Tests

Test	
<b>Always true</b>	Always returns true.
<b>Named test</b>	Returns the result of executing the specified agent test object.

##### Compound Tests

Compound tests combine the results of other tests.

Test	
<b>Not</b>	Returns the inverse of the specified test.
<b>Compound Test</b>	Combines the results of the two tests using the specified logic, and returns the result.

<b>All of</b>	Returns true if all of the specified tests return true.
<b>Any of</b>	Returns true if any of the specified tests return true.
<b>None of</b>	Returns true if none of the specified tests return true.

**Scene Test**

Scene tests return true based on conditions within the simulation. The results will vary with time.

Test	
<b>Clock in time range</b>	Returns true if the current simulation time is before/after the specified time.
<b>Events active</b>	Returns true if any/all/none of the specified events are actively engaged in the specified activity.
<b>Gate state</b>	Returns true if any/all/none of the specified gates are open/closed.
<b>Random chance</b>	Returns true the given percentage of the time.
<b>Server state</b>	Returns true if any/all/none of the specified servers are open/closed.
<b>Tally in range</b>	Returns true if the <a href="#">tally</a> resolves to a value in the specified range. This can be used to test area populations, server queue counts, or anything else that can be measured with a tally.

**Static Tests**

Static agent tests will always return the same result for a given agent and do not vary with time.

Test	
<b>Agent: Created by</b>	Returns true if the agent was created by any of the specified events.
<b>Agent: Entered at</b>	Returns true if the agent entered the simulation at any of the specified portals.
<b>Agent: Initial exit</b>	Returns true if the agent was created with any of the specified portals as its initial exit/goal. The initial exit refers to the single and final portal destination assigned to the agent by the creating event. It does not consider any goals applied via actions or any intermediate or circulation goals assigned by an event. Not all agents will have an initial exit.
<b>Agent: Uses profile</b>	Returns true if the agent was created with any of the specified profiles.

**Dynamic Tests**

Dynamic agent tests can produce different results at different times. They depend on temporal

values or agent properties that can change between frames, within frames, or even as actions are applied.

Test	
<b>Agent: Has age</b>	Returns true if the agent is currently younger/older than the specified age.
<b>Agent: Has tokens</b>	Returns true if the agent is currently holding all/any/none of the specified tokens.
<b>Agent: In area</b>	Returns true if the agent is currently in any of the specified areas. Valid areas include floors, links, ramps, stairs, escalators, paths, volumes, or collections.

#### 4.3.3.8.1 Where to use Tests

The following table lists the ways in which tests can be used.

Actions	
If/Then	Used to determine whether or not the action is applied.
If/Then/Else	Used to determine which action is applied.
Do Until	Can be used to determine when the Do Until task is done.
Task: Wait	Can be used to determine when the Wait task is done.

Events		
<a href="#">Broadcast</a>	Apply to Agents	Controls whether or not the action is applied to an agent.
<a href="#">Gate Access</a>	Apply to Agents	Controls whether or not the operation performed by the gate event is visible to an agent.
<a href="#">Vehicle</a>	Boarding/Alighting Access	Controls whether or not agents can access the vehicle gates.

Scene Objects		
<a href="#">Server</a>	Preferred/Required Access	Identify required or preferential conditions for access to the server.

<a href="#">Server</a>	Contact Time	Specify a contact time duration for an agent.
------------------------	--------------	---

#### 4.3.3.9 Time

Reference times are virtual events that do not directly impact simulation. They are useful for representing significant times in a simulation such as a fire alarm or the end of a concert. If other dependent events refer to the reference time, then only the reference time needs to be modified to reflect operational changes. See time references in [working with time](#).

##### Active

The time event is active for the brief instant corresponding to the specified time.

##### Properties

Properties	
<b>Timing: Start</b>	When the reference time event is considered to have started.

#### 4.3.3.10 Triggers

A trigger can fire in response to conditions in the simulation. When a trigger fires, connected events become active. Some triggers fire in a discrete manner and become active for a single frame at a time. Other triggers fire continuously over an interval. The type of triggers available will depend on whether the context calls for a discrete or interval based trigger. Some triggers, like the 'Whenever' or 'Beginning of' triggers, can convert the output of an interval trigger into a discrete single frame response.

For information on using triggers to drive events, see [triggering events](#).

##### Time Triggers

Time triggers are discrete and pulse once at the indicated time.

<b>Absolute time</b>	Fire once at the specified simulation time.
<b>Event start time</b>	Fire once at the specified offset from the start time of the indicated event. The offset can be negative. The event must have a resolvable fixed start time that is not scene dependent.
<b>Simulation start</b>	Fire once at the specified offset from the simulation start.
<b>Simulation end</b>	Fire once at the specified offset from the simulation end.

##### Special Triggers

Special triggers are discrete and fire at some time in relation to a reference trigger.

--

<b>Blocking</b>	Fire when the reference trigger fires, then do not fire again until after the specified interval. During the interval the reference trigger continues to be active, but is ignored.
<b>Delayed</b>	When the reference trigger fires, wait the specified interval and then fire. During the interval the reference trigger continues to be active but is ignored.
<b>Named trigger</b>	Fire whenever the reference trigger object fires.
<b>Periodic</b>	Fire when the reference trigger fires, and then repeatedly for the specified count with each iteration separated by the specified period. When repeating, the reference trigger continues to be active but is ignored until the final count has completed.

### Compound Triggers

Compound triggers combine the results from multiple triggers.

<b>All of</b>	Fire only when all of the reference triggers are firing.
<b>Any of</b>	Fire when any of the reference triggers are firing.
<b>None of</b>	Fire whenever none of the reference triggers are firing.
<b>Not</b>	Fire whenever the reference trigger is not firing.

### Impulse Triggers

Impulse triggers convert interval triggers into a discrete trigger.

<b>Beginning of</b>	Fire once when the reference trigger begins firing. Ignore the reference trigger until it has stopped firing.
<b>End of</b>	Fire once when the reference trigger stops firing. Ignore the reference trigger until it has started firing again.
<b>Whenever</b>	Fire once whenever the reference trigger fires. This can be used to convert an interval based trigger into a continuously firing discrete trigger.

### Scene Triggers

Scene triggers fire based on conditions in the simulation.

--	--

<b>Clock in time range</b>	Fire when the simulation clock is within the specified range.
<b>Event</b>	Fire when any of the specified event is actively engaged in the specified activity.
<b>Gate state</b>	Fire when any of the specified gates are in the specified state.
<b>Over duration</b>	Fire continuously over the specified time interval.
<b>Server state</b>	Fire when any of the specified servers are in an open/close state which matches the specified state.
<b>Tally in range</b>	Fire when the <a href="#">tally</a> resolves to a value in the specified range. This can be used to fire based on area populations, server queue counts, or anything else that can be measured with a tally.
<b>Transition count</b>	Fire once the specified number of agents have crossed the specified <a href="#">transition</a> . Counting begins at 0 once the reference trigger fires. Counting continues until the target number is reached or exceeded. If the trigger is set to loop, the count is reset to 0 immediately after the trigger fires.

### 4.3.4 Analysis

<b>Analysis Object Type</b>	<b>Description</b>
<a href="#">Simulation Run</a>	Represents the results of a simulation by defining links to results database files (.mmdb). These form the basis the analysis system and are used as a parameter to most types of queries.
<a href="#">Agent Filter</a>	Used by queries to target agents meeting specific criteria, including location, density, etc.
<a href="#">Trip</a>	Represents an ordered series of locations an agent may traverse over its lifetime.
<a href="#">Graphs</a>	A query object that analyzes one or more simulation runs to produce graphs of data over time. Graphs can be exported as an image or csv text files.
<a href="#">Maps</a>	A query object that analyzes one or more simulation runs and displays the results by colouring the surface of existing objects. Maps are visible in the <a href="#">3D scene view</a> .
<a href="#">Tables</a>	A query object that analyzes one or more simulation runs and produces the results in table form. Tables can be exported as csv text files.



#### 4.3.4.1 Simulation Run

A simulation run represents a single iteration of a simulation. It maintains a connection to a generated database and is used by both playback and analysis queries to extract results from disk.

A simulation run can be created automatically when a [simulation is executed](#). The new simulation run will automatically connect to the new database file (see [simulation data](#)). An empty simulation run can also be created from the main window's analysis ribbon and used to connect to an existing database file. This is useful when comparing results from different projects.

#### Multiple Runs

For the purposes of playback and queries, it is possible to have multiple simulation runs reading from the same database file at the same time (perhaps highlighting different areas through each run's agent filter). However, when executing a simulation, all other runs are blocked from reading from the file until the simulation is complete. Similarly, it is not possible to run a simulation and overwrite the database file while the file is being used by the [movie and image export tools](#).

#### Object Properties

The properties of a simulation run object allow users to change which database file is connected as well as the appearance of agents displayed by the object.

Properties	
<b>Database File</b>	The file to which the object is connected. Simulation execution writes to this file while playback and analysis read from this file.
<b>Status</b>	The status of the simulation run's connection to the database. Users can refresh the connection by pressing the refresh button.
<b>Agent Colour</b>	<p>The colour of agents from the simulation run.</p> <p>The following options are available:</p> <ul style="list-style-type: none"> <li>• <b>Custom:</b> User specified colour.</li> <li>• <b>Simulation Run Colour:</b> The colour of the simulation run object.</li> <li>• <b>Database:</b> The agent colours stored in the database. This is the default setting.</li> <li>• <b>Database Darkened:</b> The agent colours stored in the database but darkened.</li> <li>• <b>Database Lightened:</b> The agent colours stored in the database but lightened.</li> <li>• <b>Fruin LOS (auto):</b> Use object type specific Fruin Level of Service colouring. The exact colour cutoff values will vary by the type of object agents are on (eg. stairs vs floors).</li> <li>• <b>Fruin Queuing (Platform) LOS:</b> Use standard Fruin colouring for queuing.</li> <li>• <b>Fruin Stairway LOS:</b> Use standard Fruin colouring for stairs.</li> <li>• <b>Fruin Walkway LOS:</b> Use standard Fruin colouring for walkways.</li> <li>• <b>IATA Wait/Circulate LOS:</b> Use standard IATA (International Air Transport Association) Wait/Circulate colouring.</li> </ul>
<b>Agent Filter</b>	Simulation runs can use an <a href="#">agent filter</a> to modify how agents are displayed. For instance, an agent filter could be used to only show agents that were created by a particular schedule, or give a different colour to agents currently undergoing a particular <a href="#">trip</a> .

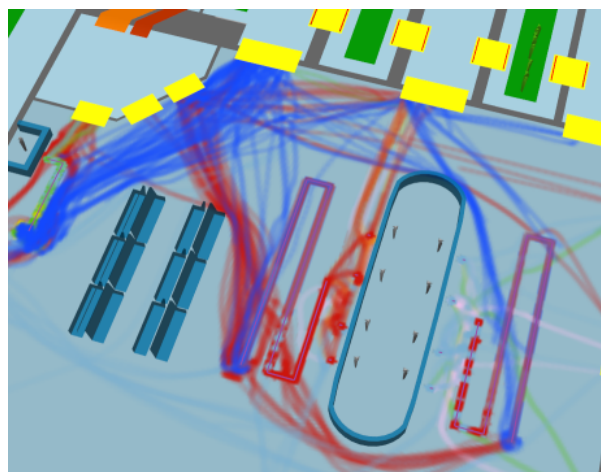
	<p>The following properties control how the filter is used:</p> <ul style="list-style-type: none"> <li>• <b>Enable agent filter:</b> This must be checked to enable the agent filter.</li> <li>• <b>Mode:</b> Set whether agents matching the filter are coloured differently or are hidden. Alternatively, agents <b>not</b> matching the filter can be hidden.</li> <li>• <b>Colour:</b> What colour to use for filtered agents (only available if 'Colour filtered agents' is specified).</li> <li>• <b>Filter:</b> The agent filter to use.</li> <li>• <b>Status:</b> Indicates whether the agent filter is currently applied. Users may need to refresh this when the agent filter or properties are changed.</li> </ul>
--	---

#### 4.3.4.2 Agent Filter

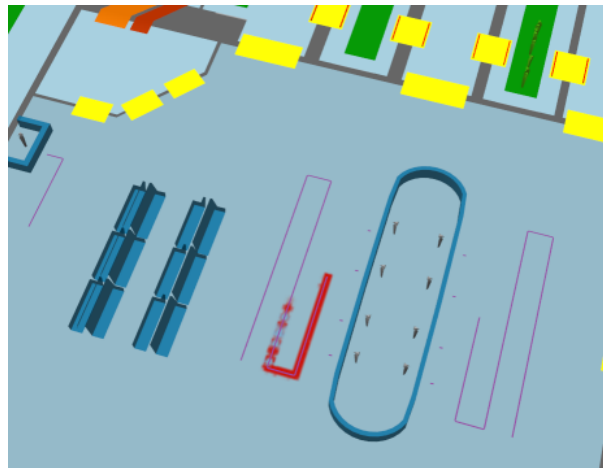
Agent filters provide a powerful way to set up queries (graphs, tables, maps) that operate on a specific (possibly time-varying) set of agents. Refer to the various [graph](#), [table](#) and [map](#) types for descriptions of how filters can be used in different types of queries.

Fundamentally, a filter is an object that, for any instant in time, generates a list of agents satisfying the filter. For instance, a 'holding token' filter would generate, at every instant, a list of all agents holding a particular token at that instant. Some filters combine or transform other filters in different ways. For instance, an 'all of' filter could combine a 'holding token' and an 'at server' filter to create a combined filter that would generate, at any instant, a list of agents at a particular server holding a particular token.

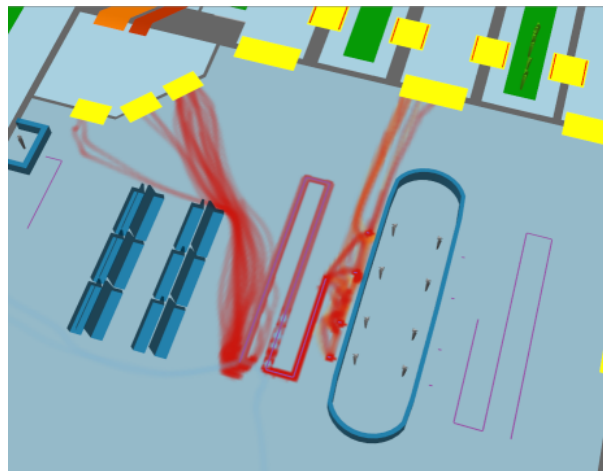
One important type of transforming filter is an 'are ever' filter. This filter can be thought of as a qualifier on another filter; instead of agents only being included in the generated list **while** they satisfy some criterion (and therefore potentially dropping in and out of the generated list over time), they are always included in the list if they **ever** satisfy the criterion during a particular time period. For instance, as shown below, an [agent path map](#) using the filter 'at server' would result in agent paths being shown only for agents currently at the server. By using 'are ever' in combination with 'at server' during a particular time period, the map would instead show paths of agents as they approached and after they left the server (but no paths at all for agents who were never at the server).



Unfiltered paths



Paths filtered by 'At server'



Paths filtered by 'Are ever' > 'At server'

### Filter Debugging

An excellent way to determine if a filter is working as expected is to set a particular [simulation run](#) to colour agents differently based on whether or not they satisfy the filter.

### Building Filters

Filters are built using a cascading set of drop-down menus. Wherever a filter is required, a drop-down menu will be available to select the filter type. Each filter type may reference one or more other objects. Depending on the type of filter chosen, more drop-down menus or entry fields may appear to select those objects. The table below shows the various filter types available, along with their required references and what agents are included in the generated list at every instant.

Note that not all filter types are immediately available in all situations, as some types of queries require a filter that produces a fixed set of agents instead of a dynamically-changing one. In these cases, an 'are ever' filter can usually be used in combination with the desired filter to obtain the desired effect. For instance, it is not possible to create an [origin/destination](#) table for agents at a particular cordon (as it is not clear what that would mean), but it is possible to create such a table for agents that are ever at a particular cordon.

Filter Type	References	Agents Included
All agents	None	All agents.

Filter Type	References	Agents Included
<b>Selected agents</b>	None	Agents that are currently selected. Note that this filter is 'live' and applies to agents that are selected when the filter is actually used, not when it is created. For instance, if an <a href="#">agent path</a> map is created using this filter, whatever agents are selected when the 'Evaluate' button is pressed will have their paths shown. If later on a different set of agents is selected and the map is re-evaluated, the map will change to show the paths of the newly-selected agents.
<b>Named filter:</b>	Named filter	Those included by the referenced named filter.
<b>Are ever:</b>	Filter	Agents that are ever included by the referenced filter at any instant during a particular time period.
<b>Not:</b>	Filter	Agents not included by the specified filter.
<b>Compound filter:</b>	Two filters	Agents included by the two filters combined with the specified logic.
<b>All of:</b>	One or more filters	Agents included by all of the referenced filters at a given instant.
<b>Any of:</b>	One or more filters	Agents included by any of the referenced filters at a given instant.
<b>None of:</b>	One or more filters	Agents included by none of the referenced filters at a given instant.
<b>Created by:</b>	<a href="#">Event</a>	Agents that were initially created by the referenced event.
<b>Entered simulation at:</b>	<a href="#">Portal</a>	Agents that entered the simulation at the given portal. With collections: Agents that entered the simulation at any of the portals in the collection.
<b>Exited simulation at:</b>	<a href="#">Portal</a>	Agents that exited the simulation at the given portal. With collections: Agents that exited the simulation at any of the portals in the collection.
<b>From profile:</b>	<a href="#">Profile</a>	Agents that were created with the referenced profile. With collections: Agents that were created with any of the profiles in the collection.
<b>Has end state:</b>	None	Agents that finished the simulation with the specified state: <b>In simulation:</b> still in scene at simulation end <b>Exited with success:</b> exited simulation as expected <b>Exited with error:</b> was deleted from simulation with an error.
<b>At server:</b>	<a href="#">Server</a>	Agents that are at the referenced server. Agents are considered to be 'at' a server from the first time when they are either being processed by the server or are queueing for the server (blocked by another agent that is at the server), until they are finished being processed by the server. With collections: Agents that are at any of the servers in the

Filter Type	References	Agents Included
		collection.
<b>At transition:</b>	<a href="#">Transition</a>	Agents that are currently performing the specified transition (this will only be true for an instant for any agent and so is often used in combination with 'Are ever').
<b>Local density:</b>	Upper and lower density bounds	Agents that currently have a local density around them that is in the given range.
<b>Current speed:</b>	Upper and lower speed bounds	Agents whose speed is currently in the given range.
<b>Holding token:</b>	<a href="#">Token</a>	Agents currently holding the referenced token. With collections: Agents currently holding any of the tokens in the collection.
<b>In area:</b>	<a href="#">Area</a>	Agents currently in the given area. See <a href="#">area</a> for a description of the different areas. With collections: Agents currently in any of the areas in the collection.
<b>In trip:</b>	<a href="#">Trip</a>	Agents currently in the referenced trip (have started and have not yet finished). See <a href="#">trips</a> for a description of when an agent is considered to be 'in' a trip.

#### 4.3.4.3 Trip

A trip in MassMotion is a way of describing a particular route or path through the model. It is defined by a list of one or more [areas](#), [cordons](#), [portals](#) or servers.

Agents are considered to have completed a trip if they have crossed each of the objects in the trip in sequence. There may be gaps in between objects. For instance, a trip could be specified as two links. Agents that crossed the first link, then traversed an intermediate floor, then crossed the second link would be considered to have completed the trip even though while crossing the floor they were not at either of the two links.

Trips are created by using the 'Trip' button in the Analysis toolbar, or can be created inline when constructing some types of queries (such as the [agent trip time](#) table) or [filters](#) ('In trip').

In addition to specifying an ordered list of objects that define the trip, options exist to define when exactly the trip is considered to begin and end (and therefore when an agent is considered to be 'in' the trip):

Begin when	
<b>Entering first item</b>	The trip begins at the moment the agent enters or crosses the first object. Agents may enter an object by transitioning from another area/object, or entering the simulation within that item.

<b>Entering or at first item</b>	The trip begins at the moment the agent enters or crosses the first object, or at the beginning of the query period if the agent is already on/in the object/area.
<b>Exiting first item</b>	The trip begins at the moment the agent exits or crosses the first object.

<b>End when</b>	
<b>Exiting last item</b>	The trip ends at the moment the agent exits or crosses the last object. Agents may exit an object by transitioning to another area/object, or by exiting the simulation while within that item.
<b>Exiting or at last item</b>	The trip ends at the moment the agent exits or crosses the last object, or at the end of the query period if the agent is still on/in the object/area.
<b>Entering last item</b>	The trip ends at the moment the agent enters or crosses the last object.

Note that for the special case of cordons, all of the above options are equivalent and simply refer to the time at which the agent crossed the cordon. If, therefore, the trip is defined as starting and ending at cordons, the options chosen are irrelevant.

**Collections in trips**

If a [collection](#) is part of a trip, agents are considered to complete that part of the trip if they reach any of the members in the collection. This can be used to construct more complex trips with branching routes.

A current limitation on trips is that each object may only appear once per trip, however collections can be used as a workaround. For example, to create a round trip from FloorA to FloorB and back, a trip can be created consisting of FloorA, FloorB and a collection containing only FloorA.

**4.3.4.4 Graphs**

Graph queries plot the number of agents under various conditions over time, including at certain locations or in crowds above a certain density.

