

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: <%EMAIL\_ADDRESS%>  
Website: <%HOME\_URL%>

# MassMotion

© 2015 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: September 2015

# Table of Contents

<b>Part I Welcome to MassMotion</b>	<b>7</b>
<b>Part II What's New in 8</b>	<b>9</b>
<b>Part III User Guide</b>	<b>12</b>
<b>1 Installation</b>	<b>12</b>
System Requirements	12
Licensing	12
MassMotion & Flow	13
<b>2 How MassMotion Works</b>	<b>14</b>
The Scene	14
People as Agents	19
<b>3 User Interface Overview</b>	<b>24</b>
Main Window	24
Objects	25
Selection	26
<b>4 Project Workflow</b>	<b>26</b>
Authoring	26
Simulation	51
Analysis	52
<b>5 Troubleshooting</b>	<b>53</b>
Auditing	53
Validation	53
Observing Agents	54
Finding Object References	54
Debugging a Simulation	54
Using Analysis to Diagnose Issues	55
<b>Part IV Reference</b>	<b>57</b>
<b>1 Project</b>	<b>57</b>
Project Settings	57
Merging Projects	58
Importing and Exporting Objects	58
Files	59
Upgrading Older Projects	59
<b>2 User Interface</b>	<b>61</b>
Main Window	61
Editing Object Properties	72
Choosing Objects	74
Working with Colours	77
Working with Time	79
Issue Window	81
Keyboard and Mouse Controls	81
Application Preferences	83
<b>3 Objects</b>	<b>84</b>



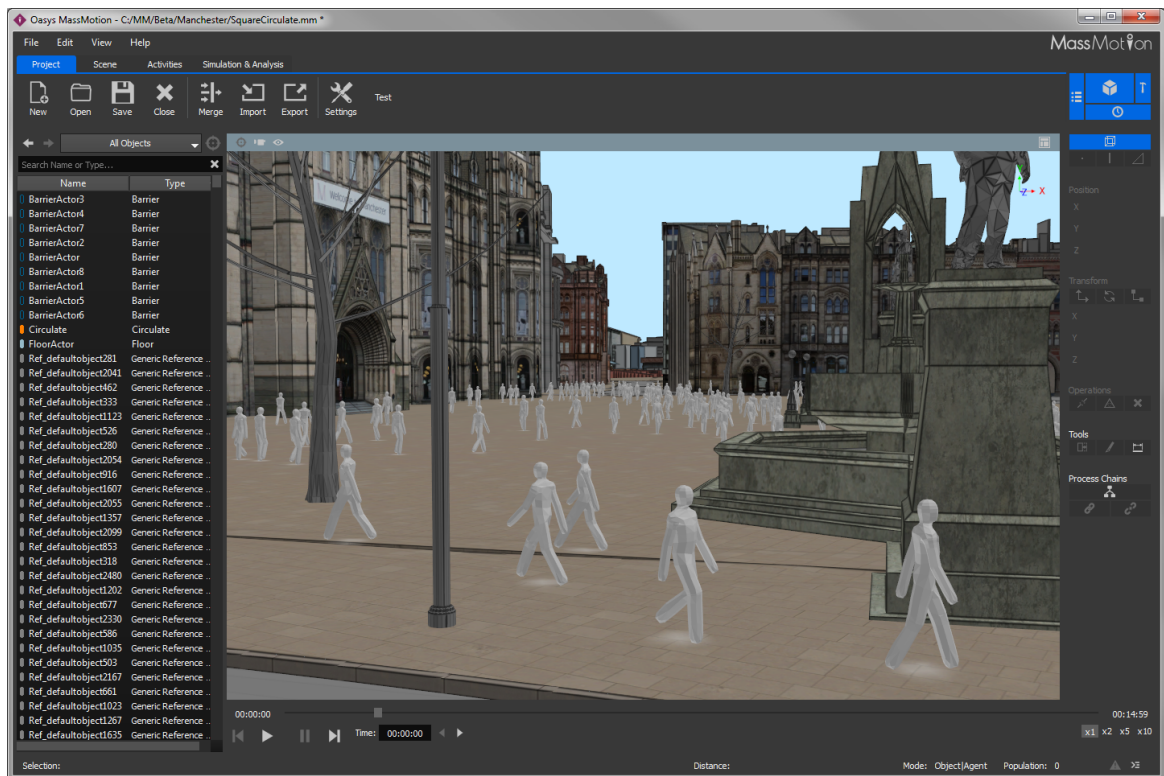
---

Scene Objects .....	84
Activity Objects .....	119
Analysis Objects .....	156
<b>4 Simulation .....</b>	<b>196</b>
Running a Simulation .....	196
Console Simulation Window .....	198
Debug Simulation Window .....	198
Running from the Command Line .....	205
Generated Simulation Files .....	207
Randomness .....	208
<b>5 Analysis &amp; Reporting .....</b>	<b>211</b>
Agent Observer .....	211
Areas .....	212
Transition .....	212
LOS Colour Mapping .....	214
Alembic Export .....	215
Agent Position Export .....	216
Movie and Image Export .....	217

# 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 8

### Migrating from Softimage Workbench

MassMotion no longer requires Softimage. To open legacy emdl projects they must be exported to mmxsi using the MassMotion 7 workbench. Some workbench objects will not be editable in MassMotion 8, but can be manually converted to equivalent objects. See [Upgrading Projects](#) for more information.

### Actions

- New interface for authoring agent [actions and tests](#).
- New agent tests: "Entered at" tests whether an agent entered the simulation via a portal, "Initial goal" tests whether an agent was given a portal as an initial goal.
- Right-click on an object and choose 'Find' to discover whether or not it is used in any actions.

### Servers and Process Chains

- New interface for authoring [server](#) objects.
- New interface for connecting servers together into [process chains](#) directly through the 3d scene.

### Timetables

- New interface for authoring [timetable](#) objects.

### General Events

- New agent demand types: "Instant" creates all agents in a single burst, "Table" provides an editable table of duration/population pairs (see [journey](#)).
- New agent colouring: complex rules colour agents by origin/destination/profile/event (see [journey](#)).
- New dwell rules for the [circulate](#) event allow for different wait times for each circulation portal.
- Support for weighted collection of profiles to generate a range of population types from a single event.

### Analysis Objects

- New [region density](#) graph: plots the average density within one or more regions over time.
- New [agent speed ratio](#) graph: plots the number of agents in different speed ratio ranges over time.
- New [dynamic path](#) map: shows dynamic agent trails with configurable length and opacity.
- All maps can display an optional colour value legend.
- Review when actions were applied by [observing](#) an agent during playback.

### Analysis Filters

- Improved editing through drag and drop copying and re-ordering of sub-components.
- New "Has end state": filters agents by the manner in which they left the simulation (exited with success, exited with error, still in scene at end of simulation).

### Rendering

- The playback of a simulation run can be exported to [alembic animation files](#) for rendering in 3rd party applications like Softimage or 3ds Max.
- Create, import, or edit the static [avatar](#) objects used to represent agents.