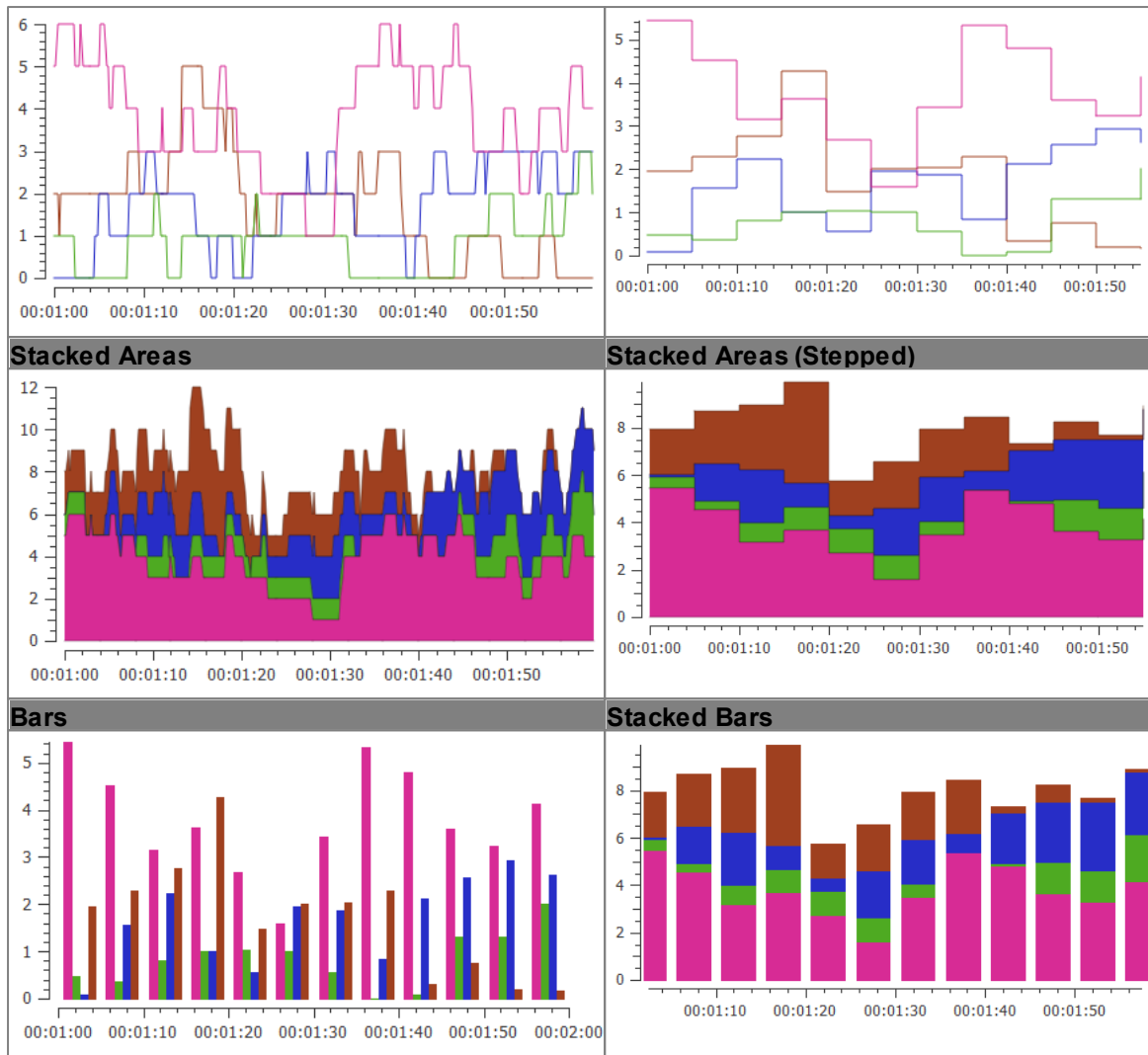
Once evaluated, graphs can be exported to an image or to a CSV text file as a set of values. The following image formats are supported: \*.png, \*.svg, \*.pdf

**Graph Styles**

The title and axis labels of graphs can be set for each graph, as well as the inclusion of a legend. The colour of datasets is set by the groups in the graph structure (see below).

Several styles of graph are available, though not all graphs support each style. Curved graphs plot instantaneous counts over time. Stepped graphs and bar graphs require a "bucket size" which measures the duration over which reported counts are collected. Stacked area and bar graphs additionally support a 'normalized' option. If this is selected, the values at every instant will be normalized to sum to 100. This allows, for instance, indicating the percentage of agents at different density levels in an [agent density](#) graphs instead of the absolute counts.

<b>Curves</b>	<b>Curves (Stepped)</b>
---------------	-------------------------



### Navigating Graphs

Once a graph is generated, subsections of the graph can be examined in further detail by clicking and dragging to zoom in on a region. The view can be reset by middle clicking the graph.

### Graph Structure

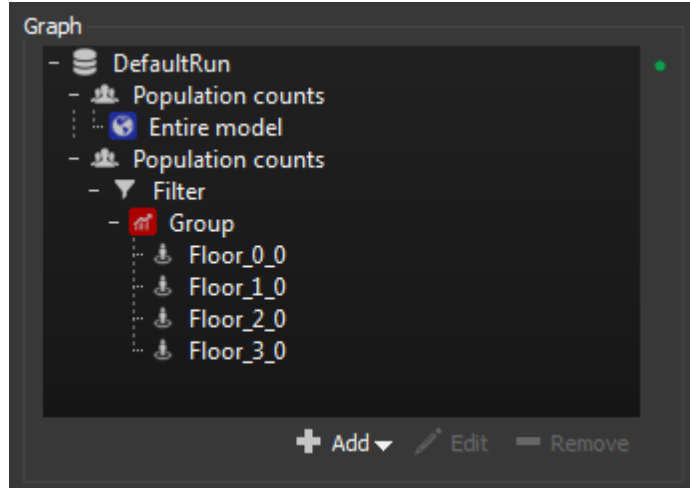
Graphs use a tree structure to organize their source data into datasets. Multiple items can be added at each level; for instance, multiple groups can be placed under a filter.

Items are nested in the following order:

Graph Item	Description
1. <a href="#">Simulation Run</a>	The simulation run to analyse. Most graphs only operate on a single simulation run and have it as a separate "General" parameter.
2. <b>Graph Type</b>	The type of data to graph. Composite graphs can combine different <a href="#">cumulative flow counts</a> , <a href="#">flow counts</a> , <a href="#">population counts</a> or <a href="#">server population counts</a> .
3. <b>Filter (optional)</b>	An <a href="#">agent filter</a> that limits which agents are counted.

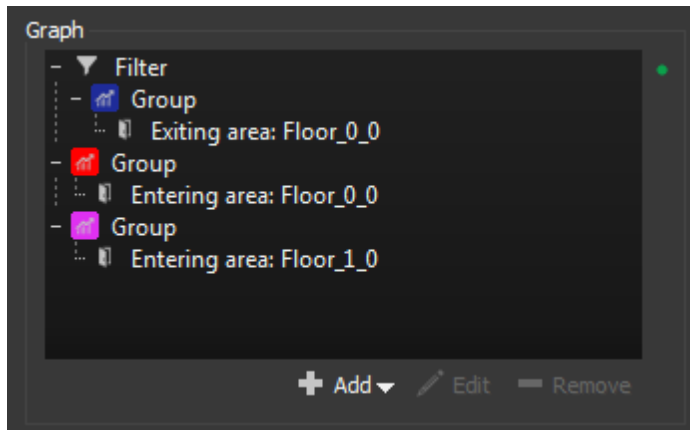
<b>4. Group</b>	A dataset. The colour shown in the graph is set by this item.
<b>5. Group Members</b>	Sources of data which are grouped into a dataset. Counts from individual group members can either be summed, or displayed as separate series on the graph.

[Composite graphs](#) allow full use of this tree structure:



**Composite graph tree structure**

Most graphs, however, use a more simplified tree structure and implicitly add simulation run and graph types.



**Flow graph tree structure**

[Agent density graphs](#) generate the entire tree structure implicitly.

Graph Types	
<a href="#">Cumulative Flow Count</a>	Counts the number of agents that have crossed various <a href="#">transitions</a> .
<a href="#">Flow Count</a>	Counts the number of agents crossing various <a href="#">transitions</a> .
<a href="#">Population Count</a>	Counts the number of agents in different <a href="#">areas</a> .
<a href="#">Server</a>	Counts the number of agents at various servers.



<a href="#">Population Count</a>	
<a href="#">Tally</a>	Graphs the counts of one or more <a href="#">tallies</a> .
<a href="#">Agent Speed Ratio</a>	Counts the number of agents with different ratios of actual to desired speed.
<a href="#">Agent Density</a>	Counts the number of agents at various levels of crowd density.
<a href="#">Volume Density</a>	Computes the average density over time for one or more <a href="#">volumes</a> .
<a href="#">Composite</a>	Fully customizable graph which can combine any number of simulation runs and values.
<a href="#">Performance Graph</a>	Compares performance of <a href="#">simulation runs</a> .

#### 4.3.4.4.1 Cumulative Flow Count Graph

Cumulative flow count graphs measure the running total of agents across a [transition](#). The value at each instant is the total number of agents that have crossed the transition from the start of the reporting period until that time.

When using time intervals (bins), counts for an interval include agents crossing the transition up to the end of the interval.

Groups in a cumulative flow count graph are composed of one or more [transitions](#). Grouped transitions can be summed into a single series, or reported as one series per transition member. If a group is created under a filter, only agents satisfied by the filter will be counted. Note the distinction between using a filter such as 'holding token' versus 'are ever holding token'. In the first case, agents will be included in the flow count only if they are holding the specified token while crossing the specified transition. In the second case, agents will be included if they have ever held the token at any time during the reporting period.

This graph can be thought of as the integral of a [flow count graph](#).

Cumulative Flow Count Parameters	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated. To compare cumulative flow counts between different runs, a <a href="#">composite graph</a> can be used.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	Cumulative flow counts can be displayed as either curves, stepped curves, stacked areas, stepped stacked areas, bars, or stacked bars.

	<p>If stepped curves, stepped areas, bars or stacked bars are chosen, a bin size must be specified. Each stepped portion or bar will then correspond to an interval of time with the size specified, and the associated value will be the <b>total</b> flow during that time interval. If sampling period exceeds bin size, zero values will be present.</p> <p>If stacked areas, stepped areas or stacked bars are chosen, a 'Normalized' option is available. If selected, flow counts will be displayed as percentages instead of absolute values and will always sum to 100.</p>
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Notes</b>	A simple field that can be used to save comments about the graph. This will be saved, but will not be exported to CSV.

4.3.4.4.2 Flow Count

Flow count graphs measure the number of agents crossing specified [transitions](#) during particular time intervals (bins).

Flow count graph are composed of one or more [transitions](#). Grouped transitions can be summed into a single series, or reported as one series per transition member. If a group is created under a filter, only agents satisfied by the filter will be counted. Note the distinction between using a filter such as 'holding token' versus 'are ever holding token'. In the first case, agents will be included in the flow count only if they are holding the specified token while crossing the specified transition. In the second case, agents will be included if they have ever held the token at any time during the reporting period.

Flow Count Parameters	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated. To compare flow counts between different runs, a <a href="#">composite graph</a> can be used.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	<p>Flow counts can be displayed as either curves, stepped curves, stacked areas, stepped stacked areas, bars, or stacked bars.</p> <p>If stepped curves, stepped areas, bars or stacked bars are chosen, a bin size must be specified. Each stepped portion or bar will then correspond to an interval of time with the size specified, and the associated value will be the <b>total</b> flow during that time interval. If sampling period exceeds bin size, zero values will be present.</p>

	<p>If stacked areas, stepped areas or stacked bars are chosen, a 'Normalized' option is available. If selected, flow counts will be displayed as percentages instead of absolute values and will always sum to 100.</p> <p>Curves and stacked areas use a single frame as the interval and are therefore difficult to read, but may useful when exported to CSV for further analysis. A <a href="#">cumulative flow count graph</a> may be easier to read.</p>
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

#### 4.3.4.4.3 Population Count

Population count graphs measure the total number of agents in specified areas over time.

Population count graph are composed of one or more [areas](#). Grouped areas can be summed into a single series, or reported as one series per area member. If a group is created under a filter, only agents satisfied by the filter will be counted. Note the distinction between using a filter such as 'holding token' versus 'are ever holding token'. In the first case, agents will be included in the count only if they are holding the specified token while in the area. In the second case, agents will be included if they have ever held the token at any time during the reporting period.

Population Count Parameters	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated. To compare population counts between different runs, a <a href="#">composite graph</a> can be used.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Sampling period</b>	How frequently to measure the population. Specifying a larger value will reduce the amount of time needed to generate the graph, but using very large values runs the risk of missing peaks in population that may occur between samples.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X Axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y Axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	<p>Population counts can be displayed as either curves, stepped curves, stacked areas, stepped stacked areas, bars, or stacked bars.</p> <p>If stepped curves, stepped areas, bars or stacked bars are chosen, a bin size must be specified. Each bar will then correspond to an interval of time with the size specified, and the associated value will be the <b>average</b> population within that time interval. If sampling period exceeds bin size, zero values will be present.</p> <p>If stacked areas, stepped areas or stacked bars are chosen, a 'Normalized' option</p>

	is available. If selected, population counts will be displayed as percentages instead of absolute values and will always sum to 100.
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

4.3.4.4.4 Server Population Count

Server population count graphs measure the total number of agents at one or more servers over time. See [server](#) for a description of when an agent is considered to be at a server. To obtain detailed information about entire process chains instead of single servers, an [agent process chain time](#) table can be used instead.

Server population count graphs are composed of one or more servers. Grouped servers can be summed into a single series, or reported as one series per server member. If a group is created under a filter, only agents satisfied by the filter will be counted. Note the distinction between using a filter such as 'holding token' versus 'are ever holding token'. In the first case, agents will be included in the count only if they are holding the specified token while at the server. In the second case, agents will be included if they have ever held the token at any time during the reporting period.

<b>Server Population Count Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated. To compare server population counts between different runs, a <a href="#">composite graph</a> can be used. Alternatively, a <a href="#">server summary</a> table can be used to compare aggregate information about different servers in different simulation runs directly.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Sampling period</b>	How frequently to measure the server population. Specifying a larger value will reduce the amount of time needed to generate the graph, but using very large values runs the risk of missing peaks in population that may occur between samples.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X Axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y Axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	<p>Server population counts can be displayed as either individual curves, stacked areas, bars, or stacked bars.</p> <p>If either bars or stacked bars are chosen, a bin size must be specified. Each bar will then correspond to an interval of time with the size specified, and the associated value will be the <b>average</b> count within that time interval. If sampling period exceeds bin size, zero values will be present. If sampling period exceeds bin size, zero values will be present.</p>

	If stacked areas, stepped areas or stacked bars are chosen, a 'Normalized' option is available. If selected, population counts will be displayed as percentages instead of absolute values and will always sum to 100.
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

#### 4.3.4.4.5 Tally Count

Tally count graphs represent the value of one or more tally objects over time. Each tally value is represented as a series. [Collections](#) can be used to sum multiple tally objects into a single series.

Tally Parameters	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Sampling period</b>	How frequently the tally value should be reported. Specifying a larger period will reduce the amount of time needed to generate the graph, but using very large periods runs the risk of missing sudden changes in value.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Tallies</b>	Which tallies to include in the graph. Each tally will be represented by a different graph series with series colour taken from the tally object colour. <a href="#">Collections</a> can be used to group tally objects into a single series.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

4.3.4.4.6 Agent Speed Ratio

An agent speed ratio graph counts the number of agents in different speed ratio bands over time. An agent's speed ratio is the ratio of the agent's current speed to its desired normal speed (as assigned by the agent's [profile](#)). A value of 1 indicates the agent is walking at its desired speed while a value of 0 indicates the agent is not moving. Speed ratios between 0 and 1 represent the degree to which an agent is obstructed or prevented from moving normally.

The speed ratio ranges can be specified by the user. The default ranges represent the expected reductions in speed at the various Fruin LOS density bands F - A for agents whose desired speed is 1.35m/s[1].

LOS	ft/min	m/s	Ratio of max 1.35m/s
A	265	1.35	1
B	260	1.321	0.963
C	250	1.270	0.941
D	240	1.219	0.904
E	225	1.143	0.844
F	150	0.762	0.563

Table 1: Default speed ratio ranges using the Fruin .

Agent Density Parameters	
<b>Simulation run</b>	The simulation run for which counts should be calculated.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Sampling period</b>	How frequently to count the numbers of agents at different speed ratios. Specifying a larger value will reduce the amount of time needed to generate the graph, but using very large values runs the risk of missing peaks that may occur between samples.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Normalized</b>	If checked, agent counts will be normalized to sum to 100 (i.e., displayed as percentages instead of absolute values).
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.

<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<a href="#">Agent Filter</a>	Used to specify which agents should be counted in the graph.
<b>Speed Ratio Ranges</b>	Specifies the speed ratio bands into which agents are sorted and counted.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

[1] Fruin, John J. Pedestrian Planning & Design, Revised Edition Chapt. 4, Elevator World, 1987 - Chapter 3

#### 4.3.4.4.7 Agent Density

An agent density graph displays the number or proportion of agents at various densities (see [LOS Colour Mapping](#)). The graph will be shown as stacked areas, with each band corresponding to a count of agents in a given density range.

<b>Agent Density Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated.
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Sampling period</b>	How frequently to count the numbers of agents at different densities. Specifying a larger value will reduce the amount of time needed to generate the graph, but using very large values runs the risk of missing peaks in density that may occur between samples.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Normalized</b>	If checked, agent counts will be normalized to sum to 100.
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<a href="#">Agent Filter</a>	Used to specify which agents should be included in the graph.
<b>Density Ranges</b>	Specifies the density ranges and corresponding colours that should be used to build the graph. In addition to the standard Fruin walkway, stairway and platform (queueing) metrics, custom density levels and colours can be defined.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph.

	This will be saved, but will not be exported to CSV.
--	--

4.3.4.4.8 Volume Density

Volume density graphs display the average density in one or more [volume](#) objects. Volume density is calculated based on the number of agents in the volume [area](#) divided by the total area of the volume. Each volume will have a corresponding series on the graph showing its average density over time, and the graph will have coloured horizontal background stripes indicating different density ranges.

Multiple simulation runs can be specified with an aggregate result shown.

Volume Density Parameters	
<a href="#">Simulation runs</a>	The simulation runs for which densities should be calculated.
<b>Aggregation type</b>	How density data should be aggregated over multiple runs. The graphed value can be either the minimum, maximum or average of the densities for that volume across each of the runs. These options are all equivalent if only a single run is chosen.
<b>Reporting period</b>	The time period over which densities should be calculated.
<b>Sampling period</b>	How frequently to measure each volume's population and calculate its density. Specifying a larger value will reduce the amount of time needed to generate the graph, but using very large values runs the risk of missing peaks in density that may occur between samples.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X Axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y Axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	Densities can be displayed as either curves, stepped curves, or bars. If stepped curves or bars are chosen, a bin size must be specified. Each bar will then correspond to an interval of time with the size specified, and the associated value will be the <b>average</b> density within that time interval. If sampling period exceeds bin size, zero values will be present.
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Volumes</b>	Which volumes to include in the graph. Each volume will correspond to one graph series.
<b>Density Ranges</b>	The colours and density ranges to use for the background reference bands on the graph. Any of the standard <a href="#">LOS Colour Mapping</a> types can be used, or custom ranges and colours can be specified.



<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.
--------------	---

## 4.3.4.4.9 Composite

Composite graphs allow data of different types and/or from different simulation runs to be compared. For instance, flow counts could be compared between two different simulation runs with slightly different configurations, or cumulative flow counts and population counts could be compared on the same graph.

The [tree structure](#) used to define composite graphs has a more complex structure than other graph types. In addition to filters, groups, and group member items, composite graph trees have separate items corresponding to different simulation runs and different graph types. As with other graph types, composite graphs are built top-down. Items are added in the following order:

1. Simulation run
2. Graph type ([cumulative flow count](#), [flow count](#), [population count](#), or [server population count](#))
3. Filter (optional)
4. Group
5. Group members

Multiple items can be added at each level; for instance, multiple graph types can be added under a single simulation run, and a graph type can have multiple groups or filtered groups beneath it.

Composite Graph Parameters	
<b>Reporting period</b>	The time period over which results should be calculated.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Style</b>	Composite graphs can be displayed as curves, stepped curves and bars. For stepped curves and bars, a bin size must be specified and values for each series on the graph will be aggregated within time intervals of the given size. Note that this will occur in whichever way is most appropriate for each series type; eg. series representing population or queue counts will be averaged within time intervals while flow counts will be summed. If sampling period exceeds bin size, zero values will be present.
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Graph</b>	The tree structure used to define the series to display in the graph.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the graph. This will be saved, but will not be exported to CSV.

4.3.4.4.10 Performance

Each frame of the simulation can be broken down into a series of steps. The times it takes to execute each of these steps is recorded and can be displayed using the performance graph.

Performance Table Parameters	
<b>Name</b>	The name of the table.
<a href="#">Simulation runs</a>	The simulation runs to be analysed
<b>Reporting period</b>	The time period over which counts should be calculated.
<b>Series: Metric</b>	How performance statistics should be listed.  <b>Duration (ms):</b> Time required to execute a frame. <b>Operations:</b> Operations involved in executing a frame. <b>1000 * Duration (ms) / Operations:</b> Time per operation. <b>1000 * Duration (ms) / Population:</b> Time per population.
<b>Series</b>	Performance statistics to display.  Serial components are executed in sequence by a single process. Speed of execution is based only on CPU speed or disk access time. Threaded components can be executed in parallel and take full advantage of all available CPUs.  <ul style="list-style-type: none"> <li>• <b>Entire Frame:</b> Time taken to execute the entire frame.</li> <li>• <b>Serial Components:</b> Time taken to execute the serial portions of the frame.</li> <li>• <b>Threaded Components:</b> Time taken to execute the parallel or threaded portions of the frame.</li> </ul> Information can also be displayed by individual frame components. Mouse over a component for a brief description of the component.
<b>Title</b>	Graph title (will be included in exported images and CSV files).
<b>X Axis</b>	X axis label (will be included in exported images and CSV files).
<b>Y Axis</b>	Y axis label (will be included in exported images but not CSV files).
<b>Show Legend</b>	Display graph legend (will be included in exported images but not CSV files). Graph must be regenerated to update dataset labels.
<b>Show Metadata</b>	Display information about the MassMotion version and database file used to generate the graph (will be included in exported images and CSV files).
<b>Notes</b>	A simple field that can be used to save comments or explanation about the table. This will be saved, but will not be exported to CSV.

#### 4.3.4.5 Maps

Map queries paint objects in the scene based on an analysis of the behaviour of agents.

All maps have a "Surface Objects" parameter which determines which objects will be painted. Most maps can only be drawn on walkable surfaces. Vision based maps can also be drawn on vertical barriers.

It is not possible to show more than one map on a surface at the same time. Evaluating or showing a map will hide any other maps on that object. Maps can be hidden from the view menu or by right clicking map objects in the list view.

When map colouring can be adjusted, transparent colours will allow the colour of the underlying object to show through the map. See [working with colours](#) for more information.

Maps cannot be directly exported but will be displayed in any images or videos produced through the [movie and image exporter](#).

##### List of Maps

Map Type	Description
<a href="#">Agent Count</a>	Counts the number of unique agents that have ever visited each point.
<a href="#">Agent Path</a>	Traces the route taken by each agent.
<a href="#">Dynamic Path</a>	Shows dynamic 'trails' behind agents as they move around the model. This map is "live" and updates with simulation playback.
<a href="#">Average Density</a>	The average density at each point.
<a href="#">Average Non-Zero Density</a>	The average non-zero density at each point.
<a href="#">Experienced Density</a>	The experienced density at each point.
<a href="#">Instantaneous Density</a>	The current density at each point. This map is "live" and updates with simulation playback.
<a href="#">Maximum Density</a>	The maximum density at each point.
<a href="#">Agent Time to Exit</a>	The time required for an agent to exit from each point.
<a href="#">Time Above Density</a>	The time agents spent above a specified density at each point.
<a href="#">Time Occupied</a>	The time for which each point was occupied by an agent.
<a href="#">Time Until Clear</a>	The time before the last agent left each point.
<a href="#">Vision Time</a>	The time agents spent looking at a surface. This map can paint barriers.

<a href="#">Vision Count</a>	Counts the number of agents that viewed a given point on an object. This map can paint barriers.
<a href="#">Vision Time Above Count</a>	The total time a given point on an object was viewed by more than the specified number of agents.
<a href="#">Static Cost</a>	The cost perceived by agents to reach various destination portals. This map does not require a simulation run.
<a href="#">Static Distance</a>	The measured distance to various destination portals. This map does not require a simulation run.

4.3.4.5.1 Counts

4.3.4.5.1.1 Agent Count

Agent count maps display the number of unique agents that visited each point on a surface within a given time range.

Agent Count Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<a href="#">Agent Filter</a>	Used to select a subset of agents to consider when generating the map.
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as agent counts. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.1.2 Agent Path

Agent path maps show where agents tend to walk. As agents move around the model, they will lay down a semi-transparent trail with the same colour as the agent itself. These paths will be laid down on top of one another, so later agents will tend to obscure earlier ones.

Agent path maps are very useful when constructing [filters](#), as they can provide immediate visual feedback on whether the filter is performing as expected.

Agent Path Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.  The agent colouring options set in the simulation run are used to colour agent trails when drawing the paths.
<b>Time range</b>	The time period over which paths should be generated.

<b>Agent Filter</b>	Used to select a subset of agents to consider when generating the map.
<b>Opacity</b>	The opacity of the path laid down by each agent, in percent.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.1.3 Dynamic Path

Dynamic path maps are similar to [agent path maps](#), but the trails fade over a set period of time. The net effect is that each agent can have (for instance) a trail showing where it has been in the last 10 seconds, which can be useful for visualization purposes.

A couple of techniques that can be used with dynamic path maps include:

- Setting the decay time to zero and opacity to a small value (10-20%): this will have the effect of adding a small circular 'shadow' under each agent.
- Setting up a dynamic path map and then hiding the corresponding simulation run: in this way only the paths themselves will be shown, which can be useful in visualizing overall flow patterns.

Dynamic path maps can be used when [exporting movies](#), but can only be effectively previewed during simulation playback. Manually dragging the playback slider will erase and reset all paths.

Dynamic Path Map Parameters	
<b>Simulation run</b>	The simulation run for which the map should be generated.  The agent colouring options set in the simulation run are used to colour agent trails when drawing the paths.
<b>Agent Filter</b>	Used to select a subset of agents to consider when generating the map.
<b>Initial Opacity</b>	The opacity of the path as initially laid down by an agent, in percent (i.e., the opacity of the path directly under the agent's feet).
<b>Decay Time</b>	Approximately how long each path will persist, in seconds. For instance, setting this to 10 seconds will mean that the trail behind each agent will remain visible for 10 seconds before fading away. Experimentation may be needed to find a combination of initial opacity and decay time that produces the desired visual effect.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.2 Density

For each density related map, the density at a point is calculated by drawing a circle with area  $3.25\text{m}^2$  around the point, counting the number of agents inside the circle, then dividing by that area.

During the simulation, the area used in density calculations is an accurate measure of the space available to an agent. With maps, the full circle area is always used regardless of obstacles or floor

edges. This means that map density near barriers or floor edges can be under reported and differ from the density recorded during simulation.

4.3.4.5.2.1 Average Density

Average density maps display what parts of an object were, on average, most crowded. The colour at each point will indicate the average density (agents/m<sup>2</sup>) at that point over the given time range.

Density calculations are based on an approximation of the available area and so can underestimate the density near obstacles or floor edges. See [map density](#) for information on how map density is calculated.

The average density is defined as:

$$LOS(t) = \frac{\sum_{n=1}^t density(n)}{t}$$

Average Density Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy.
<b>Colouring</b>	The colours that will be used in the map. Any of the standard Fruin metrics plus the IATA (International Air Transport Association) Wait/Circulate standard can be used. If 'Fruin (auto)' is used, Fruin walkway values will be used for floors and links, and Fruin stairway values will be used for stairs and escalators. Finally, custom density ranges and corresponding colours can also be defined. See <a href="#">working with colours</a> and <a href="#">LOS colour mapping</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.2.2 Average Non-Zero Density

Average non-zero density maps are the same as [average density](#) maps except that they do not include any times for which the density was zero. This is useful when dealing with traffic that occurs in bursts. It will discount the time between bursts and provide the average density only for periods when agents were present.

Density calculations are based on an approximation of the available area and so can underestimate the density near obstacles or floor edges. See [map density](#) for information on how map density is calculated.

**Average Non-Zero Density Map Parameters**

<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy.
<b>Colouring</b>	The colours that will be used in the map. Any of the standard Fruin metrics plus the IATA (International Air Transport Association) Wait/Circulate standard can be used. If 'Fruin (auto)' is used, Fruin walkway values will be used for floors and links, and Fruin stairway values will be used for stairs and escalators. Finally, custom density ranges and corresponding colours can also be defined. See <a href="#">working with colours</a> and <a href="#">LOS colour mapping</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.2.3 Experienced Density

Experienced density maps show the average density experienced by agents (the average of all experiences) around each point, computed as a weighted average. The measure is calculated as:

$$LOS(t) = \frac{\sum_{n=1}^t density(n)^2}{\sum_{n=1}^t density(n)}$$

While the regular time based [average density](#) tends to smooth out dense but intermittent bursts in traffic, the experienced average highlights those bursts regardless of their frequency.

Density calculations are based on an approximation of the available area and so can underestimate the density near obstacles or floor edges. See [map density](#) for information on how map density is calculated.

Experienced Density Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy.
<b>Colouring</b>	The colours that will be used in the map. Any of the standard Fruin metrics plus the IATA (International Air Transport Association) Wait/Circulate standard can be used. If 'Fruin (auto)' is used, Fruin walkway values will be used for floors and links, and Fruin stairway values will be used for stairs and escalators. Finally, custom density ranges and corresponding colours can also be defined. See <a href="#">working with colours</a> and <a href="#">LOS colour mapping</a> for more information.

<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.2.4 Instantaneous Density

Instantaneous density maps can be used to produce a live, animated display of what parts of one or more objects are most crowded. The colour at each point will indicate the current density (agents/ m<sup>2</sup>) at that point.

Density calculations are based on an approximation of the available area and so can underestimate the density near obstacles or floor edges. See [map density](#) for information on how map density is calculated.

Average Density Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Colouring</b>	The colours that will be used in the map. Any of the standard Fruin metrics plus the IATA (International Air Transport Association) Wait/Circulate standard can be used. If 'Fruin (auto)' is used, Fruin walkway values will be used for floors and links, and Fruin stairway values will be used for stairs and escalators. Finally, custom density ranges and corresponding colours can also be defined. See <a href="#">working with colours</a> and <a href="#">LOS colour mapping</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.2.5 Maximum Density

Maximum density maps show the maximum density reached at every point during the given time range.

Density calculations are based on an approximation of the available area and so can underestimate the density near obstacles or floor edges. See [map density](#) for information on how map density is calculated.

Maximum Density Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy (specifically, density peaks will be missed if they fall in between sample times).



<b>Colouring</b>	The colours that will be used in the map. Any of the standard Fruin metrics plus the IATA (International Air Transport Association) Wait/Circulate standard can be used. If 'Fruin (auto)' is used, Fruin walkway values will be used for floors and links, and Fruin stairway values will be used for stairs and escalators. Finally, custom density ranges and corresponding colours can also be defined. See <a href="#">working with colours</a> and <a href="#">LOS colour mapping</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.3 Times

##### 4.3.4.5.3.1 Agent Time To Exit

Agent time to exit maps display, for each point, the longest time it took an agent to evacuate the simulation from that point. In an evacuation scenario, points near the simulation exit will have short duration values while points far from the exit will have large duration values.

<b>Agent Time To Exit Map Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<a href="#">Agent Filter</a>	Used to select a subset of agents to consider when generating the map. When in use, evacuation times will only be considered for agents that pass the filter.
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

##### 4.3.4.5.3.2 Time Above Density

Time above density maps display the amount of time each point on the map spent above a specified density threshold.

<b>Time Above Density Map Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Minimum Density</b>	The density threshold. Time at a point will only be counted when density at that point is at or above the specified value. Standard Fruin walkway values can be selected, or a custom density value can be specified.

<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.3.3 Time Occupied

Time occupied maps display the amount of time each point in the map was in use by any agent in the simulation. The colour at each point will indicate the cumulative count (in seconds) that the point was under an agent. For instance, in train station scenario, a time occupied map might be used to determine how much time people spent waiting near the departures board in the main concourse.

Time Until Clear Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed. Time values will be computed from the start of the time range.
<a href="#">Agent Filter</a>	Used to select a subset of agents to consider when generating the map. When in use, a point will only be considered occupied when it is under an agent that passes the filter.
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">Working with Colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

4.3.4.5.3.4 Time Until Clear

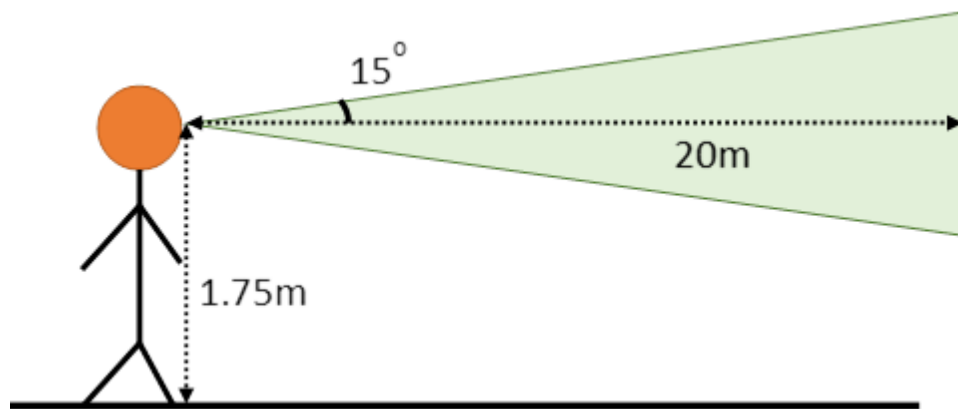
Time until clear maps display how long it took for the last agent to leave a space. The colour at each point will indicate the last time, relative to the beginning of the given time range, that any agent stood at that point. In an evacuation scenario points far away from the exits would have small values since those areas would be vacated quickly. Points near the exits would have large values since those points would continue to be visited by agents as they exited.

Time Until Clear Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed. Time values will be computed from the start of the time range.
<a href="#">Agent Filter</a>	Used to select a subset of agents to consider when generating the map. When in use, only agents that pass the filter will be considered when determining the last time an area is used.

<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.4 Vision

Vision maps can be used to qualitatively assess where agents look as they move through a simulation. Vision maps work by converting objects such as floors and walls to 'voxels' (small cubes). As agents move, they project a cone of vision ahead of them and mark which voxels they can see within that cone. The cone defaults to a 30 degree field of view with a cutoff viewing distance of 20 meters but both parameters can be customized to increase or reduce the area visible to each agent. Only those objects that intersect with the vision cone will be marked as "seen".



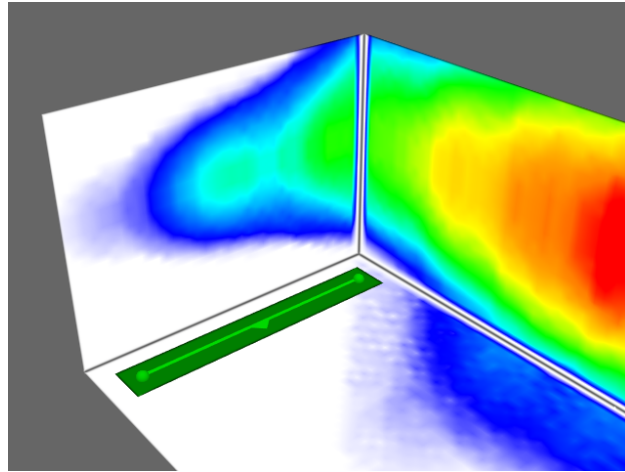
Agents will always "see" through [cordons](#), [volumes](#), and any other agents but their vision can be occluded by other scene objects depending on the way in which the map is configured.

#### Performance

Vision maps are very resource intensive, consuming significant CPU time and memory. Changing the sampling period is an effective way of speeding up the map computation. Memory usage can be reduced by choosing a smaller set of surface objects. Reducing the cutoff distance or field of view can reduce computation though this will also reduce the number of objects visible to each agent.

#### Artifacts

Note that the voxel based approach can cause certain visual artifacts as voxels obscure each other:



**Figure 1: Voxels in corners have no set values because they are blocked by adjacent voxels.**

4.3.4.5.4.1 Vision Time

Vision time maps display the cumulative time agents spent viewing an object. If each agent generates a record when it is sampled and deemed to be viewing an object, the value displayed at a point is the sum of all agent records collected for that point. Therefore 2 agents viewing a point for 3 seconds and 1 agent viewing the same point for 6 seconds would both yield a value of 6 for that point.

See [vision map](#) for a discussion of issues common to all vision maps.

Vision Time Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy.
<a href="#">Agent Filter</a>	Which agents should be included when generating the map.
<b>Cutoff distance</b>	How far agents can see (the length of their cone of vision). Objects beyond the cutoff distance will not be marked as "seen".
<b>Field of view</b>	The angle of the agents' cone of vision. This is the full angle of the cone, not the half-angle.
<b>Colouring</b>	The colours that will be used in the map. The default values may need to be changed as value ranges change dramatically based on the population counts and characteristics. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to. Unlike most maps, vision maps can include barriers and visual objects as surface objects.  The amount of memory used during vision map computation depends on the size of a rectangular box that fits around all selected objects. Therefore, selecting a

	very large object or even two small objects that are far apart will require a large amount of memory.
<b>Occluding Objects</b>	Which objects (in addition to the chosen surface objects) should occlude (block) agent vision: all visible and hidden, all visible, none, or a specified set of objects.
<b>Notes</b>	User comments about the map.

#### 4.3.4.5.4.2 Vision Count

Vision count maps display how many agents view a given point on an object.

If 'Count agents once' is selected, then the value displayed at any point corresponds to the number of unique agents who ever viewed that point. If 'Count agents once' is not selected then each agent is counted once per sample that it is found to be viewing the point. For a sampling period of 1 second, this will give identical results to the [vision time](#) map.

See [vision map](#) for a discussion of issues common to all vision maps.

Vision Time Map Parameters	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy. Specifying a larger value will also reduce the measured values if not using 'Count agents once' since there will be a smaller number of total samples taken.
<b>Count agents once</b>	If checked, each agent will only be counted once.
<a href="#">Agent Filter</a>	Which agents should be included when generating the map
<b>Cutoff Distance</b>	How far agents can see (the length of their cone of vision). Objects beyond the cutoff distance will not be marked as "seen".
<b>Field of View</b>	The angle of the agents' cone of vision. This is the full angle of the cone, not the half-angle.
<b>Colouring</b>	The colours that will be used in the map. The default values may need to be changed as value ranges change dramatically based on the population counts and characteristics. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to. Unlike most maps, vision maps can also include barriers or visual objects.  The amount of memory used during vision map computation depends on the size of a rectangular box that fits around all selected objects. Therefore, selecting a very large object or even two small objects that are far apart will require a large amount of memory.

<b>Occluding Objects</b>	Which objects (in addition to the chosen surface objects) should occlude (block) agent vision: all visible and hidden, all visible, none, or a specified set of objects.
<b>Notes</b>	User comments about the map.

4.3.4.5.4.3 Vision Time Above Count

Vision time above count maps display the amount of time a given point on an object spent being viewed by at least a certain number of agents (that is, how long that point spent being looked at by at least the given number of agents simultaneously).

See [vision map](#) for a discussion of issues common to all vision maps.

<b>Vision Time Map Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which the map should be generated.
<b>Time range</b>	The time period over which map values should be computed.
<b>Sampling period</b>	How frequently to sample agent positions when generating the map. Specifying a larger value will reduce the amount of time needed to generate the map, but decrease accuracy.
<b>Minimum viewers</b>	The minimum number of agents simultaneously looking at a point for that point to be counted as 'viewed'.
<a href="#">Agent Filter</a>	Which agents should be included when generating the map.
<b>Cutoff Distance</b>	How far agents can see (the length of their cone of vision). Objects beyond the cutoff distance will not be marked as "seen".
<b>Field of View</b>	The angle of the agents' cone of vision. This is the full angle of the cone, not the half-angle.
<b>Colouring</b>	The colours that will be used in the map. The default values may need to be changed as value ranges change dramatically based on the population counts and characteristics. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to. Unlike most maps, vision maps can also include barriers or visual objects.  The amount of memory used during vision map computation depends on the size of a rectangular box that fits around all selected objects. Therefore, selecting a very large object or even two small objects that are far apart will require a large amount of memory.
<b>Occluding Objects</b>	Which objects (in addition to the chosen surface objects) should occlude (block) agent vision: all visible and hidden, all visible, none, or a specified set of objects.
<b>Notes</b>	User comments about the map.

## 4.3.4.5.5 Network

## 4.3.4.5.5.1 Static Cost

Static cost maps visualize the network agents use when choosing a route through the scene. Values are based on static costs like horizontal and vertical distance, but do not include dynamic costs like queuing or gate penalties. Cost maps can be combined with [network](#) objects to help visualize the routes available in a given network.

**Evaluating this map rebuilds the scene network which can be time consuming and the evaluate command cannot be canceled.**

The value shown on the map at any point is the estimated time taken for an agent to get **from** that point **to** the nearest of the given portals. This will be affected by distance penalties and directionality on links, stairs, paths etc. If there is no route from a point to the given portals that point will be coloured white.

Vertical costs of vertical elements such as [stairs](#) or [escalators](#) and other cost penalties are applied as a single step before the next object which may lead to visual discontinuities. Within an object, only horizontal costs are considered. Costs are calculated using a nominal agent speed of 1 m/s.

Labels can be used to show numerical values at goal lines.

Static Cost Map Parameters	
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Portals</b>	The map will show the cost used by agents to arrive at any of the given portals. With multiple portals, the lowest cost to any of the portals will be displayed for each point.
<b>Network Objects</b>	The objects included in the network calculations. The map includes the entire model by default. By specifying a network object or a subset of objects the map can visualize constrained networks where excluded objects are unavailable to agents. This can be useful when measuring the lowest cost to a portal without using stairs.
<b>Labels</b>	Displays numerical values as text directly over the objects in the scene.  <b>None:</b> No values shown. <b>Cost at transitions:</b> Show the lowest cost to goal from each transition (connection between objects). The value could represent the cost of moving either onto or off of the floor, whichever is lower. <b>Cost of choices from floors:</b> Show the lowest cost to goal for each choice leading off of a floor. The value always represents the cost if moving from the floor across the link/stair/ramp/escalator/path to the goal and so represent the value used by an agent when choosing routes during <a href="#">agent navigation</a> . No value is displayed when a choice is not a valid route to the goal.
<b>Notes</b>	User comments about the map.

4.3.4.5.2 Static Distance

Static distance maps show the distance from each point to the given portals. Distance maps can be combined with [network](#) objects to help visualize the routes available in a given network

**Evaluating this map rebuilds the scene network which can be time consuming and the evaluate command cannot be cancelled.**

The value shown on the map at any point is the walking distance **from** that point **to** the nearest of the given portals. This will be affected by directionality on links, stairs, paths etc. but will not include object distance penalties (distance penalties are included in the [static cost](#) map). If there is no route from a point to the given portals that point will be coloured white.

Distances along vertical elements such as [stairs](#) or [escalators](#) are taken as the Cartesian distance along their lengths. For other objects, only horizontal distance are considered. Any height variation on a floor or link is ignored.

Static Distance Map Parameters	
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as distances in metres. See <a href="#">working with colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Portals</b>	The map will show the distance from any of the given portals. With multiple portals, the shortest distance to any portal will be displayed for each point.
<b>Network Objects</b>	The objects included in the network calculations. The map includes the entire model by default. By specifying a network object or a subset of objects the map can visualize constrained networks where excluded objects are unavailable to agents. This can be useful when measuring the shortest distance to a portal without using stairs.
<b>Labels</b>	Displays numerical values as text directly over the objects in the scene.  <b>None:</b> No values shown. <b>Distance at transitions:</b> Show the shortest distance to goal from each transition (connection between objects). The value could represent the distance to goal from the direction onto or off of the floor, whichever is shorter. <b>Distance for choices from floors:</b> Show the shortest distance to goal for each choice leading off of a floor. The value always represents the distance from the floor across the link/stair/ramp/escalator/path to the goal. No value is displayed when a choice is not a valid route to the goal.
<b>Notes</b>	User comments about the map.

4.3.4.6 Tables

Table queries extract data from one or more simulation runs and present it in tabular form. Once generated, tables can be exported to a csv file.