### Application Preferences

- Set default values for the various movie/image export settings (available through "Edit" menu).

### Full Release Notes

- Added authoring of servers and process chains (hotkey 'q')
- Added authoring of actions
- Added authoring of timetables
- Added authoring of action event

- Added birth actions to journey, circulate, vehicle, and evacuate events
- Added birth action to profile
- Added agent action tests 'Entered At' and 'Initial Goal'
- Added ability to migrate SI workbench schedules and evacuation events to editable MassMotion objects
- Added authoring of agent avatars
- Added actions field in agent observer window for viewing agent action history
- Added ability to specify weighted collection of profiles in events
- Added 'Instant' event demand type for creation of agents all in a single frame
- Added 'Table of intervals' event demand type for complex arrival patterns
- Added 'Color Scheme' option for colouring agents in events by entrance/exit
- Added ability to specify dwell times per portal in circulate events
- Added alembic export for rendering of simulation playback in other applications
- Added new 'Has end state' agent filter for isolating error agents during playback/analysis
- Added colour legends as overlay when displaying an analysis map
- Added new region density analysis graph
- Added improved editing of agent filters through copy/paste and drag/drop (re-order items by dragging the 'handle' to the right of the filter)
- Added orthogonal 'Side' camera viewpoint (can be rotated horizontally)
- Added camera rotation snapping (hold shift to snap to the nearest 15 degree rotation)
- Added application preferences (from 'Edit' menu) for default movie export settings
- Added support for virtual paths
- Added ability to rename objects directly from the main window using F2
- Added opacity parameter to agent path map and modified its behaviour to take agent colour from the source simulation run
- Added dynamic path map for showing agent 'trails'
- Added speed ratio graph for showing counts of agents with different actual/desired speed ratios
- Added support for importing reference geometry as ramps (including auto-detection of IfcRamp objects)
- Added ability to create normalized graphs (stacked bars or areas that sum to 100% instead of an absolute value).
- Changed simulation run objects to point to database file rather than results folder
- Improved 'Find' object context menus to provide more information
- Improved options for circulation end conditions
- Improved behaviour of mesh simplification
- Improved performance when closing projects with thousands of objects.
- Improved visibility of measuring tool points
- Fixed bug where vehicle arrivals within 10 seconds of simulation start would not produce alighting agents for that arrival.
- Fixed saved views for orthogonal viewpoints (top-down, front, etc.)
- Fixed agents losing their way when seeking servers on a virtual floor
- Fixed audit false positives in objects with disabled properties
- Fixed issue with playback agents not hiding when on hidden floors
- Fixed inconsistencies in how multi run simulation run names were generated
- Fixed resize behaviour of object choosers when widget resizes
- Fixed false positive issue raised when DefaultProfile deleted
- Fixed false positive error/warning in simulation console when overwriting old database file
- Fixed issue where where importing from an old IFC file or one with invalid geometry could cause a crash
- Fixed issue where some types of agent filters did not work with agent density graphs
- Fixed issue where server population graphs could not be toggled between combined series/series per server

# 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 introductory 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>. Once the download has completed users can double click on the .msi file to launch the installer. Users should follow the on screen prompts, supplying information as necessary.

#### 3.1.1 System Requirements

MassMotion is a high performance 64bit multi-threaded application that is capable of effectively utilizing very high specification equipment. It is particularly important when running simulations that have very high concurrent agent counts (25,000+) that CPU core counts are maximised for processing speed and that solid state storage is provided for efficient database transactions. GPU performance is important for highly detailed 3D environments and for 3D playback of high agent count environments.

**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 64bit 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 site based. For information on how to purchase MassMotion please visit <http://www.oasys-software.com> for sales information or to have a member of the commercial team contact you.

**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 to function, although an internet connection is required for initial activation and transfer.

To activate or move your license see <http://www.oasys-software.com/support/licensing.html> for details.



### Shared & Virtual Licenses

Shared licenses may be utilized by any machine with MassMotion installed within an organization, provided that the total number of concurrently active MassMotion users does not exceed the number of shared licenses within an organization. It is also the required licensing option for virtual machines. This “floating” license scheme is typical of CAD packages and other high-end technical software. It requires an active internet connection to function.

To activate your license see <http://www.oasys-software.com/support/licensing.html> for details.

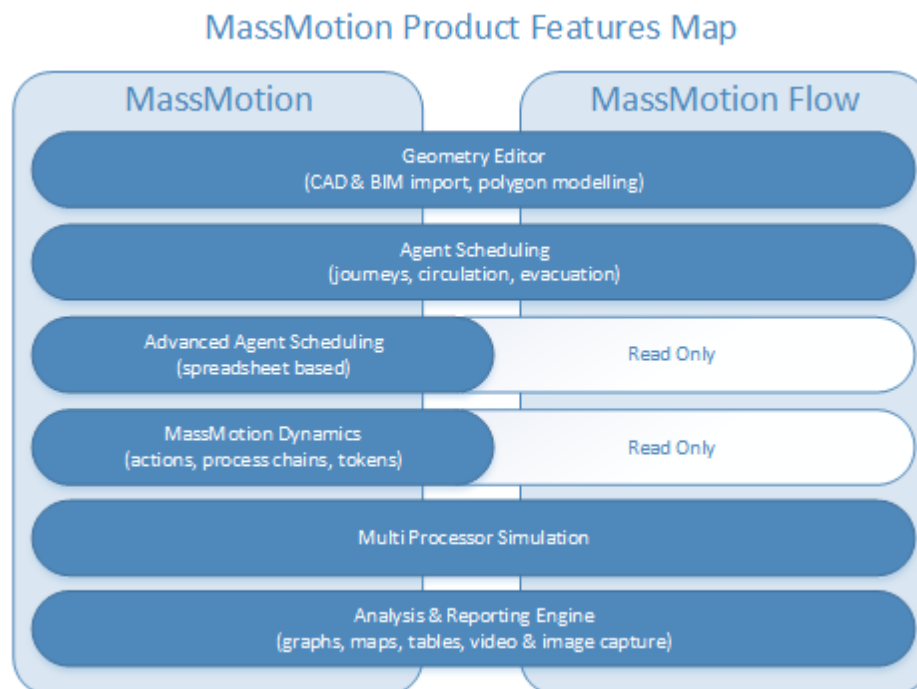
### University Licenses

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 the machine is connected to the internet through the designated network.

To activate your license see [http://www.oasys-software.com/company/university\\_licensing.html](http://www.oasys-software.com/company/university_licensing.html) for details.

## 3.1.3 MassMotion & Flow

Oasys offers two crowd simulation products: MassMotion and Flow. The project files (.mm) and results databases (.mmdb) 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..



## 3.2 How MassMotion Works

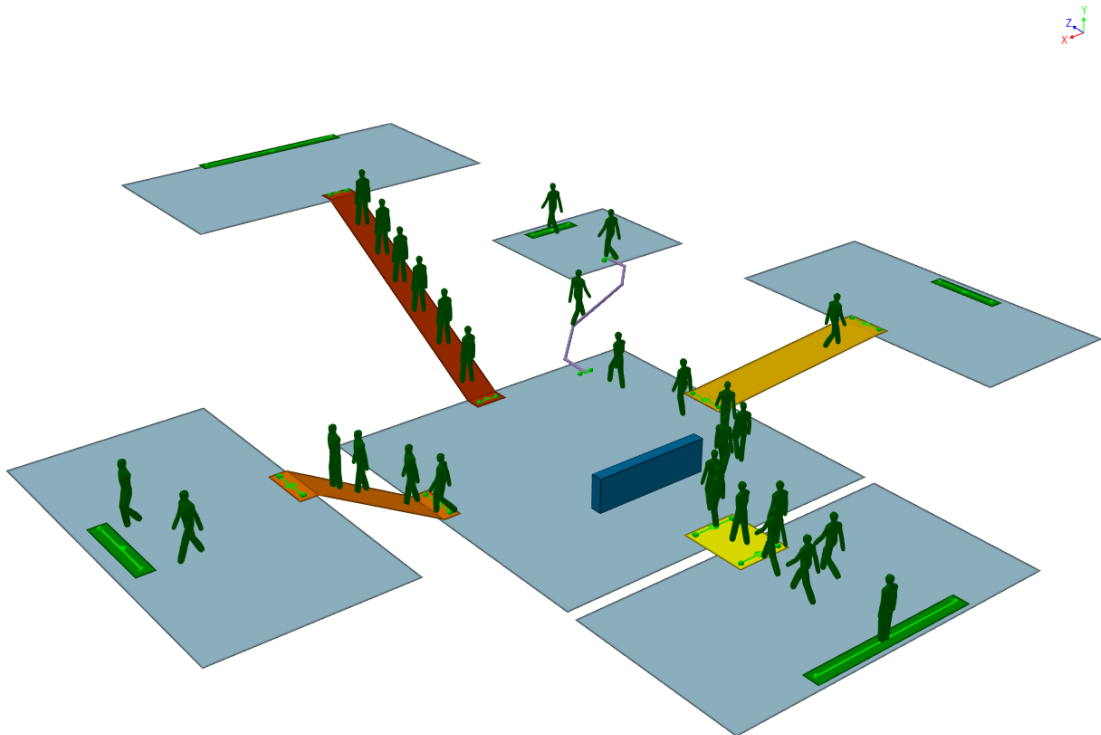
This section provides a brief introduction to some of the main concepts in MassMotion. [The Scene](#) covers topics related to the simulation environment, then [People as Agents](#) describes the representation of people.

### 3.2.1 The Scene

MassMotion models real world spaces by breaking those spaces down 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.

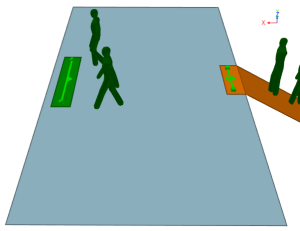
The basic elements of a scene are: [floor](#), [link](#), [stair](#), [ramp](#), [escalator](#), [path](#), [portal](#), and [barrier](#).



MassMotion Scene

#### Floors

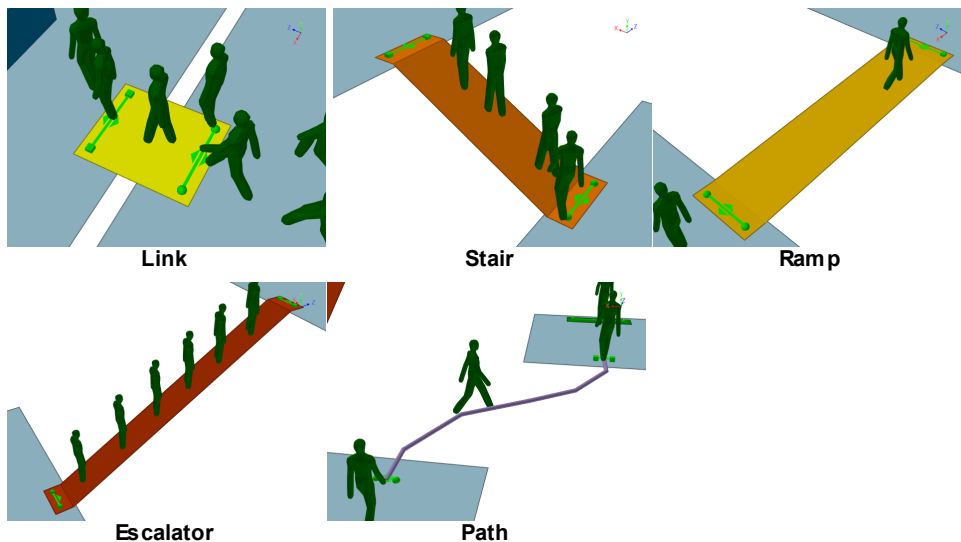
[Floors](#) are the most fundamental scene object. They represent the spaces (rooms, hallways, plazas, sidewalks, train platforms) which define the program areas of a design. Each floor defines a separate and distinct walkable area, with agent movement constrained by the floor boundary.



Floor

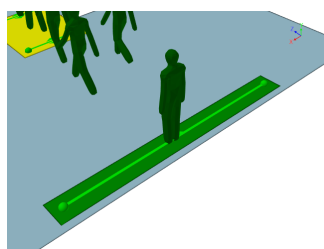
### Connections Between Floors

People may only move between floors where they are connected by [connection objects](#). [Links](#) represent simple flat doorways or turnstiles. [Stairs](#), [ramps](#), and [escalators](#) connect floors at different elevations. [Paths](#) connect floors at any elevation and restrict movement to single file. Connection objects also act as decision points in the route network.



### Entrances and Exits

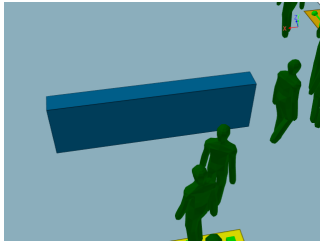
[Portals](#) serve two main functions: they mark areas where people can enter the simulation and they represent destinations to which people can be sent.



Portal

### Obstructions

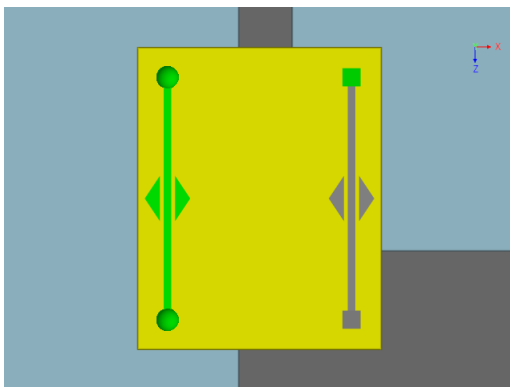
[Barriers](#) represent walls, columns, tables, benches, and anything else that can constrain movement on a floor.