Histograms of the values in some columns can be generated by right-clicking on the column header. Some columns support selecting objects, focusing on a time, or focusing on an agent at a time, all by right-clicking on a row within the column.



**List of Tables**

Table Type	Description
<a href="#">Agent Area Time</a>	Lists the amount of time agents spend in given areas.
<a href="#">Agent LOS Time</a>	Lists the amount of time agents spend at each level of service.
<a href="#">Agent Process Chain Time</a>	Lists the amount of time agents spend in a process chain.
<a href="#">Agent Social Cost</a>	Lists agent journey times expressed as a generalized time or cost.
<a href="#">Agent Summary</a>	Provides general information about each agent.
<a href="#">Agent Timetable Summary</a>	Provides general information about agents produced by a given timetable.
<a href="#">Agent Token Time</a>	Lists the amount of time agents carry given tokens.
<a href="#">Agent Transition</a>	Provides a record of each time an agent crosses a transition.
<a href="#">Agent Trip Time</a>	Lists the time required for agents to complete a given trip.
<a href="#">Area OD Count Matrix</a>	Counts the number of agents entering and exiting a zone or floor organized by entrance and exit object.
<a href="#">Expected Demand OD Count Matrix</a>	Counts the anticipated number of agents that an event will produce.
<a href="#">Simulation OD Count Matrix</a>	Counts the number of agents entering and exiting the simulation organized by entrance and exit portal.
<a href="#">Simulation OD Time Matrix</a>	Counts the time spent in the simulation organized by entrance and exit portal.
<a href="#">Simulation OD Social Cost Matrix</a>	Counts the generalized time or cost for agents organized by entrance and exit portal.
<a href="#">Server Summary</a>	Lists queue times and other information for servers.
<a href="#">Performance Table</a>	Provides diagnostics for simulation runs.

## 4.3.4.6.1 Agent Area Time

Agent area time tables display how long agents spend in different [areas](#).

Agent Area Time Parameters	
<a href="#">Simulation run</a>	The simulation run for which area times should be calculated.
<a href="#">Time range</a>	The time period over which times should be calculated.

<b>Agent Filter</b>	Determines which agents in a given frame can be included in the time count. When the filter is in use, the area time is the sum of all frames for which the agent is both in the area and satisfying the filter.
<b>Areas</b>	Which areas should be included in the table.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent. Agents will only be included in the table if they satisfy the given filter during the given time range, and are ever in any of the given areas during that time range.

Agent Area Time Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Names (variable)</b>	One column per selected <a href="#">area</a> . Each column shows how long the agent spent in that area while simultaneously satisfying the given filter.

4.3.4.6.2 Agent LOS Time

Agent LOS time tables display how long different agents spend at different levels of service.

Agent LOS Time Parameters	
<b><a href="#">Simulation run</a></b>	The simulation run for which LOS times should be calculated.
<b>Time range</b>	The time period over which times should be calculated.
<b>Agent Filter</b>	Determines which agents in a given frame can be included in the time count. When the filter is in use, the LOS time is the sum of all frames for which the agent is both at the given LOS and satisfying the filter.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent. Agents will only be included in the table if they ever satisfy the given filter during the given time range. All times are rounded to the nearest second.

Agent LOS Time Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Total Duration</b>	Total amount of time that the agent satisfied the given filter. Note that this number may be less than expected if the agent entered or exited the simulation during the specified time range.
<b>LOS [A,B,C,D,E,F] Duration</b>	Total amount of time that the agent spent at each level of service while simultaneously satisfying the given filter. For instance, using an 'in area' filter with a particular floor, each row will show how long one agent spent at each level of service while on that floor. The level of service is calculated based on standard Fruin values, with the LOS type determined by what sort of walkable object the agent is standing on. Floors and links use Fruin walkway LOS values, and stairs

and escalators use Fruin stairway LOS values.
---

#### 4.3.4.6.3 Agent Process Chain Time

Agent process chain time tables display summary information about an agent's use of a given process chain.

Agent Process Chain Time Parameters	
<a href="#">Simulation run</a>	The simulation run for which process chain times should be calculated.
<b>Finished end server at</b>	A special type of time range: only agents that <b>left</b> the final server in the process during the given time range will be included in the table. Agents that started the process before the start of the interval but finished during the interval will be included. Conversely, agents that started the process during the given time range but did not finish until afterward will not be included.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table.
<b>Servers</b>	Specifies lists of start and end servers defining the process chain. An agent is defined as starting the process chain when it starts the pre-contact wait stage at any of the first servers (see <a href="#">server</a> for details on different server stages), and is defined as finishing the process chain when it exits any of the end servers. It is possible to specify the same server(s) as both start and end.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent's processing time. Agents will only be included in the table if they satisfy the given filter during the given time range. All times are rounded to the nearest second.

Agent Process Chain Time Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Start Server</b>	Which of the specified start servers the agent started the process chain at.
<b>End Server</b>	Which of the specified end servers the agent ended the process chain at.
<b>Start Time</b>	What time the agent started the process chain. This is the first time when they are either being processed by one of the start servers or are queuing for the server (blocked by another agent that is at the server).
<b>End Time</b>	What time the agent ended the process chain. This corresponds to the time at which the agent was finished being processed by one of the end servers.
<b>Total Duration</b>	How long the agent took to finish the process chain (end time minus start time).
<b>In Transit</b>	Total amount of time the agent spent moving between servers. This includes both the time spent moving to a server line and any time spent unobstructed along the

	line.
<b>Pre-Contact Wait</b>	Total amount of time the agent spent waiting in any server input buffers before being processed..
<b>Contact Wait</b>	Total amount of time the agent spent in contact with and being processed by servers.
<b>Post-Contact Wait</b>	Total amount of time the agent spent waiting in server output buffers for a space to become available at a downstream server.

4.3.4.6.4 Agent Social Cost

The agent social cost table displays information about the journey of each agent, expressed as a weighted time or cost value. The default weights, cost factors, and algorithms are taken from the Transport For London Business Case Development Manual [1].

In general an agent journey is broken down into the time spent on various activities such as walking, waiting, or climbing stairs. A weight factor is applied to each of these component activities. The sum of the weighted components is termed the 'Generalized Journey Time' and is expressed in seconds.

An additional congestion penalty is calculated while the agent is walking or waiting. The sum of the congestion penalty and generalized journey time is used to calculate a total cost. If the total cost is assumed to be the cost for one day, an annualized cost can be calculated using the number of days in the year.

**Congestion Factor**

Density P (ppl/m <sup>2</sup> )	Congestion Factor Walking	Congestion Factor Waiting
P <= 0.5	0.0	0.0
P >= 2.0	0.5 * 1.5 = 0.75	1.5
0.5 < P < 2.0	0.5 * 0.667 * ( P - 0.5 ) <sup>2</sup>	0.667 * ( P - 0.5 ) <sup>2</sup>

JT stands for journey time and is the measured duration an agent spent engaged in the given activity. GJT is the generalized journey time and is the journey time multiplied by the activity weight. CF is the congestion factor as described in the table above.

<b>Agent Social Cost Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which social cost values should be calculated.
<b>Time range</b>	The time period over which times should be calculated.
<b>View</b>	<p><b>Activity Summary:</b> Display the time, generalized time, and cost values summed over all agents, broken down into the component activity types.</p> <p><b>All agent data:</b> Display the time, generalized time, and cost values for each agent.</p>

	<p><b>Agent costs:</b> Display cost values for each agent.</p> <p><b>Agent generalized journey times:</b> Display generalized journey times and congestion factors for each agent.</p> <p><b>Agent journey times:</b> Display journey times for each agent.</p>
<b>Agent Filter</b>	Determines which agents in a given frame can be included in the time count.
<b>Cost per Hour</b>	The price used to convert the generalized journey time to a cost value. The cost is first converted into a 'cost per second' value, then multiplied by the generalized journey times.
<b>Days per Year</b>	The number of days in a year. It is assumed that calculated costs are for one day. These day cost values are multiplied by the number of days in a year to produce the annualized cost.
<b>Walking Weight</b>	A factor applied to time spent walking on floors or ramps.
<b>Waiting Weight</b>	A factor applied to time spent waiting. This time includes waiting for a closed gate to open or waiting in response to a wait task. A wait task can be given by the creation event (evacuation event, circulation event) or given by an action.
<b>Queuing Weight</b>	A factor applied to time spent queuing at a server. This does not apply to agents queuing at a link, stair, ramp, or escalator.
<b>Processing Weight</b>	A factor applied to time spent being processed by a server.
<b>Stairs Up Weight</b>	A factor applied to time spent walking up stairs.
<b>Stairs Down Weight</b>	A factor applied to time spent walking down stairs.
<b>Escalator Weight</b>	A factor applied to time spent riding on escalators (up or down).
<b>Custom Weight</b>	<p>A custom weight that can be applied to agents based on an agent filter. When enabled, agents that pass the filter in a given frame will use the custom weight for that frame. Only agents that do not pass the filter will use the other activity weights such as walking or waiting.</p> <p>This can be useful for applying a custom weight to a specific activity in a specific area of the model. Or by setting the weight to 0, it can also be used to exclude agents from being counted under certain conditions.</p>
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent. Agents will only be included in the table if they ever satisfy the given filter during the given time range. All times are rounded to the nearest second.

[1] Business Case Development Manual, Transport For London, May 2013, Appendix E 3.1

4.3.4.6.5 Agent Summary

Agent summary tables display a variety of overall summary information for a set of agents. When combined with the agent filters the table becomes a powerful way to validate components of a simulation.

For example:

- Right-click on the Entrance column header to display a histogram of the number of agents entering through each portal.
- Right-click on the Desired Speed column header to display a histogram of the number of agents in various speed ranges.
- Right-click on an agent Start Time value to focus the 3D view on the agent at the time it entered the simulation.
- Note an agent's Agent ID value and use that in the [agent observer](#) window to view more detailed information about the agent.

Agent Summary Parameters	
<a href="#">Simulation run</a>	The simulation run for which agent summaries should be calculated.
<b>Agents alive at</b>	A special type of time range: only agents that were alive at any point during the given interval will be included in the table. Values in the table (duration, distance traveled etc.) will still refer to the entire lifetime of each agent.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the table. This will be saved, but will not be exported to CSV.

Each row of the table has information on one agent. Agents will only be included in the table if they satisfy the given filter during the given time range. All times are rounded to the nearest second.

Agent Summary Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Profile</b>	The profile assigned to the agent when it was created.
<b>Creator</b>	The event which created the agent.
<b>Entrance</b>	What portal the agent entered the simulation at.
<b>Exit</b>	What portal the agent exited the simulation at. If the field is blank, the agent was still in the scene when the simulation ended.
<b>Start Time</b>	What time the agent entered the simulation.
<b>End Time</b>	What time the agent exited the simulation. Agents still in the scene when the simulation ended will list the simulation end time.
<b>Duration</b>	Total amount of time the agent spent in the simulation (end time minus start time).

<b>Distance Traveled (m)</b>	Total distance traveled by the agent. This includes any stuttering back and forth when congested.
<b>Desired Speed (m/s)</b>	Innate desired speed of the agent as designated by the agent's <a href="#">profile</a> .
<b>End State</b>	The end state of the agent after leaving the simulation or when the simulation ends.  <b>in simulation:</b> still in scene at simulation end <b>exited with success:</b> exited simulation as expected <b>exited with error:</b> was deleted from simulation with an error.

#### 4.3.4.6.6 Agent Timetable Summary

Agent timetable summary tables display information about agents created by a particular [timetable](#). It is possible to display all agents created by the timetable, or only those agents created based on one of the timetable reference events. For example, if the timetable is describing the operations of an airport and reference events correspond to flights, the table can be made to display information about all those agents who were either arriving or departing on a particular flight.

Agent Timetable Summary Parameters	
<a href="#">Simulation run</a>	The simulation run for which agent summaries should be calculated.
<b>Agents alive at</b>	A special type of time range: only agents that were alive at any point during the given interval will be included in the table. Values in the table (duration, distance traveled etc.) will still refer to the entire lifetime of each agent.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table.
<b>Timetable</b>	The timetable object of interest. Only agents created by this timetable will be included in the table.
<b>Reference events</b>	Optionally provide a list of timetable reference events. If used, only agents that were produced by or sent to any of the given reference events will be included in the table.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent. Agents will only be included in the table if they satisfy the given filter during the given time range. All times are rounded to the nearest second.

Agent Timetable Summary Columns	
<b>Agent ID</b>	Internal agent ID.
<b>From Reference Event</b>	Reference event from which the agent was created (blank if the timetable schedule did not specify a 'From' reference event).

<b>To Reference Event</b>	Reference event the agent was sent to (blank if the timetable schedule did not specify a 'To' reference event).
<b>Entrance</b>	What portal the agent entered the simulation at.
<b>Exit</b>	What portal the agent exited the simulation at. If the field is blank, the agent was still in the scene when the simulation ended.
<b>Start Time</b>	What time the agent entered the simulation.
<b>End Time</b>	What time the agent exited the simulation. Agents still in the scene when the simulation ended will list the simulation end time.
<b>Duration</b>	Total amount of time the agent spent in the simulation (end time minus start time).
<b>Distance Traveled (m)</b>	Total distance traveled by the agent.
<b>Desired Speed (m/s)</b>	Innate desired speed of the agent as designated by the agent's <a href="#">profile</a> .

4.3.4.6.7 Agent Token Time

Agent token time tables display how long different agents spent holding different [tokens](#).

<b>Agent Token Time Parameters</b>	
<a href="#">Simulation run</a>	The simulation run for which token times should be calculated.
<b>Time range</b>	The time period over which token times should be calculated.
<a href="#">Agent Filter</a>	Determines which agents in a given frame can be included in the time count. When the filter is in use, the token time is the sum of all frames for which the agent is both holding the token and satisfying the filter.
<b>Tokens</b>	Which <a href="#">tokens</a> should be included in the table.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent. Agents will only be included in the table if they satisfy the given filter during the given time range, and ever possess any of the given tokens during that time range. All times are rounded to the nearest second.

<b>Agent Token Time Columns</b>	
<b>Agent ID</b>	Internal agent ID.
<b>Names (variable)</b>	One column per selected token, showing how long the agent spent holding that token while simultaneously satisfying the given filter. For instance, using an 'in



trip' filter will result in each row showing the amount of time one agent spent holding various tokens while undergoing a particular [trip](#).

#### 4.3.4.6.8 Agent Transition

The agent transition table generates a record of each time an agent executes a given [transition](#). This can be useful for looking at how many times the same agent crosses between two objects or enters or exits an area.

Agent Transition Parameters	
<a href="#">Simulation run</a>	The simulation run for which transitions should be calculated.
<b>Time range</b>	The period over which transitions should be reported.
<a href="#">Agent Filter</a>	Determines which agents in a given frame can be included when tracking transitions. When in use, only agents crossing the transition while satisfying the filter will be included.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one occurrence of the transition. If the same agent executed the transition more than once, there will be a row for each occurrence. All times are rounded to the nearest second.

Agent Transition Columns	
<b>Agent ID</b>	The internal ID of the agent that executed the transition.
<b>Transition Time</b>	The time at which the transition occurred.
<b>Iteration</b>	If the transition was executed multiple times by this agent, identify this iteration as the 1st, 2nd, 3rd, etc.
<b>Total Count</b>	The total number of times the transition was executed by this agent.

#### 4.3.4.6.9 Agent Trip Time

Agent trip time tables can be used to determine how long agents spent to complete a certain [trip](#).

Agent Trip Time Parameters	
<a href="#">Simulation run</a>	The simulation run for which trip times should be calculated.
<b>Time range</b>	The time period over which times should be calculated.
<a href="#">Agent Filter</a>	Determines which agents in a given frame can be included in the time count. When the filter is in use, the trip time is the sum of all frames for which the agent is both in the trip and satisfying the filter.

<b>Trip</b>	The trip the table will refer to.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one agent's trip. Agents will only be included in the table if they satisfy the given filter during the given time range, and fully complete the trip within the specified time range. See [Trips](#) for the different ways in which a trip can be defined in terms of start and end criteria. All times are rounded to the nearest second.

Agent Trip Time Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Start Time</b>	When the agent started the trip.
<b>End Time</b>	When the agent finished the trip.
<b>Duration</b>	Total time the agent spent in the trip (end time minus start time).

4.3.4.6.10 Area OD Count Matrix

The origin/destination matrix displays a count of agents organized by the objects through which they enter and exit an [area](#). Possible entry and exit objects include portals, links, stairs, ramps, escalators, paths, or elevators. A time range can be used to target a particular interval within the simulation. Results can be displayed in matrix or list form.

Area entrances are listed as row headers down the left hand side. Area exits are listed as column headers along the top.

Origin/Destination Parameters	
<b><a href="#">Simulation run</a></b>	The simulation run for which counts should be calculated.
<b>Count if:</b>	<p><b>Agent starts in range:</b> Count/list agents who enter the area during the specified time range.</p> <p><b>Agent ends in range:</b> Count/list agents who exit the area during the specified time range.</p> <p><b>Agent starts and ends in range:</b> Count/list agents who both enter and exit the area completely within the specified time range.</p> <p><b>Agent exists during range:</b> Count/list agents who are present in the area during the specified time range.</p>
<b>Time range</b>	The time period used to determine which agents to count/list.
<b>View as</b>	<p>Whether to show results as a matrix or list.</p> <p><b>Matrix:</b> Each cell represents the total number of agents that entered and exited the zone/floor using the given object pair.</p> <p><b>List:</b> Each row is a single agent, with columns displaying the entrance portal and time and the exit portal and time.</p>

<b>Hide empty rows and columns</b>	When checked, rows and columns that contain only zeros are hidden.
<b><a href="#">Include Agents</a></b>	An agent filter that determines which agents will be included in the table.
<b>Target</b>	<p><b>Zone boundary objects:</b> All portals, links, stairs, ramps, escalators, paths, and elevators leading into and out of the specified zone are used. Agents are counted based on the object through which they enter the zone and the object through which they exit the zone.</p> <p><b>Floor boundary objects:</b> All portals, links, stairs, ramps, escalators, paths, and elevators leading onto and off of the specified floor are used. Agents are counted based on the object through which they enter the floor and the object through which they exit the floor.</p>
<b>Notes</b>	User comments about the table.

#### 4.3.4.6.11 Expected Demand OD Count Matrix

The query displays the number of agents the chosen events will create, grouped by the expected assigned origins and destinations. It is assumed that each event will execute a single time, and execute entirely to completion. Simulation time is ignored.

Only [journey](#), [evacuate](#), and [circulate](#) events are supported.

Origin/Destination Parameters	
<b>View as</b>	<p>Whether to show results as a matrix or list.</p> <p><b>Matrix:</b> Each cell represents the total number of agents that will be created at the origin and sent to the destination. Entrance portals are listed as row headers. Exit portals are listed as column headers.</p> <p><b>List:</b> Each row lists the count of agents that will be created for a single event between a single origin/destination pair.</p>
<b>Hide empty rows and columns</b>	When checked, rows and columns that contain only zeros are hidden.
<b>Expand collections</b>	By default collections are listed as origins and destinations and their counts are the sum of all counts for the contained portals. If checked, collections are not listed directly in the table by are expanded to create a single unified list of portals.
<b>Notes</b>	User comments about the table.

4.3.4.6.12 Simulation OD Count Matrix

The origin/destination count matrix displays a count of agents by their entrance and exit portals. A time range can be used to target a particular interval within the simulation. Results can be displayed in matrix or list form.

Entrance portals are listed as row headers. Exit portals are listed as column headers.

Origin/Destination Parameters	
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated.
<b>Count if:</b>	<p><b>Agent starts in range:</b> Count/list agents who enter the simulation during the specified time range.</p> <p><b>Agent ends in range:</b> Count/list agents who exit the simulation during the specified time range.</p> <p><b>Agent starts and ends in range:</b> Count/list agents who both enter and exit the simulation completely within the specified time range.</p> <p><b>Agent exists during range:</b> Count/list agents who are present in the simulation during the specified time range.</p>
<b>Time range</b>	The time period used to determine which agents to count/list.
<b>View as</b>	<p>Whether to show results as a matrix or list.</p> <p><b>Matrix:</b> Each cell represents the total number of agents that completed the trip starting at the origin portal and exiting the simulation at the exit portal.</p> <p><b>List:</b> Each row is a single agent, with columns displaying the entrance portal and time and the exit portal and time.</p>
<b>Hide empty rows and columns</b>	When checked, rows and columns that contain only zeros are hidden.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table.
<b>Target</b>	<p><b>All portals in model:</b> Agents entering or exiting from any portal in the scene are included. Portals are listed in alphabetical order.</p> <p><b>Specified portals:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in alphabetical order.</p> <p><b>Specified portals in order:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in the order specified.</p>
<b>Notes</b>	User comments about the table.

## 4.3.4.6.13 Simulation OD Time Matrix

The origin/destination time matrix displays the number of seconds agents were in the simulation, organized by entrance/exit. Results can be displayed as the total time for all agents with the given entrance/exit pair, the average trip time, the maximum, or the minimum. When using an agent filter, each agent's time count includes only those frames when the agent satisfied the filter.

Entrance portals are listed as row headers. Exit portals are listed as column headers.

Origin/Destination Parameters	
<a href="#">Simulation run</a>	The simulation run for which time values should be calculated.
<b>Count if:</b>	<p>This can be used to specify which agents to include based on whether or not they enter or exit the simulation within a specified time range. If an agent is included in the count, the count is of the entire agent's journey time, even if a portion of the journey takes place outside of the specified time range.</p> <p><b>Agent starts in range:</b> Count/list agents who enter the simulation during the specified time range.</p> <p><b>Agent ends in range:</b> Count/list agents who exit the simulation during the specified time range.</p> <p><b>Agent starts and ends in range:</b> Count/list agents who both enter and exit the simulation completely within the specified time range.</p> <p><b>Agent exists during range:</b> Count/list agents who are present in the simulation during the specified time range.</p>
<b>Time range</b>	The time period used to determine which agents to count/list.
<b>View as</b>	<p>Whether to show results as a matrix or list.</p> <p><b>Matrix:</b> Each cell represents the total number of agents that completed the trip starting at the origin portal and exiting the simulation at the exit portal.</p> <p><b>List:</b> Each row is a single agent, with columns displaying the entrance portal and time and the exit portal and time.</p>
<b>Aggregation Type</b>	<p><b>Total:</b> Add times for all agents with the same entrance/exit pair.</p> <p><b>Average:</b> Calculate the average trip time between each entrance/exit pair.</p> <p><b>Minimum:</b> Calculate the minimum trip time between each entrance/exit pair.</p> <p><b>Maximum:</b> Calculate the maximum trip time between each entrance/exit pair.</p>
<b>Hide empty rows and columns</b>	When checked, rows and columns that contain only zeros are hidden.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table. When set, time will only be recorded for an agent when it satisfies the filter. This means that some agents may have only a portion of their journey included.
<b>Target</b>	<b>All portals in model:</b> Agents entering or exiting from any portal in the scene are included. Portals are listed in alphabetical order.