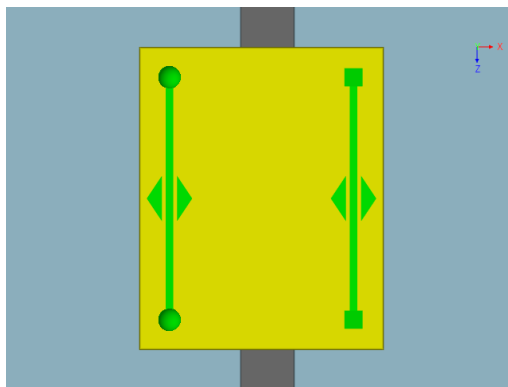
Barrier

### 3.2.1.1 Connecting Objects Together

Each [connection object](#) (ie. [link](#), [stair](#), [ramp](#), [escalator](#) or [path](#)) has a goal line at either end. [Portals](#) have a single goal line. It is the goal line which connects an object to the underlying floor. Arrows on either side of the goal line indicate the possible directions of travel over the connection. Goal lines must be above the floor but not higher than 0.4m, with both ends at least 0.2m from the edge of the floor. The "balls" and "boxes" at the ends of the goal lines are positioned with respect to these offset requirements and may be used as guides when positioning connection object edges.



Box Disconnected



Ball and Box Connected

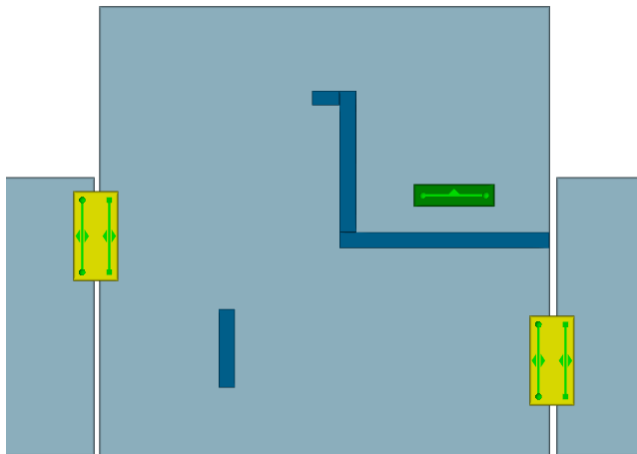


Portal Connected

### 3.2.1.2 Determining Walkable Space

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 not on the walkable object or blocked by an obstacle. Black, white, or grey areas represent useable space. Surface maps are created automatically for each object in the scene at the time of simulation. Understanding surface maps can help with understanding how people in MassMotion navigate a space.

Surface maps can be exported as part of the simulation results (see [Generated Simulation Files](#)) or viewed within a debug simulation (see [Debug Simulation Window](#)).



A simple room with one portal and two links

#### Obstacle Maps

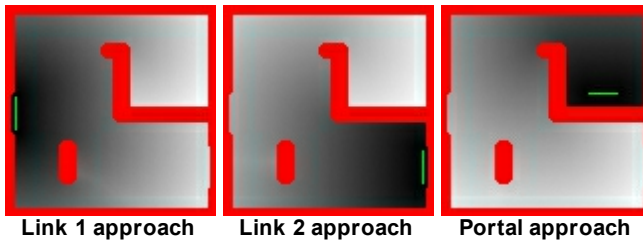
The obstacle map of the simple room marks the location of floor edges and [obstacles](#). Obstacles are included if they intersect any part of the floor, or are within 0.4m of the top of the floor. Any portion of an obstacle that is below the floor or above the 0.4m cutoff is ignored. Floor edges and 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

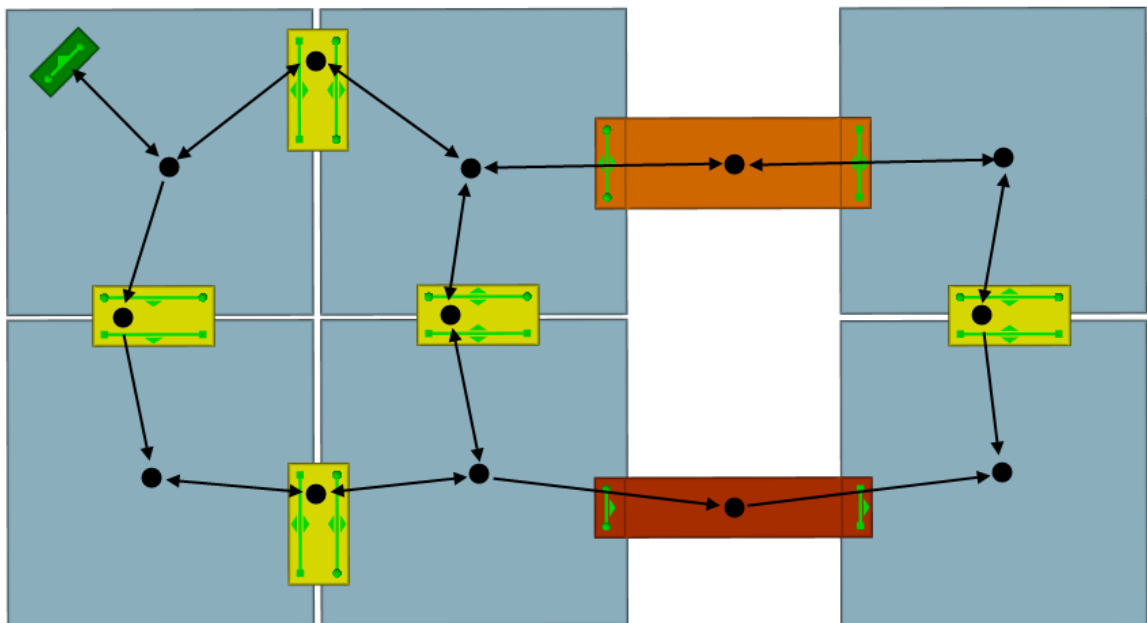
#### Approach Maps

As with the obstacle map, an approach map represents 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.



### 3.2.1.3 The Network

The arrangement of [floors](#), [connection objects](#) (ie. [links](#), [stairs](#), [ramps](#), [escalators](#) and [paths](#)) and [portals](#) is termed the network. The network describes all possible routes available when navigating the scene. In the diagram below, black circles represent nodes in the network. When a person is on a node, the arrows leading from the node represent possible route options.



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

### 3.2.1.4 Destinations

A destination is represented by a [portal](#). People in the scene can be asked to seek out a particular destination and will then use the network to determine the best available route.

Route options are constructed during simulation initialization. The tracing algorithm begins at the destination portal and moves through all available connections, marking the shortest route to every other object in the scene. This information is stored in the form of a cost tree. There is a separate cost tree for each possible portal destination in the scene. The cost tree is then used by people when navigating the network to assist them in finding the shortest route to a particular portal.

## 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. Agents are created and placed in the [scene](#) using [events](#).