	<p><b>Specified portals:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in alphabetical order.</p> <p><b>Specified portals in order:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in the order specified.</p>
<b>Notes</b>	User comments about the table.

4.3.4.6.14 Simulation OD Social Cost Matrix

The origin/destination social cost matrix displays the social cost or generalized journey times for agents, organized by entrance/exit. Results can be displayed as the total value for all agents with the given entrance/exit pair, the average value, the maximum, or the minimum. When using an agent filter, each agent's time/cost count includes only those frames when the agent satisfied the filter.

The "Social Cost" tab provides options for configuring the type of value to display and how that value should be calculated. For details on the agent social cost calculations see [agent social cost](#).

Entrance portals are listed as row headers. Exit portals are listed as column headers.

General	
<b><a href="#">Simulation run</a></b>	The simulation run for which time values should be calculated.
<b>Count if:</b>	<p>This can be used to specify which agents to include based on whether or not they enter or exit the simulation within a specified time range. If an agent is included in the count, the count is of the entire agent's journey time, even if a portion of the journey takes place outside of the specified time range.</p> <p><b>Agent starts in range:</b> Count/list agents who enter the simulation during the specified time range.</p> <p><b>Agent ends in range:</b> Count/list agents who exit the simulation during the specified time range.</p> <p><b>Agent starts and ends in range:</b> Count/list agents who both enter and exit the simulation completely within the specified time range.</p> <p><b>Agent exists during range:</b> Count/list agents who are present in the simulation during the specified time range.</p>
<b>Time range</b>	The time period used to determine which agents to count/list.
<b>View as</b>	<p>Whether to show results as a matrix or list.</p> <p><b>Matrix:</b> Each cell represents the total number of agents that completed the trip starting at the origin portal and exiting the simulation at the exit portal.</p> <p><b>List:</b> Each row is a single agent, with columns displaying the entrance portal and time and the exit portal and time.</p>

<b>Aggregation Type</b>	<p><b>Total:</b> Add social costs for all agents with the same entrance/exit pair.</p> <p><b>Average:</b> Calculate the average social cost between each entrance/exit pair.</p> <p><b>Minimum:</b> Calculate the minimum social cost between each entrance/exit pair.</p> <p><b>Maximum:</b> Calculate the maximum social cost between each entrance/exit pair.</p>
<b>Hide empty rows and columns</b>	When checked, rows and columns that contain only zeros are hidden.
<a href="#">Include Agents</a>	An agent filter that determines which agents will be included in the table. When set, time will only be recorded for an agent when it satisfies the filter. This means that some agents may have only a portion of their journey included.
<b>Target</b>	<p><b>All portals in model:</b> Agents entering or exiting from any portal in the scene are included. Portals are listed in alphabetical order.</p> <p><b>Specified portals:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in alphabetical order.</p> <p><b>Specified portals in order:</b> Agents entering or exiting from the specified portals. If collections are used, counts for all portals in the collection are reported for the collection. If the same portal occurs in multiple collections then counts for that portal will be presented multiple times. Entries are listed in the order specified.</p>
<b>Notes</b>	User comments about the table.

<b>Social Cost</b>	
<b>Display Type</b>	Specifies the type of value to calculate and display in the matrix or list.
<b>Cost per Hour</b>	The price used to convert the generalized journey time to a cost value. The cost is first converted into a 'cost per second' value, then multiplied by the generalized journey times.
<b>Days per Year</b>	The number of days in a year. It is assumed that calculated costs are for one day. These day cost values are multiplied by the number of days in a year to produce the annualized cost.
<b>Walking Weight</b>	A factor applied to time spent walking on floors or ramps.
<b>Waiting Weight</b>	A factor applied to time spent waiting. This time includes waiting for a closed gate to open or waiting in response to a wait task. A wait task can be given by the creation event (evacuation event, circulation event) or given by an action.

<b>Queuing Weight</b>	A factor applied to time spent queuing at a server. This does not apply to agents queuing at a link, stair, ramp, or escalator.
<b>Processing Weight</b>	A factor applied to time spent being processed by a server.
<b>Stairs Up Weight</b>	A factor applied to time spent walking up stairs.
<b>Stairs Down Weight</b>	A factor applied to time spent walking down stairs.
<b>Escalator Weight</b>	A factor applied to time spent riding on escalators (up or down).
<b>Custom Weight</b>	<p>A custom weight that can be applied to agents based on an agent filter. When enabled, agents that pass the filter in a given frame will use the custom weight for that frame. Only agents that do not pass the filter will use the other activity weights such as walking or waiting.</p> <p>This can be useful for applying a custom weight to a specific activity in a specific area of the model. Or by setting the weight to 0, it can also be used to exclude agents from being counted under certain conditions.</p>

4.3.4.6.15 Server Summary

Server summary tables display the average, maximum or minimum values of various [server](#) performance metrics over several simulation runs (e.g., several runs with different random seeds used to check for random variation).

<b>Server Summary Parameters</b>	
<b><a href="#">Simulation run</a></b>	The simulation runs over which server summaries will be calculated.
<b>Aggregation</b>	How values should be aggregated across multiple simulation runs (average, maximum or minimum).
<b>Time range</b>	The time period over which values should be calculated.
<b><a href="#">Include Agents</a></b>	An agent filter that determines which agents will be included in the table.
<b>Servers</b>	Which servers the table will refer to.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one server, aggregated across the given runs using the given aggregation type. Only agents that satisfy the given filter will contribute to the computed values. All times are rounded to the nearest second.

**Server Summary Columns**



<b>Server Name</b>	Name of the server.
<b>Total Agents Processed</b>	Total number of agents successfully processed by the server.
<b>Mean Population</b>	Average number of agents at the server.
<b>Mean Pre-Contact Wait</b>	Average amount of time agents waited in the input buffer before being processed by the server.
<b>Mean Contact Wait</b>	Average amount of time agents spent in contact being processed by the server.
<b>Mean Post-Contact Wait</b>	Average amount of time agents waited in the output buffer after being processed (until a space became available at a downstream server).
<b>Mean Total Duration</b>	Average total amount of time agents spent waiting at the server (pre-contact + contact + post-contact).
<b>Max Population</b>	Maximum number of agents ever at the server at one time.
<b>Max Pre-Contact Wait</b>	Maximum amount of time any agent waited in the input buffer before being processed by the server.
<b>Max Contact Wait</b>	Maximum amount of time any agent spent in contact being processed by the server.
<b>Max Post-Contact Wait</b>	Maximum amount of time any agent waited in the output buffer after being processed (until a space became available at a downstream server).
<b>Max Total Duration</b>	Maximum total amount of time any agent spent waiting at the server (pre-contact + contact + post-contact).

Note that 'Mean' or 'Max' in each column name refers to a mean or max that is performed **within** each run; these values are then aggregated **across** runs using the given aggregation type. For example:

- If aggregation type is set to 'Average over simulations', then the 'Max Queue Size' column has the following interpretation: For a particular server, calculate the maximum queue size at the server within each run, then take the average of those values to report in the table.
- If the aggregation type is set to 'Maximum over simulations', then the 'Mean Queue Size' column has the following interpretation: For a particular server, calculate the mean queue size at the server within each run, then take the maximum of those values to report in the table.

4.3.4.6.16 Performance Table

Performance tables can be used to compare performance statistics about different simulation runs. This includes both population data and timing information from when the simulation was run.

Performance Table Parameters	
<a href="#">Simulation run</a>	The simulation runs to be analysed
<b>Aggregation</b>	How timing information for each frame should be combined.  <b>Total Duration (s):</b> Display the total time the simulation spent on each simulation component over the entire time range. <b>Average Frame Duration (s):</b> Display the average time the simulation spent on each simulation component for each frame. <b>Maximum Frame Durations (s):</b> Display the maximum time the simulation spent on each simulation component for each frame.
<b>Time range</b>	The time period over which simulation statistics should be calculated.
<b>Notes</b>	User comments about the table.

Each row of the table has information on one simulation run.

Agent Token Time Columns	
<b>Simulation Run</b>	Name of the simulation run.
<b>Population Data</b>	Information about each simulation run's population over the specified time range.  <b>Avg Population:</b> The average population over the time range. <b>Avg Density:</b> The average population density over the time range. <b>Max Population:</b> The maximum population over the time range. <b>Max Density:</b> The maximum density over the time range.
<b>Timing Information</b>	Timing information from when the simulation was run. Aggregates of each simulation component's run time are displayed here.  <b>Entire Frame:</b> The total time taken to process a frame. If the aggregation type is "Total Duration", this column will list the total time the simulation took to run. If the aggregation type is "Average Frame Duration", this column will list the average time to simulate one frame. If the aggregation type is "Maximum Frame Duration", this will list the longest time to simulate one frame.  Additional Debug Information: <ul style="list-style-type: none"> <li>• <b>Serial Components:</b> The total time taken to process the serial components.</li> <li>• <b>Threaded Components:</b> The total time taken to process the threaded components.</li> <li>• <b>Update Factories</b></li> <li>• <b>Create/Delete Agents</b></li> <li>• <b>Agent Spatial Hash</b></li> <li>• <b>Cache Agent State</b></li> </ul>

	<ul style="list-style-type: none"> <li>• <b>Find Agent Neighbours</b></li> <li>• <b>Update Events</b></li> <li>• <b>Frame Begin</b></li> <li>• <b>Execute Tasks</b></li> <li>• <b>Update Queues</b></li> <li>• <b>Update Controllers</b></li> <li>• <b>Update Process Chains</b></li> <li>• <b>Move Agents</b></li> <li>• <b>Correct Agent Overlap</b></li> <li>• <b>Correct Agent Height</b></li> <li>• <b>Write Database</b></li> <li>• <b>Process Agents</b></li> <li>• <b>Assess Task Progress</b></li> <li>• <b>Frame End</b></li> </ul>
--	---

### 4.3.5 Bookmark

Bookmarks are used to store how a scene currently appears in a view, and then quickly re-apply that same configuration at a latter time. Bookmarks can optionally set the viewpoint, object visibility, simulation time, and any of the view options.

Bookmarks might be used during authoring to quickly jump to a view of the inside of a building where all exterior walls are hidden, or to jump between views of different levels of a train station. Bookmarks might also be created for use in a presentation, highlighting different areas of interest at different times in a simulation.

Bookmarks can be created and applied through the bookmark menu in the [3D Scene View](#).

#### Properties

Scene Tab	
<b>Bookmark Type</b>	<p>Three checkboxes determine the operations performed when the bookmark is applied:</p> <p><b>Viewpoint:</b> When checked, applying the bookmark will change the viewpoint.</p> <p><b>Time:</b> When checked, applying the bookmark will change the simulation playback time.</p> <p><b>Visibility:</b> When checked, applying the bookmark will change object visibility.</p>
<b>Time</b>	<p>If the <b>Time</b> property is checked in the <b>Bookmark Type</b> section, applying this bookmark will change the current playback time to the specified time. If the time is before the simulation begins or after it ends, it will skip to as close as possible.</p>
<b>Visibility</b>	<p>If the <b>Visibility</b> property is checked in the <b>Bookmark Type</b> section, applying this bookmark will change which objects are shown or hidden in the scene.</p> <p><b>Show all objects except:</b> Hide the specified objects and show all others.</p> <p><b>Hide all objects except:</b> Show the specified objects and hide all others.</p>

View Options Tab	
<b>View Options</b>	<p>A bookmark can be used to configure the appearance or behaviour of a scene view. When checked, the corresponding options or settings as displayed in the bookmark will be applied to the target scene view. The options are grouped by menu.</p> <p><b>Render Type:</b> Change the render type to "Shaded" or "Wireframe".  <b>Agent Appearance:</b> Change the appearance of agents.  <b>Decoration Appearance:</b> Change which decorations appear in the scene.  <b>Geometry Appearance:</b> Change the appearance of geometry.  <b>Overlay appearance:</b> Change which overlay elements appear.</p>

Notes Tab	
<b>Notes</b>	<p>A simple field that can be used to save comments or explanation about the bookmark.</p> <p>Bookmarks can be included in a <a href="#">simulation slice export</a>. When opened in the MassMotion Viewer, the notes will be displayed.</p>

#### Collections in bookmarks

[Collections](#) can be used in the "Visibility" property. All scene objects included in the collection will be shown or hidden if visibility changes are active.

## 4.4 Simulation

### 4.4.1 Agent Behaviour

#### 4.4.1.1 Agent Tasks

Each agent in the simulation maintains a list of tasks or "things to do". The agent is only capable of working on one task at a time, and the task currently being considered is said to be the *active* task. Each frame, the agent determines what to do and how to do it based on the active task. Once the active task is complete, the agent will move on to the next task in its list. Tasks are always executed in order.

#### Giving Agents New Tasks

Agents can be assigned new tasks using [actions](#). When an agent receives one or more tasks from an action, those tasks are pushed in front of any currently active tasks and executed in order. The agent will return to the deferred tasks when all of the new tasks have been completed.

#### Types of Tasks

For a complete list of available tasks, please see the task giving actions in [Agent Actions](#). Typical tasks include:

- Moving to a portal destination.
- Moving to and entering a process chain.
- Evacuating a zone.
- Waiting in an area for some duration.
- Executing a sequence of sub tasks (in order).
- Exiting the simulation.

#### 4.4.1.2 Agent Navigation

Navigation is required when an agent's task is to seek one or more destinations. Agents use the [scene network](#) to determine available route options, evaluate the cost of each option, and choose routes with optimal cost. The first step in the chosen route is called the local target. The choice of target is periodically re-evaluated as local conditions change. Once the agent has reached the target and transitioned onto that target, the routes leading from that target are evaluated and the next local target is chosen.

Agents are only aware of route options off of their immediate floor. Any congestion or surprise conditions on downstream floors are not factored into the local choice.

##### Networks

By default, agents are aware of all routes in the scene. It is possible to restrict access to a subset of routes using [network](#) objects. An agent that should not use stairs can be given a network that doesn't include any stairs.

##### Costing Routes

A local target is chosen based on a comparison of the cost of each possible route off of the current floor. A number of components are considered for each route. Distance values are converted into time by dividing by the agent speed. Component time values are then summed to produce a total cost for the route.

Route Cost Components	
<b>Downstream Horizontal Distance</b>	The shortest possible horizontal distance from the target to the goal.
<b>Weighted Downstream Vertical Displacement</b>	The vertical displacement measured along the route that was traced to measure the downstream horizontal distance. Components of the vertical displacement are multiplied by a factor based on object type ( <a href="#">stair</a> , <a href="#">escalator</a> , <a href="#">ramp</a> , etc.) [1].
<b>Near Horizontal Distance</b>	The horizontal distance from the agent to the target.
<b>Queue Time</b>	The expected time it will take to queue for the target, calculated using the number of people queuing in front of the agent and the expected flow rate onto the target. Agents are only aware of queuing for objects leading off of their current floor.
<b>Opposing Flow</b>	A penalty time based on the magnitude of the oncoming flow across the target.
<b>Closed Penalty</b>	A penalty time if the target is currently closed to the agent (See <a href="#">Connection Objects</a> for information on gates and priority access).
<b>Backtrack Penalty</b>	A penalty time if the agent has already used the target (bias against backtracking).

##### Stochastic Elements

There are two areas where [randomness](#) is introduced into the navigation process: agent personality and choice variability.

Each agent is assigned a unique personality based on a set of costing weights. These weights are

applied to the various cost components when evaluating routes. The weights are calculated from distributions defined in the agent's [profile](#). An agent with a high queue cost weight and a low horizontal distance weight will tend to avoid large queues in favour of longer uncongested options.

Each route choice is assigned a small random factor. This factor will be different for each agent each time the agent steps onto a floor. As a result, agents will on occasion choose slightly less than optimal routes ensuring that not all agents make the same choice when routes are very close in cost.

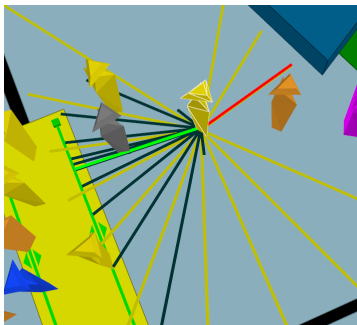
[1] Business Case Development Manual, Transport For London, May 2013, Appendix E 3.1

### 4.4.1.3 Agent Movement

Agent movement is directed by a series of forces acting on the agent. These forces are based on an awareness of the environment including direction to target, location of neighbors, and the location of obstacles.

#### Finding the Target

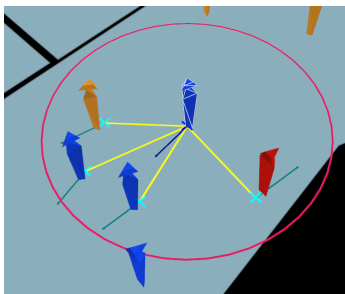
Agents moving towards a local target will determine the direction to that target by looking at the [approach map](#) for the target's goal line. The agent extends straight feelers out in various directions and measures the distance to the target along each of the feelers. The feeler that ends up closest to the target goal line is taken as the direction to goal.



Agent awareness of direction to target.

#### Neighbours

Each agent is aware of other agents that are within a particular range. This range changes with the speed of the agent and the local density. Other agents inside the awareness range are called neighbours. An agent is aware of the location, speed, and size of its neighbours.

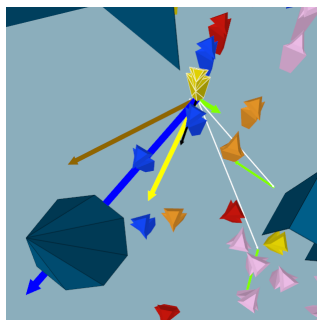


Agent awareness of surrounding neighbours

#### Summing Forces

The social forces algorithm generates a series of component forces based on the agent's goal and

environment. The component forces are summed together and used to calculate the agent's acceleration, velocity, and new position. Component forces are derived from the agent's desired target, the presence of neighbouring agents, the location of obstacles and other situational factors.



Forces acting on agent

Component Forces	
<b>Goal</b>	Force required to nudge agent so that it is at its desired speed heading towards its target.
<b>Neighbour</b>	Repulsive force from each neighbour within range.
<b>Cohesion</b>	Force pushing towards centroid of neighbours with similar targets.
<b>Collision</b>	Force pushing agent away from collisions with oncoming neighbours.
<b>Drift</b>	Force pushing agent in bias direction when faced with oncoming agents in narrow spaces.
<b>Orderly Queuing</b>	Force pushing agents towards the middle of a target when approaching.
<b>Corner</b>	Force pushing agents to hug or swing wide around a corner.

It should be noted that obstacles do not result in a repulsive force of their own, but are used to constrain other forces. When component forces are summed, the resulting net force is reduced such that it does not push the agent into a barrier.

### Agent Speed

The agent's desired speed is the speed at which the agent will walk when on flat ground in an uncongested environment. This speed is assigned through the agent's [profile](#) when the agent is created.

The actual speed of an agent at any given time depends on a number of additional factors including density and the object on which the agent is walking.

Factors Influencing Agent Speed	
<b>Density</b>	In order to simulate the reduced stride length and reduced mobility of people in crowded spaces, agent speed is reduced as density increases. The exact relationship between speed and density has been tuned to match the data in John Fruin's Pedestrian Planning & Design [1]. This relationship can be disabled in the agent's profile.

Factors Influencing Agent Speed	
<b>Object Type</b>	<a href="#">Escalator</a> : Agent speed is set to exactly match the escalator speed property. <a href="#">Floor</a> or <a href="#">Link</a> : Agent speed is not altered. <a href="#">Path</a> or <a href="#">Server</a> : When immediately behind another agent, speed is reduced to match the agent in front. <a href="#">Ramp</a> or <a href="#">Stair</a> : Agent speed is modified based on whether travel is in the up or down direction. See the object reference pages for more information.
<b>Object Speed Limit Property</b>	Objects have a property for capping agent speed. Any agent on the object with a speed above the cut-off will have their speed reduced to the cutoff. Agents below the cutoff are unaffected.

[1] Fruin, John J. Pedestrian Planning & Design, Revised Edition Chapt. 4, Elevator World, 1987

## 4.4.2 Surface Maps

Walkable space on a [floor](#), [link](#), [stair](#), [ramp](#), or [escalator](#) is represented through surface maps. A surface map is a 2D grid. Red values indicate areas that are blocked by an obstacle or beyond the edges of the walkable object. Black, white, or grey areas represent useable space. Surface maps are created automatically for each object in the scene during simulation initialization. Understanding surface maps can help with understanding how people in MassMotion navigate a space.

Surface maps can be viewed by right-clicking on an object and choosing the 'Display' menu, or they can be exported as part of the simulation results (see [Generated Simulation Files](#)).

There are two types of surface maps:

- [Obstacle Maps](#)
- [Approach Maps](#)

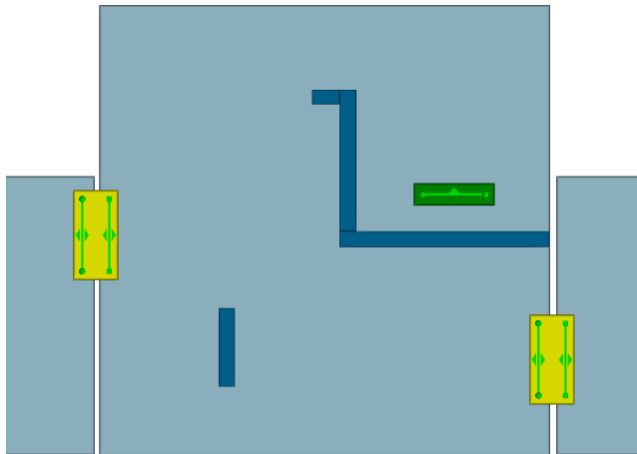
### 4.4.2.1 Obstacle Maps

Obstacle maps describe the distance to the closest obstacle or edge for each point on an object. Obstacle maps can also be used to determine which space is passable and which is blocked (by an obstacle or floor edge).

Maps can be viewed by right-clicking on an object and choosing the 'Display' menu

#### Simple Scene

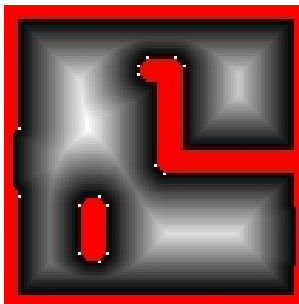




A room with one portal and two links

### Obstacle Map

Obstacles influence the map if they intersect any part of the floor or are within 0.4m of the top of the floor. Any portion of an obstacle that is below the floor or above the 0.4m cut-off is ignored. Floor edges and included obstacles are marked as red and are unavailable for walking. The shaded regions represent the normalized distance to the nearest obstacle, with black as a distance of 0 and white the farthest distance.



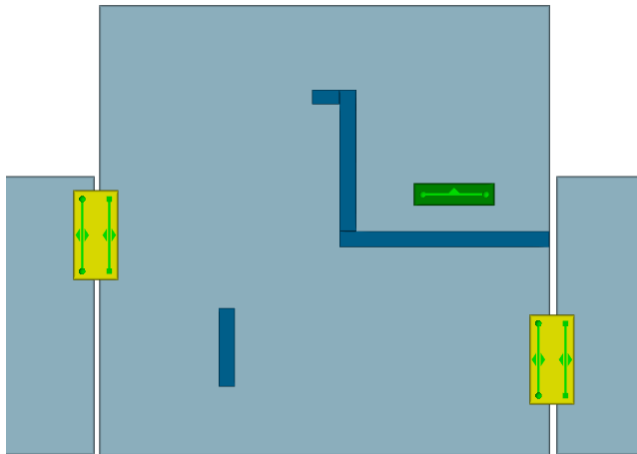
Obstacle map of room

#### 4.4.2.2 Approach Maps

Approach maps describe the shortest distance to a goal line from any point on an object. A single floor will have a different approach map for each goal line connected to the floor. Approach maps can also be used to determine whether space is passable or blocked (by an obstacle or off of the floor edge).

Maps can be viewed by right-clicking on an object and choosing the 'Display' menu.

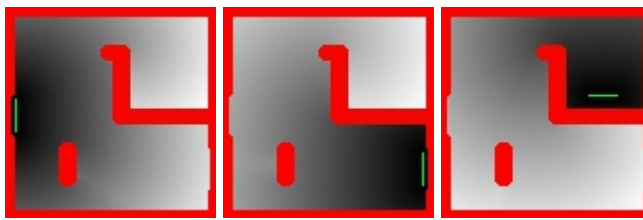
### Simple Scene



A room with one portal and two links

### Approach Maps

Approach maps represent unavailable areas in red. The shaded areas describe distance to a single goal line on the walkable surface. For the simple room above, there are three goal lines and so three approach maps. In each case black represents a distance value of 0, and white represents the furthest distance from the goal line.



Link 1 approach      Link 2 approach      Portal approach

### 4.4.3 Areas

Areas are regions of the scene which can contain agents. Areas are commonly used in tests, triggers, and analysis objects to identify whether or not agents are inside a specific region. Areas can be defined using different objects.

Area Type	Agents Considered 'In'
<a href="#">Collection</a>	Collections which contain area objects can be used as areas themselves; they will be displayed using the '.Areas' suffix. Agents are 'in' the collection area if they are in any of the member's areas. Any members which are not areas will be ignored.
<a href="#">Volume</a>	Agents with the point at the centre of their feet contained within the volume.
<a href="#">Wait Space</a>	Agents are inside the wait space if both of the following are true: 1) The point at the centre of their feet is contained within the wait space volume. 2) The agent is currently waiting on the wait space.
<b>Walkable object</b>	Agents currently on the walkable object (floor, link, escalator, ramp, stair, path).

Area Type	Agents Considered 'In'
<a href="#">Zone</a>	Agents on any of the walkable objects that are part of the zone.

#### 4.4.4 Wait Style

An agent's wait style determines how the agent behaves while waiting. Agents might stand in place, spread out, or move to a specified [wait space](#). The wait style is specified by the gate, elevator, or action which is causing the agent to wait.

Style	Used by	Description
<b>Stand Still</b>	<b>Gates, Elevators, Wait Spaces, Circulate, Evacuate, Actions</b>	Agents will stand motionless wherever they are. Agents can be jostled out of place by passing neighbours but will otherwise remain in place.
<b>Spread out</b>	<b>Gates, Elevators, Wait Spaces, Circulate, Evacuate, Actions</b>	Agents will spread out to use all available space.
<b>Focus on target</b>	<b>Gates, Elevators</b>	When waiting for a particular object such as a gate or elevator, agents will move to and wait in front of that object.
<b>Focus beside target</b>	<b>Gates, Elevators</b>	When waiting for a particular object such as a gate or elevator, agents will move to that object and stand just on either side. This will leave the area directly in front of the object free.
<b>Use wait space (assigned)</b>	<b>Circulate, Evacuate, Actions</b>	Agents will be assigned a <a href="#">wait space</a> on the current floor and move towards the goal line of that wait space. As soon as the agent is standing on the wait space it will assume the wait style of that wait space.
<b>Use wait space (chosen)</b>	<b>Gates, Elevators, Circulate, Evacuate, Actions</b>	Agents will choose from the available <a href="#">wait spaces</a> on the current floor and move towards the goal line of the chosen wait space. As soon as an agent is standing on the wait space it will assume the wait style of that wait space.  Agents assign costs to each wait space and choose the option with the lowest cost. Costs are calculated using the distance to the wait space and the relative density at the wait space.
<b>Cluster around target</b>	<b>Wait Spaces</b>	Agents will cluster around the wait space goal line with the 'Weight' property regulating how densely packed the agents will cluster. A value of 1.0 will produce a dense tightly packed cluster. A value of 0.0 will produce a very loose clustering with most agents spreading out.

## 4.4.5 Simulation Execution

### 4.4.5.1 Starting a Simulation

A new simulation can be started from the 'Run Simulation' button in the simulation & analysis ribbon of the main window. The launch dialog provides the ability to specify the type of run and the simulation run object in which to store the results.

Type	
<b>Standard console</b>	Run a <a href="#">Console Simulation</a> , which is the fastest option but does not include a graphical window for viewing the simulation as it runs.
<b>Debug viewer</b>	Run a <a href="#">Debug Simulation</a> , which will execute more slowly than a console simulation but present a <a href="#">scene view</a> that allows interactive viewing and debugging of the simulation as it runs. Additionally, a breakpoint can be set to pause the simulation at the specified time. The breakpoint is useful when debugging problems that occur late in the simulation at known times.
<b>Multiple runs</b>	Run several iterations of the same simulation with different random seeds (see below for details). The results for each iteration will be placed in a different simulation run object.

Simulation Run	
<b>Create new</b>	<p><b>Console/Debug:</b> Create a new simulation run object with the given name. The results from the simulation will be placed in the specified database file and referenced by the new simulation run.</p> <p><b>Multiple Runs:</b> Create the specified number of simulation run objects. The objects will be given unique names by appending numbers to the specified name stem. Numbers are chosen so as not to collide with existing names in the project. Database files will be named after the corresponding run and placed in the specified path.</p>
<b>Overwrite existing</b>	<p><b>Console/Debug:</b> Use an existing simulation run object; the database file referenced by the run will be overwritten.</p> <p><b>Multiple Runs:</b> Create or overwrite the specified number of simulation run objects. If new simulation run objects are created, database files are named after the run and placed in the specified path. If simulation run objects with the requested names already exist they are used and their existing database files overwritten.</p>

#### Random Seeds

It is possible to either set the [random seed](#) used to run the simulation, or leave it at the default value (which is taken from the [project settings](#)). Running a simulation twice with the same random seed will produce identical results. Running a simulation with a different random seed will introduce random variation in agent behaviour but should produce statistically similar overall behaviour; varying the random seed is one way to determine the sensitivity of the model to small changes.

In the case of a multi-run simulation, the specified random seed will be used for the first run, and will then be used to generate a new random seed for the second run and so on. This is done in a consistent way so if a second multi-run simulation is performed with the same initial seed, all

subsequent seeds will be the same as in the first multi-run simulation.

### Threading

By default, MassMotion will run a fully multi-threaded simulation using all available CPU cores. This will typically result in the shortest simulation time but other applications on the same computer may become sluggish. To avoid this, it is possible to disable threading entirely (so that MassMotion will only use one CPU core) or specify the number of threads that should be used. When setting the number of threads to be used, there are two main considerations to keep in mind:

- Specifying a number of threads greater than the number of CPU cores available can result in slower performance.
- Larger simulation populations are required to take full advantage of a larger number of threads. In some cases when dealing with a small population over a long period of time, it is better to specify a lower number of threads.

### Stopping a Simulation

All simulation types provide a 'stop' button; closing a simulation window will also stop the simulation. Stopping a simulation causes all current results to be written to the results database. A simulation that is stopped early can still be used for playback and analysis, although care must be taken when interpreting results.

#### 4.4.5.2 Console Simulation Window

The Console Simulation runs with minimal overhead, thereby maximizing use of available computing resources to minimize run time. A console simulation runs faster than a [debug simulation](#), however, details of the simulation cannot be accessed while the simulation is running. The results of the run become available for playback and analysis once the run is complete.

Console Simulation Window Components	
<b>Log Window</b>	On the left side of the console simulation window is a live console that displays diagnostic information, warnings, and errors about the initialization and execution of the current project. The level of detail reported in the console can be specified by right-clicking on the console or by selecting an option from the drop-down menu above the console. The output can also be saved to a text file with the 'save' icon above the console.
<b>Issues Window</b>	On the right side of the console simulation window is an embedded version of the <a href="#">issues window</a> that shows warnings and errors that were encountered while running the simulation. The buttons along the top of the issue window allow expanding or collapsing all items, saving the issues to a file or clearing all current issues.
<b>Simulation Controls</b>	At the bottom of the console simulation window is a progress bar showing the current progress of the simulation and buttons for pausing the simulation or stopping it entirely. The 'reload' button will reload any changes made to the current project, and then restart the simulation with those changes.

#### 4.4.5.3 Debug Simulation Window

Running a debug simulation creates a new window that allows visual debugging of the simulation. The debug simulation window includes all the functionality of the console simulation window, including a console window for displaying information about simulation execution, an issues window for displaying warnings/errors, and controls for reloading, pausing/resuming and stopping the simulation; see [Console Simulation Window](#) for details.

In addition, the debug simulation window has a list view that works exactly the same way as the main window [list view](#) to allow selection of objects in the simulation, and its own scene view similar to the one in the main window but with additional functionality available only while simulating (see [Simulation Scene View](#)). A properties pane is available on the right-hand side to [view details](#) about the state of the currently selected agent or scene object.

The debug simulation window toolbar contains buttons on the right-hand side for hiding and showing the list view, scene view, console/issue windows and object details pane.

4.4.5.3.1 Simulation Scene View

The Debug simulation window has 3D view controls similar to those of the main window as described in [3D Scene View](#). In addition, there are controls for pausing/stepping/resuming the simulation, and options for displaying debug information about individual agents and scene objects.

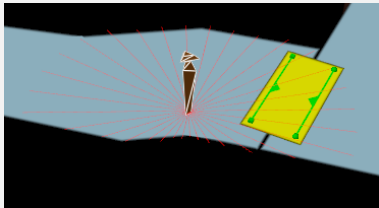
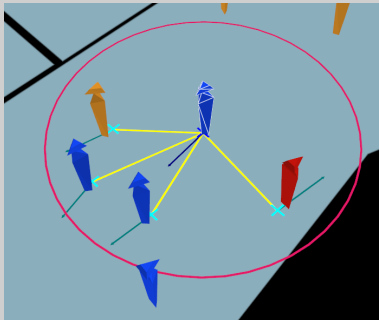
**Simulation Control**

Simulation execution can be paused and controlled using the keyboard.

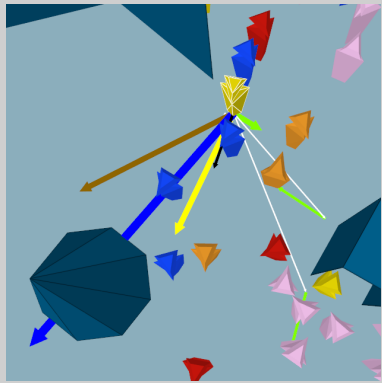
Simulation Control	
<b>Space Bar/ Up Arrow Key</b>	Toggles the simulation paused state.
<b>Left Arrow Key</b>	Advances the simulation by one frame at a time. This feature is only available when the simulation is paused.

**Agent Display Options**

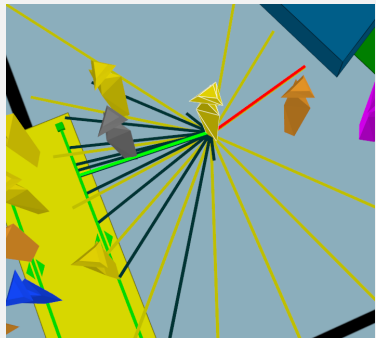
Right-clicking on one or more agents in a debug simulation window will bring up a context menu with various options under the 'Display' sub-menu that are not available during playback. These include:

Agent Display Options							
<b>Avail able Spac e</b>	 <p>Displays the available space around an agent. Each line indicates how far the agent can move in a certain direction.</p> <p>The lines turn red when the agent is approaching a <a href="#">connection object</a>.</p>						
<b>Neig hbou rho od</b>	 <p>Displays the local area vision bounds and indicates neighbouring agents within the selected agent's awareness. Note: the information displayed is for the previous simulation frame.</p> <table border="1"> <tbody> <tr> <td><b>Yello w Line</b></td> <td>Lines from central agent to surrounding neighbours.</td> </tr> <tr> <td><b>Pink Ring</b></td> <td>Displays geometric bounds of agent's awareness.</td> </tr> <tr> <td><b>Turq uois</b></td> <td>Velocity of neighbouring agent.</td> </tr> </tbody> </table>	<b>Yello w Line</b>	Lines from central agent to surrounding neighbours.	<b>Pink Ring</b>	Displays geometric bounds of agent's awareness.	<b>Turq uois</b>	Velocity of neighbouring agent.
<b>Yello w Line</b>	Lines from central agent to surrounding neighbours.						
<b>Pink Ring</b>	Displays geometric bounds of agent's awareness.						
<b>Turq uois</b>	Velocity of neighbouring agent.						

Agent Display Options																			
	<table border="1"> <tr> <td><b>e Arrow</b></td> <td></td> </tr> <tr> <td><b>Dark Blue Arrow</b></td> <td>Velocity of central agent.</td> </tr> </table>	<b>e Arrow</b>		<b>Dark Blue Arrow</b>	Velocity of central agent.														
<b>e Arrow</b>																			
<b>Dark Blue Arrow</b>	Velocity of central agent.																		
<b>Route Costing</b>	<div style="display: flex; align-items: flex-start;"> <div style="flex: 1;"> </div> <div style="flex: 2;"> <p>Displays instantaneous route costs (distance, queue, vertical, total) for the routes leading off of the current floor.</p> <table border="1"> <tr> <td><b>D Distance cost</b></td> <td>Total distance cost</td> </tr> <tr> <td><b>Q Queue cost</b></td> <td>Total queue cost (proportional to size of queue)</td> </tr> <tr> <td><b>C Opposing cost</b></td> <td>Cost proportional to the magnitude of the flow in the opposing direction</td> </tr> <tr> <td><b>V Vertical cost</b></td> <td>Total vertical cost (proportional to vertical stair/ramp/escalator height)</td> </tr> <tr> <td><b>T Total cost</b></td> <td>Cumulative cost *</td> </tr> </table> <p>*Note: Total cost may not always equal the sum of individual costs, as only significant individual costs are shown.</p> <p>Text colour is used to indicate route status:</p> <table border="1"> <tr> <td><b>Green</b></td> <td>This route has been chosen as the target.</td> </tr> <tr> <td><b>Red</b></td> <td>This route is not available as it does not lead to goal.</td> </tr> <tr> <td><b>Yellow</b></td> <td>This object has already been used - backtracking costs may be applied.</td> </tr> <tr> <td><b>White/pale colour</b></td> <td>This route is available and has not been chosen.</td> </tr> </table> </div> </div>	<b>D Distance cost</b>	Total distance cost	<b>Q Queue cost</b>	Total queue cost (proportional to size of queue)	<b>C Opposing cost</b>	Cost proportional to the magnitude of the flow in the opposing direction	<b>V Vertical cost</b>	Total vertical cost (proportional to vertical stair/ramp/escalator height)	<b>T Total cost</b>	Cumulative cost *	<b>Green</b>	This route has been chosen as the target.	<b>Red</b>	This route is not available as it does not lead to goal.	<b>Yellow</b>	This object has already been used - backtracking costs may be applied.	<b>White/pale colour</b>	This route is available and has not been chosen.
<b>D Distance cost</b>	Total distance cost																		
<b>Q Queue cost</b>	Total queue cost (proportional to size of queue)																		
<b>C Opposing cost</b>	Cost proportional to the magnitude of the flow in the opposing direction																		
<b>V Vertical cost</b>	Total vertical cost (proportional to vertical stair/ramp/escalator height)																		
<b>T Total cost</b>	Cumulative cost *																		
<b>Green</b>	This route has been chosen as the target.																		
<b>Red</b>	This route is not available as it does not lead to goal.																		
<b>Yellow</b>	This object has already been used - backtracking costs may be applied.																		
<b>White/pale colour</b>	This route is available and has not been chosen.																		

Agent Display Options				
<b>Social Forces</b>		Displays the "social forces" to which the agent is subjected.		
		<b>Bright Green</b>	Goal force	Pulls neighbour towards desired destination
		<b>Bright Yellow</b>	Obstacle constrained neighbour force	Repels from surrounding neighbours
		<b>Purple</b>	Drift force	Drifts an agent in direction of bias when interacting with oncoming crowd
		<b>Turquoise</b>	Collision veer force	Veers agent towards direction bias in a head-on collision
		<b>Orange</b>	Collision yield force	Slows down and torques agent to avoid perpendicular collision
		<b>White</b>	Cohesion force	Attracts agents together within a crowd.
		<b>Grey</b>	Orderly Queuing Force	Pushes agents towards the front of the goal to which they are targeted, helping to ensure a straight and orderly queue.
		<b>Brown</b>	Corner Force	Orients agent motion with respect to near corners, other agents, and veer direction bias. (Also note the white and green arrows showing corners of interest)
		<b>Blue</b>	Obstacle constrained net force	Resulting net force
<b>Pink</b>	Correction force	Strong force pushing agent back to walkable area when an agent has been bumped onto a barrier or off of a floor		
<b>Black</b>	Obstacle constrained velocity	Resulting velocity		



Agent Display Options									
<b>Surface Probe</b>	<div style="display: flex; align-items: center;">  <div style="margin-left: 20px;"> <p>Displays agent awareness of surrounding space as it relates to obstacle avoidance and the direction of goal.</p> <table border="1" style="margin-top: 10px;"> <tr> <td><b>Black</b></td> <td>Test target ray</td> </tr> <tr> <td><b>Green</b></td> <td>Test target ray in direction of target</td> </tr> <tr> <td><b>Gold</b></td> <td>Obstacle ray has detected obstacle</td> </tr> <tr> <td><b>Red</b></td> <td>Obstacle ray is in direction of closest obstacle</td> </tr> </table> </div> </div>	<b>Black</b>	Test target ray	<b>Green</b>	Test target ray in direction of target	<b>Gold</b>	Obstacle ray has detected obstacle	<b>Red</b>	Obstacle ray is in direction of closest obstacle
<b>Black</b>	Test target ray								
<b>Green</b>	Test target ray in direction of target								
<b>Gold</b>	Obstacle ray has detected obstacle								
<b>Red</b>	Obstacle ray is in direction of closest obstacle								

4.4.5.3.2 Object Properties

The properties tab is available in the Information Pane at the right-hand side of the simulation window. It can be used to monitor property values for selected agents and scene objects in the simulation. Information is unique to the type of object selected.