### Characteristics

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

### Scheduling

Agents are placed in the scene through [events](#). All events will specify one or more starting locations for the new agents. Some events like the [Journey](#) will specify a single entry time, while others like the [Vehicle](#) will specify a repeating cycle of entry times. Some events like the [Journey](#) will give a single task (seek this portal than exit), while others like [Evacuate](#) will give a series of tasks to be executed in order (wait for 20 seconds, then evacuate the train platform zone, then seek the street portal).

### Behaviour

When placed in the scene, agents are given one or more tasks to accomplish. The agent will execute its tasks in order, relying on two independent intelligence systems: Navigation and Movement.

Scene Interaction	
<b>Task</b>	The <a href="#">task system</a> determines the goals or purpose of an agent. An agent is always attempting to execute its current task.
<b>Navigation</b>	The <a href="#">navigation system</a> 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.
<b>Movement</b>	Once the agent has chosen where to go, its <a href="#">movement system</a> 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

### 3.2.2.1 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.

#### 3.2.2.2 Agent Navigation

Agent navigation makes use of the scene network, assessing route options and planning the best path to a given destination. When choosing between a number of available routes, agents will consider the network objects connected to the current [floor](#), and select the object with the best route cost. The chosen route is called the local target. The choice of target is periodically re-evaluated as the situation on the floor changes. Once the agent has reached the target and transitioned onto a new floor, the routes leading off of that floor are evaluated and the next local target selected.

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.

### 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. 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.



Route Cost Components	
<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.

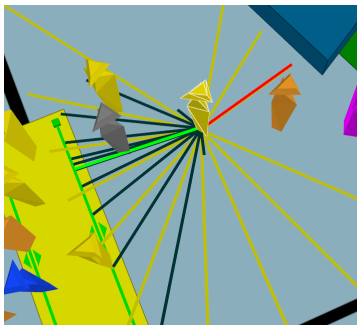
Each route choice is assigned a small random factor. This factor will be different for each agent each time the agent steps onto the floor. As a result, agents will on occasion choose slightly less than optimal routes ensuring that not all agents make the same choice when routes are very close in cost.

[1] Business Case Development Manual, Transport For London, May 2013, Appendix E 3.1

### 3.2.2.3 Agent Movement

#### Finding the Target

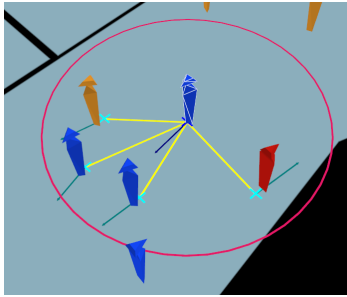
Agents moving towards a local target will determine the direction to that target by looking at the [approach map](#) for the target's goal line. The agent extends feelers out in various directions and measures the distance to the target along each of the feelers. The feeler that ends up closest to the target goal line is taken as the direction to goal.



Agent awareness of direction to target.

#### Neighbours

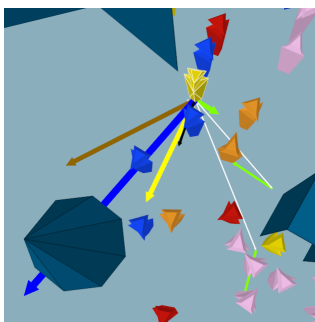
Each agent is aware of other agents that are within a particular range. This range changes with the speed of the agent and the local density. Other agents inside the awareness range are called neighbours. An agent is aware of the location, speed, and size of its neighbours.



**Agent awareness of surrounding neighbours**

**Social Forces**

The social forces algorithm generates a series of forces based on the agent's desired target, the presence of neighbouring agents, and the location of obstacles. These forces are summed together and used to modify the agent's acceleration.



**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 a corner or swing wide.

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.

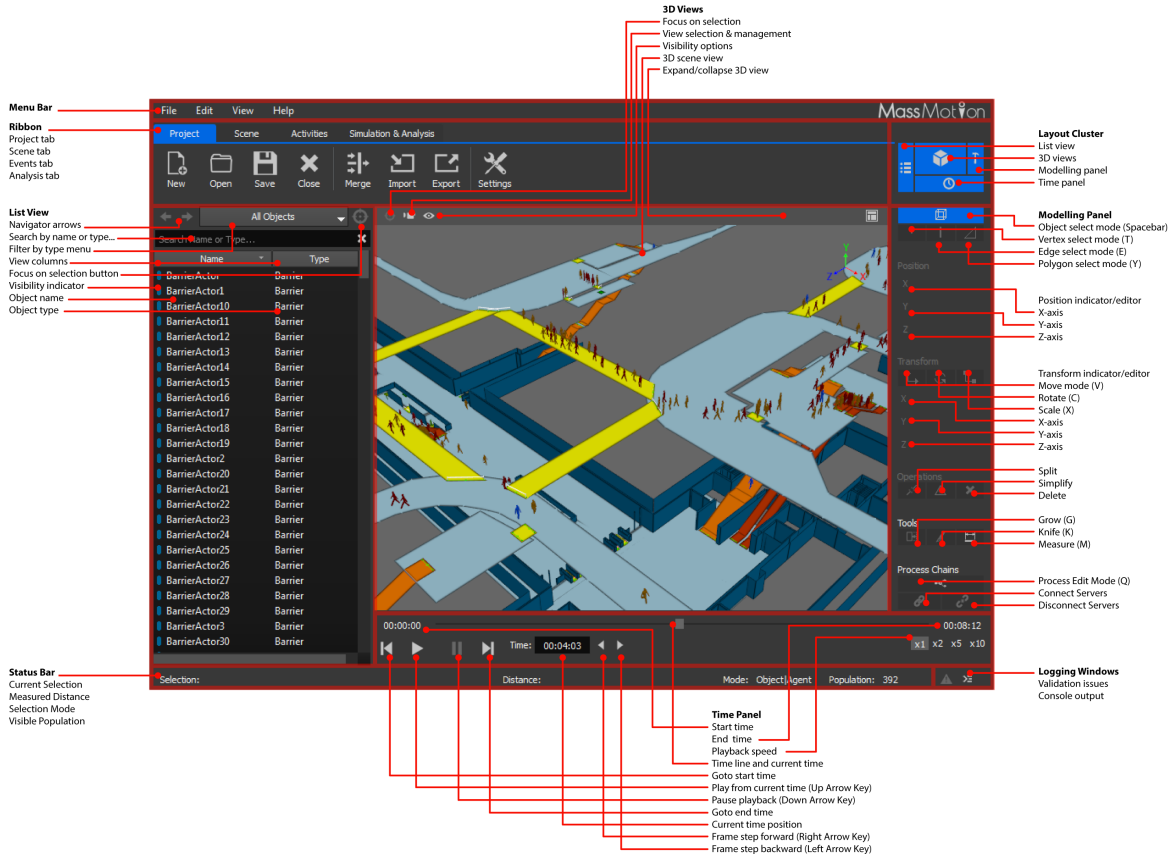
<b>Factors Influencing Agent Speed</b>	
<b>Density</b>	In order to simulate the reduced stride length and reduced mobility of people in crowded spaces, agent speed is reduced as density increases. The exact relationship between speed and density has been tuned to match the data in John Fruin's Pedestrian Planning & Design [1].
<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 cutoff 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

### 3.3 User Interface Overview

#### 3.3.1 Main Window

The following image enumerates some of the more important features in the MassMotion main user interface.