Agents	
<b>Properties Tab</b>	
<b>ID</b>	The unique agent ID.
<b>Age</b>	How long this agent has been in the simulation.
<b>Avatar</b>	The avatar currently used by this agent. This is initially assigned by the schedule which created the agent, but may be modified mid-simulation by an action.
<b>Profile</b>	The name of the profile used by this agent.
<b>Radius (m)</b>	The current body radius of the agent.
<b>Tokens</b>	A list of tokens currently held by the agent - this can change over the course of the simulation.
<b>Current Network</b>	The network object currently used by the agent (World if non displayed).
<b>Active Task</b>	The current task being performed by the agent. In the case of the Seek task, the ultimate goal of the agent is also listed.
<b>Active Dispatch</b>	The name of the dispatch currently controlling the agent.
<b>Next Waypoint</b>	B -> C: The agent's next floor transition (moving from B to C where C is considered the local goal).
<b>Desired Stop</b>	The elevator stop requested (if targeting an elevator).
<b>Current Floor</b>	The floor, link, stair, escalator, or ramp that the agent believes it is

	standing on.
<b>Last Waypoint</b>	A -> B: The agent's previous floor transition (moving from A to B).
<b>Speed Current (m/s)</b>	The current speed of the agent.
<b>Speed Current Max (m/s)</b>	The current maximum speed the agent could achieve (affected by local limits imposed by the current floor, link, escalator, ramp, or stair, or by density related constraints).
<b>Speed Default/Natural Walkway (m/s)</b>	The natural walking speed of the agent if on flat ground and unconstrained by neighbours.
<b>Activity</b>	<p>What kind of activity the agent is currently involved in. Possible values are:</p> <ul style="list-style-type: none"> <li>• In Input Buffer (wait): The agent is waiting for a gate to open.</li> <li>• In Input Buffer (queue): The agent is queueing to reach the next waypoint.</li> <li>• In Input Buffer (free): The agent is freely moving towards the next waypoint.</li> <li>• Being Processed: The agent has reached and is currently being processed by the next waypoint.</li> <li>• In Output Buffer: The agent has finished processing and is waiting for downstream capacity.</li> <li>• Waiting: The agent has been put in a wait state by an action.</li> <li>• Not Registered: Agent is in an error state and may be deleted.</li> </ul>
<b>Waiting?</b>	True if the agent has chosen a gated link as its next waypoint, and is currently waiting for the gate to open.
<b>Queueing?</b>	True if the agent is queueing to reach the next waypoint.
<b>Time in Queue (s)</b>	If queueing, this is the time the agent has so far spent queueing for the next waypoint.
<b>LOS Letter Value</b>	The Fruin LOS letter grade given the stated density (always uses walkway mapping - never queue or stair mappings).
<b>Density (ppl/m<sup>2</sup>)</b>	The current density immediately around the agent.
<b>Space (m<sup>2</sup>/ppl)</b>	The personal space immediately around the agent (inverse of density).
<b>Created By</b>	The agent schedule or timetable that generated the agent.
<b>Notes</b>	Any notes from the agent creation. If generated by a timetable, this will include the corresponding timetable schedule file name and line number.
<b>Start Floor</b>	The floor or portal where the agent entered the simulation.
<b>Start Time</b>	The time at which the agent entered the simulation.
<b>Tasks Tab</b>	

<b>ID</b>	The unique agent ID.
<b>Task Stack</b>	The queue of tasks that are currently part of the agent's itinerary. Tasks are executed in order from top to bottom with new tasks added to the top of the list. The current task is indicated in bold.
<b>Tokens Tab</b>	
<b>ID</b>	The unique agent ID.
<b>Token List</b>	Enumeration of all the tokens that the agent is currently holding.
<b>Actions Tab</b>	
<b>ID</b>	The unique agent ID.
<b>Action List</b>	A list of the actions that have been applied to the selected agent. Each action is represented as an expandable group, with the title of the group containing the time at which the action was applied, the name of the action, and the manner in which the action was triggered (e.g. entering a zone, from an event). The expandable group contains a record of the operations carried out by the action, including TEST, DO, MODIFY, and TASK.

<b>Floors</b>	
<b>ID</b>	The unique object ID.
<b>Name</b>	The unique object name.
<b>Type</b>	The type of object.
<b>Zones</b>	Zones of which this floor is a member
<b>Travel Type</b>	Indicates if agents will traverse floor instantly (virtual), ignoring barriers, or normally.
<b>Surface Resolution</b>	Sampling frequency for goal and obstacle distances on this floor.
<b>Population</b>	Current number of agents on floor.
<b>Route Information</b>	See the description of Waypoint Route Information in the table below.

<b>Portals</b>	
<b>ID</b>	The unique object ID.
<b>Name</b>	The unique object name.
<b>Type</b>	The type of object.

<b>Zones</b>	Zones of which this portal is a member.
<b>On Floor</b>	The name of the floor under the portal.
<b>Is Entrance?</b>	True if the portal is configured as an entrance.
<b>Is Exit?</b>	True if the portal is configured as an exit.
<b>Exit Information</b>	See the description of Waypoint Route Information in the table below.

<b>Links, Stairs, Escalators, Ramps</b>	
<b>ID</b>	The unique object ID.
<b>Name</b>	The unique object name.
<b>Type</b>	The type of object.
<b>Zones</b>	Zones of which this connector is a member.
<b>Travel Type</b>	Indicates if agents will traverse floor instantly (virtual), ignoring barriers, or normally.
<b>Surface Resolution</b>	Sampling frequency for goal and obstacle distances on this object.
<b>Population</b>	Current number of agents on object.
<b>Rise Angle (deg)</b>	The angle of inclination for the stair, ramp, or escalator (not available for links).
<b>Distance Penalty (m)</b>	The distance penalty added to all distance based route costs for this object.
<b>Queue Penalty Factor</b>	The cost factor applied to all queuing at this object.
<b>Perimeter</b>	Indicates perimeter membership.
<b>Bank</b>	Indicates bank membership.
<b>Is Gated?</b>	True if the object is configured as a gate and can be opened or closed.
<b>Wait Style</b>	The agent behavior when waiting for a gate.
<b>Route Information</b>	See the description of Waypoint Route Information in the table below.

<b>Waypoint Route Information</b>	
<b>Available Width (m)</b>	The width of the goal line between the connected floors.
<b>Flow Limit (ppl/min)</b>	Cap (if any) on the allowed flow rate through the waypoint.

<b>Flow Average (ppl/min)</b>	The average flow rate through the waypoint over the previous 5 seconds.
<b>Total Processed</b>	The number of agents who have successfully been processed by this waypoint.
<b>Approaching</b>	The number of agents currently approaching (but not queuing or waiting for) the waypoint.
<b>Queuing</b>	The number of agents currently queuing for the waypoint (an agent is only considered queuing if it has the waypoint as its 'Next Waypoint' and has a speed below a certain threshold).

#### 4.4.5.4 Running from the Command Line

A simulation can be executed using MassMotionConsole from a DOS command prompt. This is useful when running multiple projects in sequence, or when running the same project multiple times with different random seeds.

##### How to Run

1. From a DOS command console, navigate to the MassMotion installation folder (by default C:\Program Files\Oasys\MassMotion 10.0).
2. Run MassMotionConsole.exe with the desired parameters (see table below).
3. Note all diagnostic information will be written to the simulation log text file which will be placed alongside the generated database file.

##### Arguments / Parameters

Parameters are prefixed with a hyphen "-". Some parameters require values separated from the parameter name by a space (e.g., -seed 5).

Option	Description	Example
-csvseparator #	If used in combination with -query, specifies the separating character used in exported csv files.	-csvseparator ;
-dump	Write diagnostic information to a 'debug' folder in the project's working folder (see <a href="#">Project Settings</a> ).	-dump
-fullscreen	If used in combination with -vis, the 3D viewer is drawn in full-screen mode.	-fullscreen
-help	Display the list of available parameters then quit.	-help
-nothreads	Disable the use of threads during the simulation.	-nothreads
-	Used to scale the number of	-popscale 2.0

popscal e	agents generated by all events (must be greater than 0).	
-project #	Specify the MassMotion project file (.mm file) to open and run.	-project C:\mm\testproject\testproject.mm
-query #	Execute the named query and export the table results to a csv. Any simulation run saved in the query is replaced by the database just generated. The file name is constructed from the database file name and query name. This option can be used once for each query object in the project.	-query AgentSummaryTable1 -query AgentSummaryTable2
- queryall	Execute -query on all query objects in the project.	-queryall
-results #	Specify the output database file for results. If the path is relative it is assumed to be relative to the folder containing the project file.	-results C:\mm\Testproject\firstrun.mmdb
-seed #	Override the seed value from the project settings with the given seed value. The same project run multiple times with the same seed value will always produce the same results. If no seed value is specified, the seed value from the project settings is used.	-seed 44321
- threads #	Use the specified number of threads in executing a simulation. By using multiple threads, multiple operations can be performed at the same time, greatly improving performance. The default number is equal to the number of system processors (e.g., 4 for a quad-core computer). A value of 1 will disable multithreading. Note that more threads does not necessarily mean faster execution given the overhead required to start, stop, and manage each thread. The default value is recommended.	-threads 1 (all operations are performed in the main thread) -threads 8 (8 threads are used)
- verbosity #	Control the number and verbosity of messages written to the project log.txt file.	-verbosity DEBUG -verbosity APPLICATION

	Possible values include: <ul style="list-style-type: none"> <li>• ERROR</li> <li>• WARNING</li> <li>• APPLICATION</li> <li>• VERBOSE</li> <li>• DEBUG</li> </ul>	
-vis	Display a 3D view of the scene. The view can be navigated and controlled using the same controls as the regular MassMotion <a href="#">3D scene view</a> , but there are no menus and it is not possible to run more than one project. Running with the 3D view shown will have a negative impact on performance.	-vis

#### 4.4.6 Generated Simulation Files

The following files can be produced when executing a simulation:

Output Type	
<b>DefaultRun.mmdb</b>	An sqlite database file containing all of the information required to analyse and playback a single simulation run. For information on the database see <a href="#">Simulation Data</a> . For information on using map, table, and graph queries to interrogate the database, see <a href="#">Analysis</a> .
<b>DefaultRun.txt</b>	A text log file is created each time a simulation is run. The file contains diagnostic information on project initialization, execution, and general performance. All output displayed in the MassMotion console is also written to the log file. The file is created in the same folder as the database file and given the same name.

A simulation can be configured to generate debug information about the project. The files are placed in a 'debug' folder created inside the project's working path.

Debug File	
<b>Obstacle Map (*.jpg)</b>	File (.jpg) containing an <a href="#">obstacle map</a> of the available space on the given surface and the distance from every point to the nearest obstacle or surface edge.  White - Point farthest from obstacle or surface edge, or indicates the presence of a corner Black - Point closest to an obstacle or surface edge Red - Covered by obstacle or not on the surface

	<p><b>Note:</b> this file is only generated if the dumping of surface maps is enabled in the debug tab of the <a href="#">project settings</a>.</p>
<p><b>Approach Map (*.jpg)</b></p>	<p>File (.jpg) containing an <a href="#">approach map</a> of the distance from every point on a surface to the connected destination object.</p> <p>Green - Goal line                  White - Point farthest from the destination goal line                  Black - Point closest to the destination goal line                  Red - Covered by an obstacle or not on the surface                  Blue - Indicates an unreachable area not connected to the goal line</p> <p><b>Note:</b> this file is only generated if the dumping of surface maps is enabled in the debug tab of the <a href="#">project settings</a>.</p>
<p><b>CostTree (*.csv)</b></p>	<p>A file which describes the distance from every decision point in the scene to the specified goal.</p> <p><b>Note:</b> this file is only generated if the dumping of route costs is enabled in the debug tab of the <a href="#">project settings</a>.</p>

#### 4.4.7 Randomness

MassMotion uses random numbers throughout the simulation. All random numbers are generated from an initial integer seed (see [Project Settings](#)). A project simulated multiple times with the same random seed should produce exactly the same results.

Changing the seed, changing the project (adding, deleting, or changing objects), or running with a different version of MassMotion will result in different simulation results.

##### Distributions

Distributions are used throughout MassMotion as a way of specifying a possible range of values.

Distributions	
<p><a href="#">Duration Distributions</a></p>	<p>Duration based distribution used by <a href="#">journey</a> and <a href="#">circulate</a> events to determine when agents enter the simulation.</p>
<p><a href="#">Standard Distributions</a></p>	<p>Distributions used nearly everywhere, influencing how agents may interact with the scene.</p>

##### 4.4.7.1 Duration Distributions

A duration distribution is used to determine agent arrival times within an interval. There are fewer options than with a [standard distribution](#) as the min and max are taken automatically from the event start time and duration. The specified distribution automatically uses 0 as the min and the event duration as the max. Values generated by the distribution are added to the event start time to produce an agent's ultimate arrival time.

Possible distributions are as follows:



Distribution Types	
<b>Uniform</b>	Agents are assigned random start times according to a uniform distribution. With a large enough number of agents, this should converge on results similar to the constant distribution.
<b>Normal</b>	Agent start times will follow a normal distribution.  <b>Mean:</b> The mean of the normal distribution relative to the start of the event. <b>Std:</b> The standard deviation.
<b>Triangular</b>	Agent start times will follow a triangular distribution.  <b>Mode:</b> The mode of the triangular distribution is relative to the start of the event.
<b>Log Normal</b>	Agents will have start times as if they were assigned by a log normal <a href="#">single value distribution</a> with "Shift" as the event start time and "Max" as the event duration in seconds.  <b>Mu:</b> The mean of the log of the distribution. <b>Sigma:</b> The standard deviation of the log of the distribution.
<b>Exponential</b>	Agents will have start times as if they were assigned by an exponential <a href="#">single value distribution</a> with "Shift" as the event start time and "Max" as the event duration in seconds.  <b>1 / Lambda:</b> The mean start time relative to the start of the event.

#### 4.4.7.2 Standard Distributions

Many object properties are described using a distribution. These properties resolve to single values based on the probability function of the distribution. For example, a [profile](#) defines agent speed as a distribution. During the simulation, each agent is given a single speed according to the distribution.

For information on duration based distributions describing agent arrival times, see [Duration Distributions](#).

The following single value distribution types are supported:

Distribution Types	
<b>Constant</b>	The distribution will always produce the same constant value.  <b>Value:</b> A single number.  <b>Resultant Mean:</b> Value
<b>Uniform</b>	The distribution will produce a random number between the minimum and maximum value. All values within the range are equally likely.  <b>Min:</b> The minimum possible value. <b>Max:</b> The maximum possible value.

	<p><b>Resultant Mean:</b> <math>(\text{max} - \text{min}) / 2.0</math></p>
<b>Normal</b>	<p>A value is produced by iteratively generating numbers using a boundless normal distribution and rejecting any values that lie outside of the allowed range.</p> <p><b>Min:</b> The minimum possible value.  <b>Max:</b> The maximum possible value.  <b>Mean:</b> The mean of the normal distribution.  <b>Std:</b> The standard deviation.</p> <p><b>Resultant Mean:</b> Mean</p>
<b>Triangular</b>	<p>The distribution will produce a random number between the minimum and maximum value according to a regular triangular distribution, with values being more likely around the mode.</p> <p><b>Min:</b> The minimum possible value.  <b>Max:</b> The maximum possible value.  <b>Mode:</b> The mode of the triangular distribution.</p> <p><b>Resultant Mean:</b> <math>(\text{Min} + \text{Max} + \text{Mode}) / 3.0</math></p>
<b>Log Normal</b>	<p>A value is produced by iteratively generating numbers using the given boundless log normal distribution, shifting the resulting values by the minimum, and rejecting any values that are greater than the maximum.</p> <p><b>Shift:</b> The minimum possible value. This value is added to the number produced by a regular log normal distribution.  <b>Max:</b> The maximum possible value.  <b>Mu:</b> The mean of the log of the distribution.  <b>Sigma:</b> The standard deviation of the log of the distribution.</p> <p><b>Resultant Mean:</b> <math>\text{Min} + e^{(\text{Mu} + ((\text{Sigma}^2) / 2))}</math></p>
<b>Exponential</b>	<p>A value is produced by iteratively generating numbers using the given boundless exponential distribution, shifting the resulting values by the minimum, and rejecting any values that are greater than the maximum.</p> <p><b>Shift:</b> The minimum possible value. This value is added to the number produced by a regular exponential distribution.  <b>Max:</b> The maximum possible value.  <b>1 / Lambda:</b> The average (inverse of the lambda rate).</p> <p><b>Resultant Mean:</b> <math>\text{Min} + (1.0 / \text{Lambda})</math></p>

## 4.5 Analysis & Reporting

Analysis is accomplished through the creation and evaluation of [graph](#), [map](#), and [table](#) objects. These objects rely on simulation results made available through one or more [simulation run](#) objects. For a comprehensive description of all the available analysis functions please refer to the [Analysis Objects](#) section.

## 4.5.1 Observers

Observer windows are used during playback to display information about an object or agent. The information is taken from the simulation run and updated based on the current playback time. Observers can be shown by right-clicking on a selected object or agent and choosing "Observer".

The supported observers include:

- [Agent Observer](#)
- [Event Observer](#)
- [Tally Observer](#)

### 4.5.1.1 Agent Observer

The Agent Observer window is used to view properties about a particular agent from a recorded simulation run. The window can be shown by right-clicking on an agent and choosing 'Observe', or by selecting an agent and using the main window's View -> Observer Agent menu.

The focus button at the top of the window will focus the window on the currently selected agent. The agent ID and simulation run are displayed immediately below the focus button. The target button to the right of the simulation run can be used to find the agent in the scene. It is possible to change the simulation run or manually enter a new agent ID. Manually entering an ID is useful when trying to find agents mentioned by errors during the simulation.

Right-click on an object in the agent's route to find the object in the scene or jump to the time when the agent enters or leaves the object.

Agent Observer	
<b>Created by</b>	The event which created the agent.
<b>Profile</b>	The <a href="#">profile</a> used by the agent.
<b>Network</b>	The <a href="#">network</a> used by the agent.
<b>Start time</b>	The time at which the agent entered the simulation.
<b>End time</b>	The time at which the agent exited the simulation.
<b>Age</b>	The amount of time the agent has been in the simulation.
<b>Avatar</b>	The <a href="#">avatar</a> used by the agent.
<b>Density</b>	The density currently experienced by the agent ( ppl / m <sup>2</sup> )
<b>Speed</b>	The current speed of the agent ( m/s )
<b>State</b>	<p>The state of the agent:</p> <p><b>Waiting:</b> The agent is executing a wait task.  <b>In Transit:</b> The agent is moving freely towards its target.  <b>Queuing:</b> The agent is queuing for its target.  <b>Waiting for Access:</b> The agent is waiting for access to its target (waiting for a gate to open).  <b>Pre-Contact Wait:</b> The agent is queuing for a server.  <b>In-Contact:</b> The agent is being processed by a server.  <b>Post-Contact Wait:</b> The agent is being held by a server until there is</p>

	capacity downstream in the process chain.
<b>Target</b>	The object to which the agent is moving.
<b>Tokens</b>	A list of <a href="#">tokens</a> held by the agent over the course of its life. Tokens currently held by the agent are indicated with an arrow.
<b>Route</b>	An ordered list of objects on which the agent walked over the course of its life. The object that the agent is currently on is marked with an arrow.
<b>Servers</b>	An ordered list of the <a href="#">servers</a> used by the agent over the course of its life. The server that the agent is currently on or moving towards is marked with an arrow.
<b>Actions</b>	<p>A list of all <a href="#">actions</a> applied to the agent, identified by the source of the action. Actions that have already been applied are marked with a check. Actions that are being applied in the current frame are marked with an arrow.</p> <p>The first column is the object which applied the action. The second column further clarifies the source (see <a href="#">Where to Use Actions</a>). Most values in the second column are self explanatory, with the exception of <b>'Zone Event'</b>. When an agent receives an action from a <a href="#">broadcast</a> event as it fires, it is recorded as Broadcast Event'. However, if the broadcast targets a zone, and the agent receives the action as it enters the zone while the event is active, the action is recorded as 'Zone Event' and references the zone instead of the broadcast event.</p>

**4.5.1.2 Event Observer**

The Event Observer window is used to view the activity of a particular event from a recorded simulation run. The window can be shown by right-clicking on an event and choosing 'Observe'.

The refresh button at the top of the window will reload data about the event from the chosen simulation run. The chosen event and simulation run are displayed immediately below the refresh button.

Right-click on a time in the event details to focus the playback on that time.

Event Observer	
<b>Type</b>	The event which created the agent.
<b>State</b>	The state of all possible types of activation (create agents, open gate, open server, etc.).
<b>Time</b>	A list of times represented by the event (used by Time events only).
<b>Create Agents</b>	The intervals over which the event was creating agents. Note that this does not mean that agents were actually being created, but that the event was in agent creation mode.
<b>Open Gate</b>	The intervals over which the event was opening one or more gates.
<b>Close Gate</b>	The intervals over which the event was closing one or more gates.

<b>Open Server</b>	The intervals over which the event was opening one or more servers.
<b>Close Server</b>	The intervals over which the event was closing one or more servers.
<b>Action</b>	The intervals over which the event was applying an action (other than any agent initialization actions). This is currently only used by the <a href="#">Broadcast</a> and <a href="#">Timetable</a> events.

#### 4.5.1.3 Tally Observer

The Tally Observer window is used to view the changing value of one or more tally objects. The window can be shown by right-clicking on one or more tally objects and choosing 'Observe'.

The refresh button at the top of the window will reload data about the tallies from the chosen simulation run.

## 4.5.2 Transition

The concept of a transition is used within MassMotion to mean a point in time when an agent moves from one location or state to another. Transitions can be used to define a [flow count graph](#) or an 'At transition' [agent filter](#). Transitions can also be used in the Transition Count [trigger](#).

Transitions are defined by selecting a transition type in a drop-down menu. Depending on the type of transition selected, other entry fields will be made available to define the transition.

### Collections in transitions

[Collections](#) can be used to define complex transitions.

Transition Types	
<b>At portal</b>	<p>Transition occurs when an agent enters at a portal, exits at a portal, or reaches a portal that they have been given as a target (such as by a 'Seek Portal' task).</p> <p>With a collection: Agents at any portal within the collection are at the transition.</p>
<b>Between objects</b>	<p>Transition occurs when an agent steps immediately <b>from</b> one given object <b>to</b> a second given object.</p> <p>With a collection as the <b>from</b> object: Agents stepping off any walkable in the collection to the <b>to</b> object are at the transition.</p> <p>With a collection as the <b>to</b> object: Agents stepping off the <b>from</b> object to any walkable in the collection are at the transition.</p> <p>With collections as both <b>from</b> and <b>to</b> objects: Agents stepping off any object in the <b>from</b> collection onto any walkable in the <b>to</b> collection are at the transition. The same collection can be used for both <b>from</b> and <b>to</b> to track internal transitions.</p>
<b>Crossing cordon</b>	<p>Transition occurs when an agent passes through a given <a href="#">cordon</a>.</p> <p>With a collection: Agents crossing any cordon in the collection are at the transition.</p>

Transition Types	
<b>Entering area</b>	<p>Transition occurs when an agent enters a given <a href="#">area</a>, or enters the simulation in the given area.</p> <p>With a collection: Agents entering the areas in the collection are at the transition. Internal transitions between areas in the collection are not counted.</p>
<b>Entering simulation at</b>	<p>Transition occurs as soon as agent enters simulation from a given portal.</p> <p>With a collection: Agents entering the simulation at any of the portals in the collection are at the transition.</p>
<b>Exiting area</b>	<p>Transition occurs when an agent exits a given area (possibly by exiting the simulation).</p> <p>With a collection: Agents exiting the areas in the collection are at the transition. Internal transitions between areas in the collection are not counted.</p>
<b>Exiting simulation at</b>	<p>Transition occurs when an agent exits the simulation at the same time as reaching a portal. This usually occurs when simply exiting at a destination, but may also happen when removed by an action as an agent reaches a portal they have been given as a target.</p> <p>With a collection: Agents exiting the simulation at any of the portals in the collection are at the transition.</p>
<b>Server begin</b>	<p>Transition occurs when an agent enters the pre-contact stage of a given <a href="#">server</a>.</p> <p>With a collection: Agents entering the pre-contact stage of any server in the collection are at the transition.</p>
<b>Server end</b>	<p>Transition occurs when an agent leaves a given server.</p> <p>With a collection: Agents leaving any server in the collection are at the transition.</p>

### 4.5.3 LOS Colour Mapping

Colour mapping describes how density values are converted into colours in [maps](#) and [graphs](#). Density colour mapping can also be used to colour agents during playback through the [simulation run](#). Values are taken from standard Fruin and IATA (International Air Transport Association) LOS mappings. When used in maps, black is used to indicate 'no data' (no agent walked in that area).