#### Main Menu

The main menu duplicates much of the functionality available elsewhere in the user interface, but consolidated in one place. See [Main Menu Bar](#) for more information.

#### Ribbon

The ribbon is a tabbed toolbar which provides buttons for performing high level operations. The project tab provides buttons for managing the project, the scene and activities tabs provide buttons for creating scene and activity related objects, and the analysis tab provides buttons for validating the project, running a simulation, creating analysis objects, and exporting results to text files, images or videos.

#### Layout Cluster

The buttons here can be used to collapse and expand the list view, scene views, model panel and time panel.

#### List View

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 [List View](#) for more information.

### Scene View

The 3D scene view 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 [Selection](#)).

The camera button at the top provides a variety of preset cameras and the ability to save or recall custom viewpoints. The eye button provides options for controlling how the scene is rendered, from different agent representations, to wireframe mode, to whether or not agents should be hidden when on a hidden floor. The focus button will focus the view on the current selection. See [3D Scene View](#) for more information.

### Model Panel

The panel 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 and measuring the distance between points in the scene. See [Model Panel](#) for more information.

### Time Panel

The time panel 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. See [Time Panel](#) for more information.

### Status Bar

The status bar provides information about the current selection and agent population and has buttons for showing the console and issue windows. See [Main Status Bar](#) for more information,

## 3.3.2 Objects

Much of the data in a project is represented in the form of objects. The MassMotion interface is designed around viewing and managing these objects.

Some objects have a physical presence. These objects are referred to as [scene objects](#) and are visible in the graphical [scene view](#). All objects are displayed in the [list view](#).

### 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 in batch 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>	Objects that have been hidden will not appear in the <a href="#">scene view</a> and will display an outline only for the type capsule in the <a href="#">list view</a> . All scene

	objects can be hidden. 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> .

### 3.3.3 Selection

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

## 3.4 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.

### 3.4.1 Authoring

The process of authoring involves constructing the scene, establishing a network of connected objects, and creating events.

The geometry for a scene can be either [imported](#) from other modeling packages or [created](#) from

scratch. In both cases the geometry of the resulting objects can be [edited](#) to achieve the desired form. The section on [scene objects](#) outlines recommended strategies for representing real world objects and spaces. Finally, [events](#) are used to populate the scene with agents.

### 3.4.1.1 Working with Geometry

#### 3.4.1.1.1 Importing Geometry

MassMotion can import geometry from a variety of formats: **.3ds, .dae, .dxf, .fbx, .ifc, .obj**. Imported geometry is classified as [Reference Geometry](#) and grouped under a single [Reference Model](#) collection. This collection allows the reference geometry to be scaled, rotated, translated, or hidden as a group. Reference geometry itself cannot be edited and will not be included in any simulation or analysis. **Reference geometry can be imported through the 'Scene' tab of the main window.**

Reference geometry can be used through its right-click menu to generate a [floor](#), [link](#), [stair](#), [ramp](#), [escalator](#), [barrier](#), or [portal](#). Once used, the reference geometry is automatically marked as used and hidden. There are two ways to convert reference geometry into a MassMotion object: 'Duplicate as' and 'Use to Generate'.

Creating MassMotion Objects		
<b>Duplicate as</b>	<b>Right-click on a selection of faces</b>	Creates an object of the specified type, using duplicates of the selected faces as its geometry.
<b>Use to Generate</b>	<b>Right-click on reference geometry or a selection of faces</b>	Creates an object of the specified type. Examines the reference geometry or selected faces and based on the object type, uses a subset of the faces to generate new representative geometry. When generating a floor, this might involve taking only the top faces from the reference geometry.

**IFC** based reference geometry contains additional contextual information about the function of the object. When generating objects from IFC based reference geometry, the 'Auto' object type indicates that MassMotion should choose a target object type based on the original IFC type. For example, `lfcSlabs` and `lfcSpaces` might become floors, while an `lfcDoor` would become a link.

**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.

**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.

#### 3.4.1.1.2 Creating Geometry

MassMotion objects can be created through the ribbon buttons at the top of the main window, or through the right click menu in the 3d scene view. New standard objects come with a default geometry which can then be [edited](#) to match requirements.

New geometry can be created from existing geometry. Select the desired face components, right-

click on the selected faces, and choose the desired creation option:

- Duplicate as: Copy the selected faces and combine them into a single new object.
- Extract as: Remove the selected faces from the original objects and combine them into a new object.
- Use to generate: Copy the selected faces and use them to intelligently construct an object of the specified type.

#### 3.4.1.1.3 Editing Geometry

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

##### **Selecting Objects or 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, then change the selection mode in the top right of the main window, then click on the desired components with the left mouse button. Possible components include vertex, edge, or face/polygon.

##### **Setting Positions**

Objects or components can be set to a specific location using the 'Position' entries in the Main Window's [Model Panel](#). This is useful when a number of vertices need to be along the same line. Select one vertex, note the X, Y, or Z value. Select multiple vertices, enter the required X, Y, or Z value. If setting the Y value, all of the vertices will then be at exactly the same height.

In the case of objects, faces, or edges, the position displayed is of the combined object or component centroid.

##### **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 [Model 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 axis (a yellow square indicates the plane of translation).

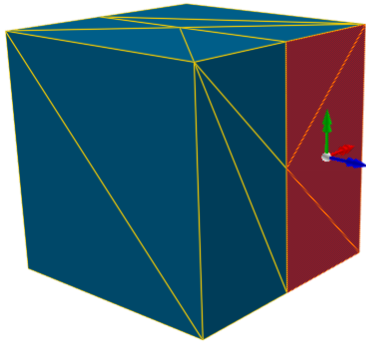
When scaling, the axis arrows can be used to scale only in a particular direction. Clicking between two axes will scale uniformly within that plane (both directions will scale equally). Clicking away from the manipulator (such that the entire manipulator is yellow) will scale equally in all directions.

##### **Extruding (Growing)**

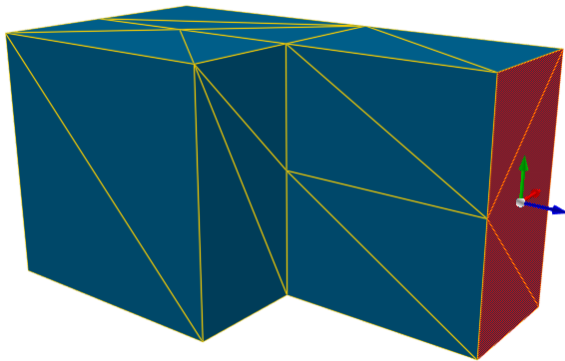
The grow manipulator can be used on mesh objects to extrude one or more edges or faces. In the case of line based objects like paths, end vertices can be grown to extend the line.

Select a vertex (line object), edge (mesh object), or face (mesh object) and use the 'Grow' button in the [Model 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. So for instance, selecting an edge, then growing the edge away from the rest of the object will produce two new faces connected the previously selected and newly grown edge.

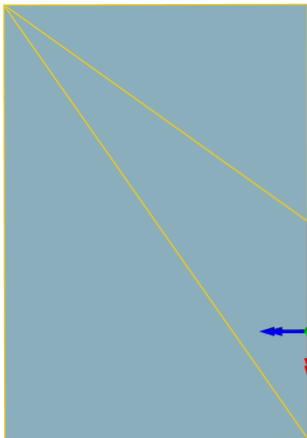




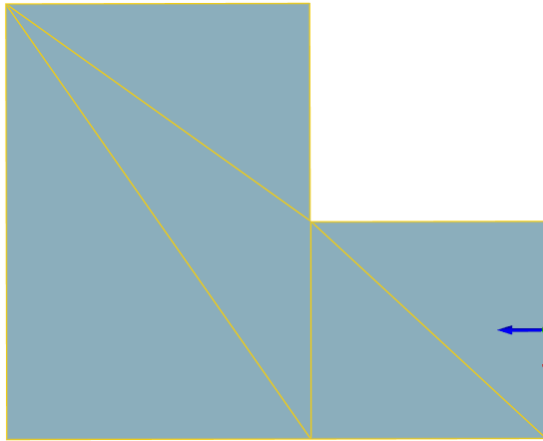
**Before grow faces**



**After grow operation on the selected faces**



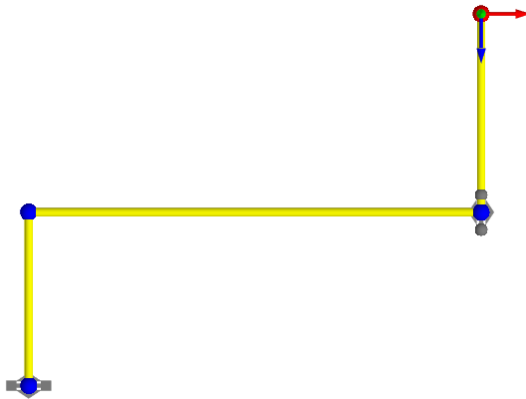
**Before grow edge**



After grow operation on the selected edge



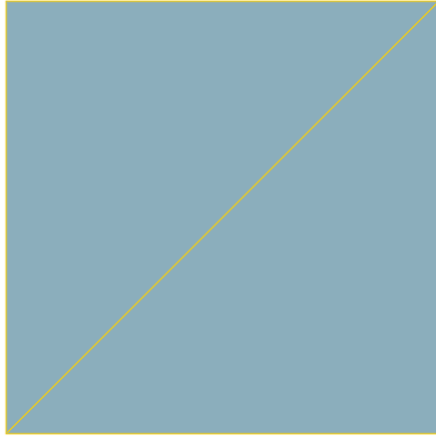
Before grow vertex



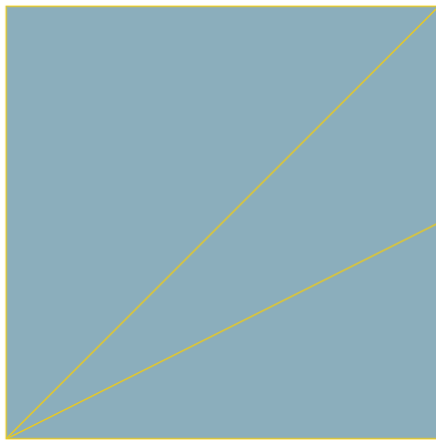
After grow operation on the selected vertex

### Splitting

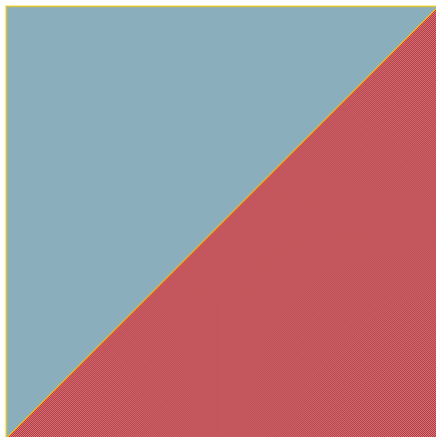
Edges or faces can be split to provide additional edges or vertices for manipulation. Select one or more faces or edges and use the 'Split' button in the [Model Panel](#) or component right-click menu.



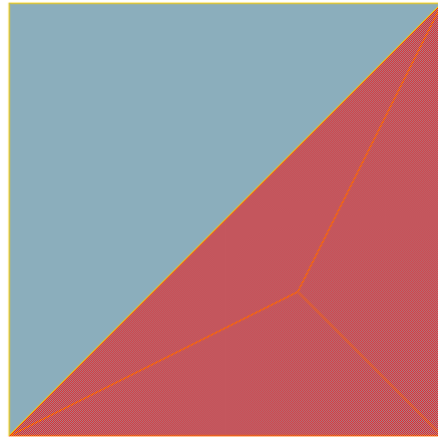
**Before split edge**



**After split operation on the selected edge**



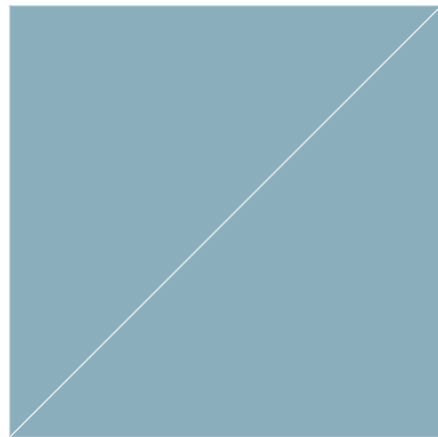
**Before split face**



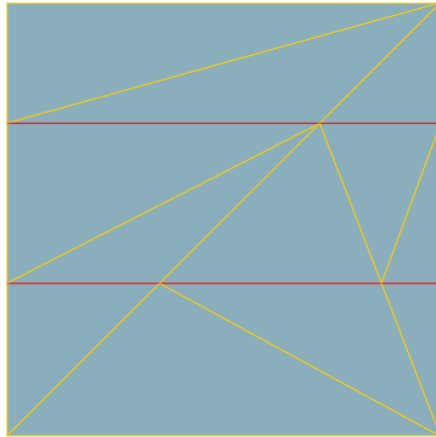
After split operation on the selected face

### Slicing (Knifing)

The knife tool is used to create new edges in an object along a user defined line. Select an object in the scene and then enable the knife manipulator in the 'Tools' section of the [Model Panel](#). Drag a line across the object where the slice is to be made. The new edges can be useful for isolating a section of an object. The new section can then be deleted to create a hole or grown to produce an extrusions. Note that only objects that are actually contacted by the line will be cut; in shaded display mode, the cut line must touch at least one triangle of a mesh and in wireframe display mode the cut line must touch at least one edge.