LOS Colour Mapping Values			
<b>Fruin Walkways</b>	Area of circle used to calculate density: 3.24m <sup>2</sup> .		
	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)

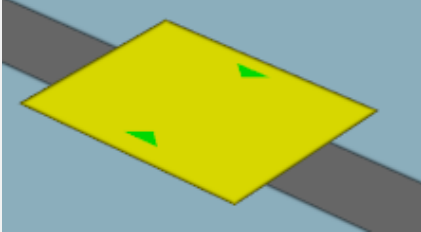
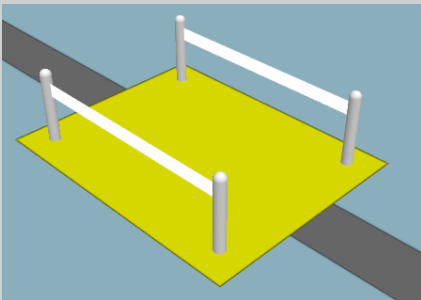
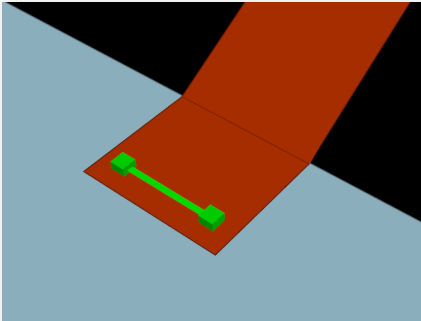
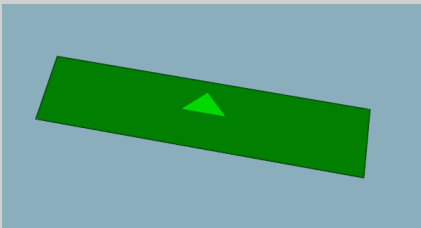
	<table border="1"> <tbody> <tr> <td>A</td> <td><math>x \leq 0.309</math></td> <td><math>x \geq 3.24</math></td> <td></td> </tr> <tr> <td>B</td> <td><math>0.309 &lt; x \leq 0.431</math></td> <td><math>3.24 &gt; x \geq 2.32</math></td> <td></td> </tr> <tr> <td>C</td> <td><math>0.431 &lt; x \leq 0.719</math></td> <td><math>2.32 &gt; x \geq 1.39</math></td> <td></td> </tr> <tr> <td>D</td> <td><math>0.719 &lt; x \leq 1.075</math></td> <td><math>1.39 &gt; x \geq 0.93</math></td> <td></td> </tr> <tr> <td>E</td> <td><math>1.075 &lt; x \leq 2.174</math></td> <td><math>0.93 &gt; x \geq 0.46</math></td> <td></td> </tr> <tr> <td>F</td> <td><math>2.174 &lt; x</math></td> <td><math>0.46 &gt; x</math></td> <td></td> </tr> </tbody> </table>	A	$x \leq 0.309$	$x \geq 3.24$		B	$0.309 < x \leq 0.431$	$3.24 > x \geq 2.32$		C	$0.431 < x \leq 0.719$	$2.32 > x \geq 1.39$		D	$0.719 < x \leq 1.075$	$1.39 > x \geq 0.93$		E	$1.075 < x \leq 2.174$	$0.93 > x \geq 0.46$		F	$2.174 < x$	$0.46 > x$					
A	$x \leq 0.309$	$x \geq 3.24$																											
B	$0.309 < x \leq 0.431$	$3.24 > x \geq 2.32$																											
C	$0.431 < x \leq 0.719$	$2.32 > x \geq 1.39$																											
D	$0.719 < x \leq 1.075$	$1.39 > x \geq 0.93$																											
E	$1.075 < x \leq 2.174$	$0.93 > x \geq 0.46$																											
F	$2.174 < x$	$0.46 > x$																											
<b>Fruin Stairways</b>	<p>Area of circle used to calculate density: 1.81m<sup>2</sup>.</p> <table border="1"> <thead> <tr> <th>LOS</th> <th>Density (person/m<sup>2</sup>)</th> <th>Space (m<sup>2</sup>/person)</th> <th>Colour</th> </tr> </thead> <tbody> <tr> <td>A</td> <td><math>x \leq 0.541</math></td> <td><math>x \geq 1.85</math></td> <td></td> </tr> <tr> <td>B</td> <td><math>0.541 &lt; x \leq 0.719</math></td> <td><math>1.85 &gt; x \geq 1.39</math></td> <td></td> </tr> <tr> <td>C</td> <td><math>0.719 &lt; x \leq 1.076</math></td> <td><math>1.39 &gt; x \geq 0.93</math></td> <td></td> </tr> <tr> <td>D</td> <td><math>1.076 &lt; x \leq 1.539</math></td> <td><math>0.93 &gt; x \geq 0.65</math></td> <td></td> </tr> <tr> <td>E</td> <td><math>1.539 &lt; x \leq 2.702</math></td> <td><math>0.65 &gt; x \geq 0.37</math></td> <td></td> </tr> <tr> <td>F</td> <td><math>2.702 &lt; x</math></td> <td><math>0.37 &gt; x</math></td> <td></td> </tr> </tbody> </table>	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour	A	$x \leq 0.541$	$x \geq 1.85$		B	$0.541 < x \leq 0.719$	$1.85 > x \geq 1.39$		C	$0.719 < x \leq 1.076$	$1.39 > x \geq 0.93$		D	$1.076 < x \leq 1.539$	$0.93 > x \geq 0.65$		E	$1.539 < x \leq 2.702$	$0.65 > x \geq 0.37$		F	$2.702 < x$	$0.37 > x$	
LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour																										
A	$x \leq 0.541$	$x \geq 1.85$																											
B	$0.541 < x \leq 0.719$	$1.85 > x \geq 1.39$																											
C	$0.719 < x \leq 1.076$	$1.39 > x \geq 0.93$																											
D	$1.076 < x \leq 1.539$	$0.93 > x \geq 0.65$																											
E	$1.539 < x \leq 2.702$	$0.65 > x \geq 0.37$																											
F	$2.702 < x$	$0.37 > x$																											
<b>Fruin Platforms (Queuing)</b>	<p>Area of circle used to calculate density: 1.21m<sup>2</sup>.</p> <table border="1"> <thead> <tr> <th>LOS</th> <th>Density (person/m<sup>2</sup>)</th> <th>Space (m<sup>2</sup>/person)</th> <th>Colour</th> </tr> </thead> <tbody> <tr> <td>A</td> <td><math>x \leq 0.826</math></td> <td><math>x \geq 1.21</math></td> <td></td> </tr> <tr> <td>B</td> <td><math>0.826 &lt; x \leq 1.075</math></td> <td><math>1.21 &gt; x \geq 0.93</math></td> <td></td> </tr> <tr> <td>C</td> <td><math>1.075 &lt; x \leq 1.538</math></td> <td><math>0.93 &gt; x \geq 0.65</math></td> <td></td> </tr> <tr> <td>D</td> <td><math>1.538 &lt; x \leq 3.571</math></td> <td><math>0.65 &gt; x \geq 0.28</math></td> <td></td> </tr> <tr> <td>E</td> <td><math>3.571 &lt; x \leq 5.263</math></td> <td><math>0.28 &gt; x \geq 0.19</math></td> <td></td> </tr> <tr> <td>F</td> <td><math>5.263 &lt; x</math></td> <td><math>0.19 &gt; x</math></td> <td></td> </tr> </tbody> </table>	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour	A	$x \leq 0.826$	$x \geq 1.21$		B	$0.826 < x \leq 1.075$	$1.21 > x \geq 0.93$		C	$1.075 < x \leq 1.538$	$0.93 > x \geq 0.65$		D	$1.538 < x \leq 3.571$	$0.65 > x \geq 0.28$		E	$3.571 < x \leq 5.263$	$0.28 > x \geq 0.19$		F	$5.263 < x$	$0.19 > x$	
LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour																										
A	$x \leq 0.826$	$x \geq 1.21$																											
B	$0.826 < x \leq 1.075$	$1.21 > x \geq 0.93$																											
C	$1.075 < x \leq 1.538$	$0.93 > x \geq 0.65$																											
D	$1.538 < x \leq 3.571$	$0.65 > x \geq 0.28$																											
E	$3.571 < x \leq 5.263$	$0.28 > x \geq 0.19$																											
F	$5.263 < x$	$0.19 > x$																											
<b>IATA Wait/Circulate</b>	<p>Area of circle used to calculate density: 2.70m<sup>2</sup>.</p> <table border="1"> <thead> <tr> <th>LOS</th> <th>Density (person/m<sup>2</sup>)</th> <th>Space (m<sup>2</sup>/person)</th> <th>Colour</th> </tr> </thead> <tbody> <tr> <td>A</td> <td><math>x \leq 0.370</math></td> <td><math>x \geq 2.70</math></td> <td></td> </tr> <tr> <td>B</td> <td><math>0.370 &lt; x \leq 0.435</math></td> <td><math>2.70 &gt; x \geq 2.30</math></td> <td></td> </tr> <tr> <td>C</td> <td><math>0.435 &lt; x \leq 0.526</math></td> <td><math>2.30 &gt; x \geq 1.90</math></td> <td></td> </tr> <tr> <td>D</td> <td><math>0.526 &lt; x \leq 0.667</math></td> <td><math>1.90 &gt; x \geq 1.50</math></td> <td></td> </tr> <tr> <td>E</td> <td><math>0.667 &lt; x \leq 1.00</math></td> <td><math>1.50 &gt; x \geq 1.00</math></td> <td></td> </tr> <tr> <td>F</td> <td><math>1.00 &lt; x</math></td> <td><math>1.00 &gt; x</math></td> <td></td> </tr> </tbody> </table>	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour	A	$x \leq 0.370$	$x \geq 2.70$		B	$0.370 < x \leq 0.435$	$2.70 > x \geq 2.30$		C	$0.435 < x \leq 0.526$	$2.30 > x \geq 1.90$		D	$0.526 < x \leq 0.667$	$1.90 > x \geq 1.50$		E	$0.667 < x \leq 1.00$	$1.50 > x \geq 1.00$		F	$1.00 < x$	$1.00 > x$	
LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour																										
A	$x \leq 0.370$	$x \geq 2.70$																											
B	$0.370 < x \leq 0.435$	$2.70 > x \geq 2.30$																											
C	$0.435 < x \leq 0.526$	$2.30 > x \geq 1.90$																											
D	$0.526 < x \leq 0.667$	$1.90 > x \geq 1.50$																											
E	$0.667 < x \leq 1.00$	$1.50 > x \geq 1.00$																											
F	$1.00 < x$	$1.00 > x$																											

#### 4.5.4 Decorations

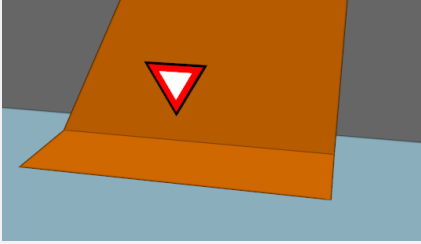
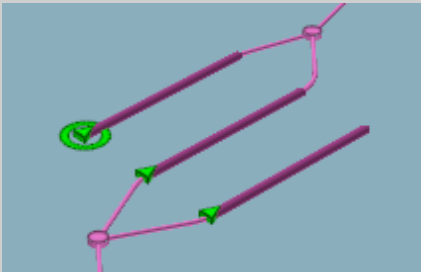
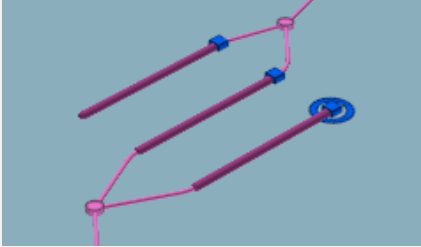
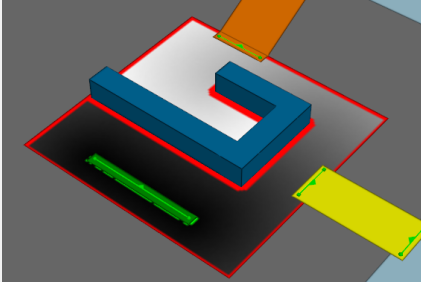
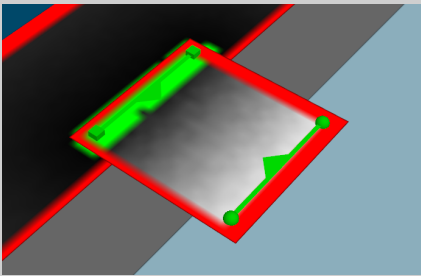
Decorations show properties of objects in the scene.

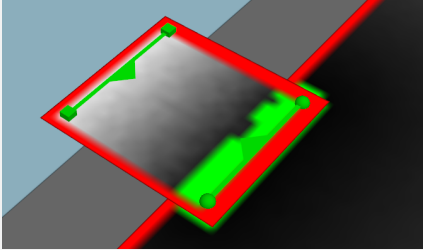
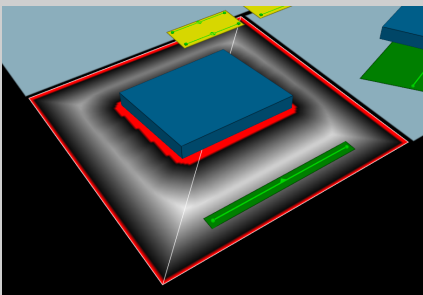
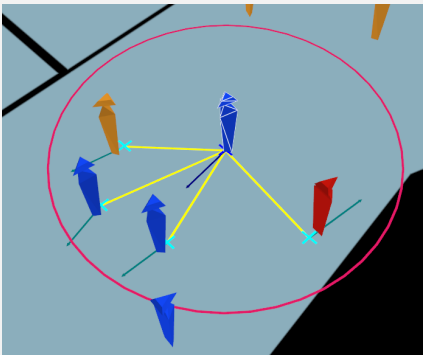
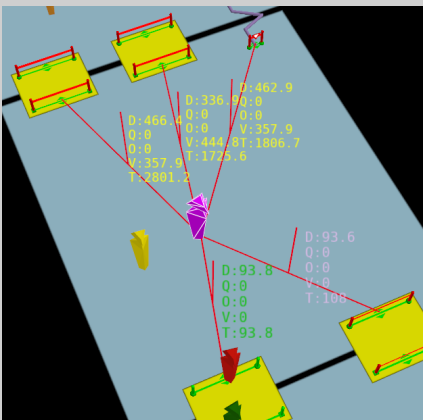
- General decorations appear on various scene objects.
- [Surface Map](#) decorations can be applied to surface objects upon which agents can walk. Surface map decorations are not updated live and do not respond to geometry changes. Hide and show again to refresh them.
- Debug simulation decorations appear on agents during a debug simulation.

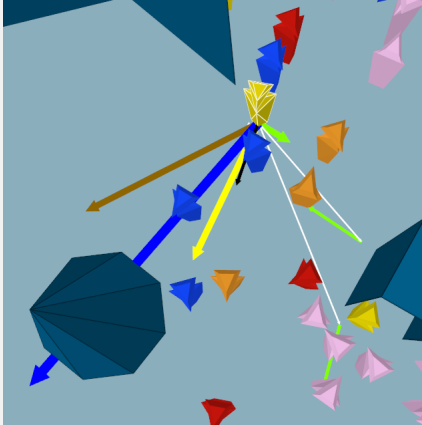
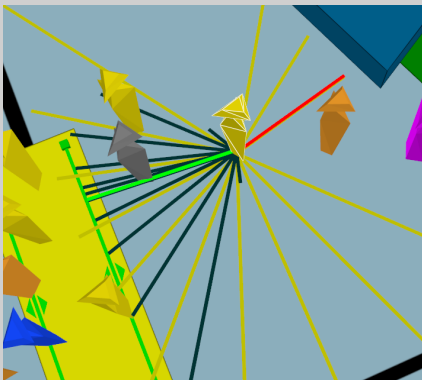
They can be toggled individually by right clicking the relevant object/agent and accessing the 'Display' sub-menu. General and debug simulation decorations can be toggled globally in a [Scene View](#)'s visibility options.

General Decorations		
<p><b>Direction Arrows</b></p>		<p>Displays whether a connection object is unidirectional or bidirectional, allowing agents to cross in each direction.</p> <p>The triangle will be shown on the goal line(s) where agents can cross onto the object. The image on the left, therefore, shows a bidirectional link.</p> <p>See <a href="#">Links</a> and <a href="#">Connection Objects</a> for more information.</p>
<p><b>Gate</b></p>		<p>Displays whether a connection object is gated. During authoring and playback the symbol is white. In simulation, the gate is coloured red when closed, yellow when open for some agents and green when open for all agents. See <a href="#">Links</a> and <a href="#">Connection Objects</a> for more information.</p>
<p><b>Goal Line</b></p>		<p>Displays the goal lines of <a href="#">portals</a> and <a href="#">connection objects</a>. Goal lines are coloured green when connected to an underlying floor and grey otherwise. Connection objects have two goal lines, one with box terminals and one with balls. See <a href="#">Connecting Objects Together</a> for more information.</p>
<p><b>Portal Start Angle</b></p>		<p>Displays the start angle for <a href="#">portals</a>. Agents created at this portal will be created facing this direction.</p>



<p><b>Priority Flow</b></p>		<p>Displays which side will yield when agents approach a connection object from both sides.</p>
<p><b>Server Entry Arrow</b></p>		<p>Displays the entry point and direction of a <a href="#">server</a> as a green triangle. A ring will appear if the server is also the start of a <a href="#">process chain</a>.</p>
<p><b>Server End Point</b></p>		<p>Displays the end point of a <a href="#">server</a> as a blue box. A ring will appear if the server is also the end of a <a href="#">process chain</a>.</p>
<p><b>Surface Map Decorations</b></p>		
<p><b>Approach Map</b></p>		<p>Displays the approach map to a <a href="#">portal</a> goal line on its floor.</p> <p>See <a href="#">Determining Walkable Space</a> for more information.</p>
<p><b>Ball Approach Map</b></p>		<p>Displays the approach map to the ball goal line of a <a href="#">connection object</a> on the floor and the object itself.</p> <p>See <a href="#">Determining Walkable Space</a> for more information.</p>

<p><b>Box Approach Map</b></p>		<p>Displays the approach map to the box goal line of a <a href="#">connection object</a> on the floor and the object itself.</p> <p>See <a href="#">Determining Walkable Space</a> for more information.</p>
<p><b>Obstacle Map</b></p>		<p>Displays the obstacle map for surfaces.</p> <p>See <a href="#">Determining Walkable Space</a> for more information.</p>
<p><b>Debug Simulation Decorations</b></p>		
<p><b>Neighbourhood</b></p>		<p>Displays an agent's neighbourhood in debug simulation. See <a href="#">Simulation Scene View</a> for more information.</p>
<p><b>Route Costing</b></p>		<p>Displays how agents perceive <a href="#">the network</a> in debug simulation. See the <a href="#">Simulation Scene View</a> for more information.</p>

<b>Social Forces</b>		Displays the forces which affect an agent's movement in debug simulation. See the <a href="#">Simulation Scene View</a> for more information.
<b>Surface Probe</b>		Displays an agent's awareness of its immediate surrounding in debug simulation. See the <a href="#">Simulation Scene View</a> for more information.

## 4.6 Working with the Viewer

The MassMotion Viewer is a free application for viewing previously recorded simulation runs. It allows a project to be shared with others in a read-only controlled form. Once the viewer has been downloaded and installed it can open and play any MassMotion database file. To download the viewer visit the [Oasys product page](#).

### Creating Projects for the Viewer

The viewer can open and play back any raw MassMotion database file (mmdb) created by a simulation run. The [simulation slice export](#) tool can generate presentation versions of these files. Presentation files may strip away much of the data and include only a particular time range or they may use bookmarks to control what the user sees or highlight particular times or areas of interest.

### Using Bookmarks

[Bookmarks](#) are an excellent way to guide the process of viewing a project. They can also be used to control what is shown.

Bookmarks can:

- Focus on a particular area at a particular time
- Change object visibility (switch between exterior/interior view by showing/hiding walls)
- Configure the 3d scene view (show/hide gate indicators, clock overlay, etc.)
- Hide unwanted geometry (objects cannot be shown through the viewer except using bookmarks).

Users cannot directly show objects in the viewer. Bookmarks are the only way to control object visibility. If it is important to show both the exterior and interior of a building, two bookmarks could be provided to allow the user to switch between the two (exterior walls shown vs hidden). Because

bookmarks are the only way to show objects, any objects that are hidden in all bookmarks will always be hidden from the user.

**Locked Viewer File (MMV)**

The MassMotion database (mmdb) file can be opened by the viewer or by MassMotion. When distributing a project via an mmdb file it is possible that others could gain access to the original project, make modifications, and run additional simulations. The MassMotion Viewer (mmv) file on the other hand, has had all events stripped from the project and cannot be opened in MassMotion or Flow. A MassMotion Viewer file can only be opened for playback in the MassMotion Viewer.

---

# Index

## - B -

bookmark 69

## - G -

Graph 243

Graph Structure 242

Endnotes 2... (after index)

Back Cover