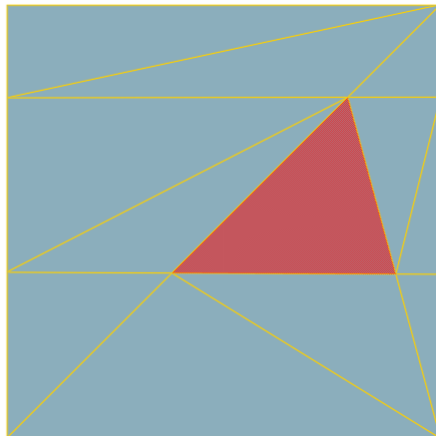
Before knife



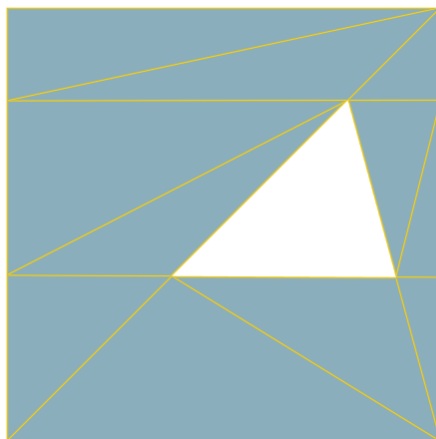
After two horizontal knife operations on the object.

### Deleting

Select an object or component and use the delete button in the [Model Panel](#) to delete the selection.



Before delete face

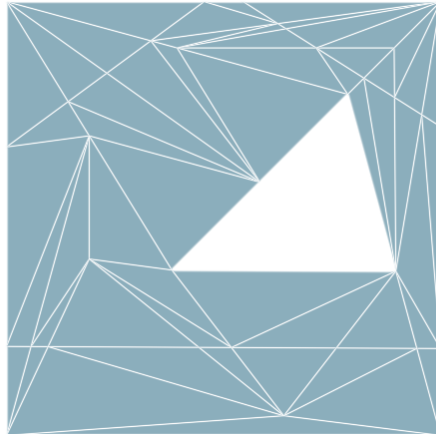


After deletion of selected face

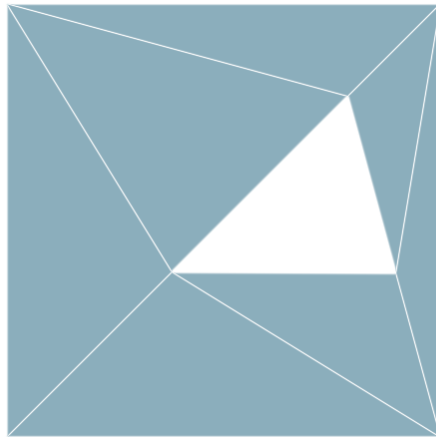
### Simplifying

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 [Model 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**

### 3.4.1.2 Scene Development

The contained sections detail strategies for modeling real world objects in MassMotion.

#### 3.4.1.2.1 Floors

[Floors](#) are the basic building block of MassMotion scenes and getting them right is important to developing effective simulations. They 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 to the boundary of a single floor. [Links](#), [ramps](#), [stairs](#), [escalators](#), and [paths](#) act as [connections](#) between floors enabling agents to transition from one floor to another.

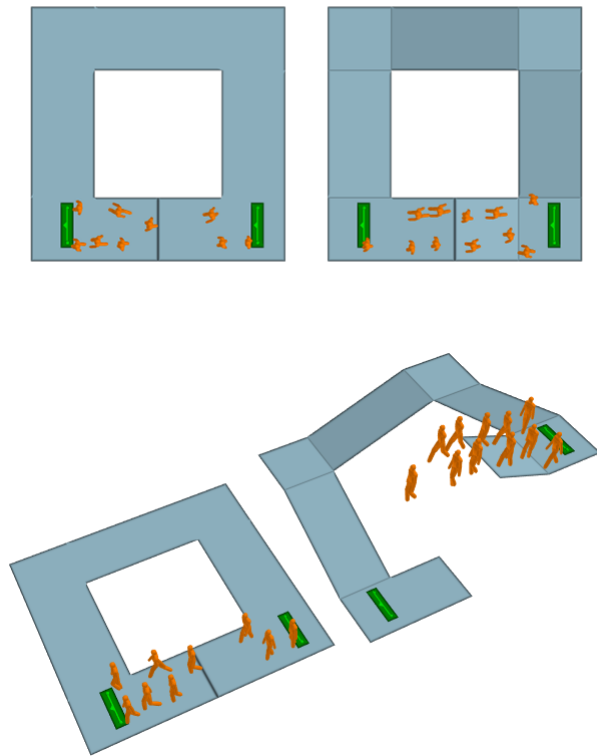
#### Size

The size of floor objects is not explicitly constrained in MassMotion but there are a number of factors that should be accounted for when considering how big to make a single floor. During simulation runtime there are a number of data structures that are automatically constructed which describe the [walkable space](#) of the object. The size of these data structures is directly related to the size of the object's geometrical bounding box so a 2m x 2m floor requires four times as much memory to

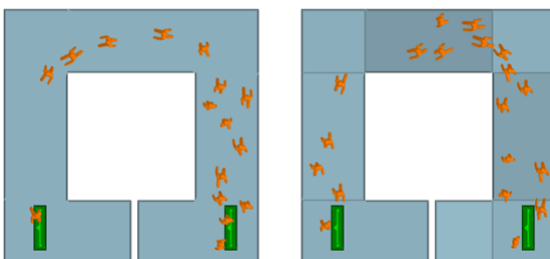
describe than a 1m x 1m floor. If there are very large floors in a scene there may be noticeable increases in simulation setup times and in some extreme cases may cause memory to exceed local RAM capacity which can lead to slow simulations or crashes.

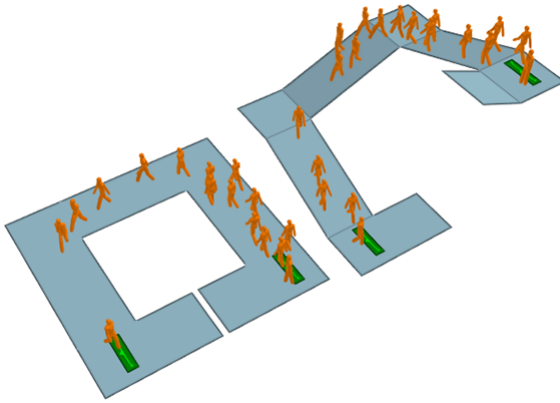
### Shape

The topology of floors (or any other walkable object) is explicitly constrained by the fact that the [walkable space](#) data structures are two dimensional in nature as height is not considered. This means that (from a navigation point of view) the agents on the floors below are free to walk directly between the entry and exit portals. While this is acceptable for the flat floor configuration it results in clearly wrong agent behavior in the sloped configuration.



When the floors above are corrected such that there is a gap between the two portals (as shown below) the agents are then presented with one unambiguous route from the entry to exit portal in both the flat and sloped configurations.





### Slopes on Floors

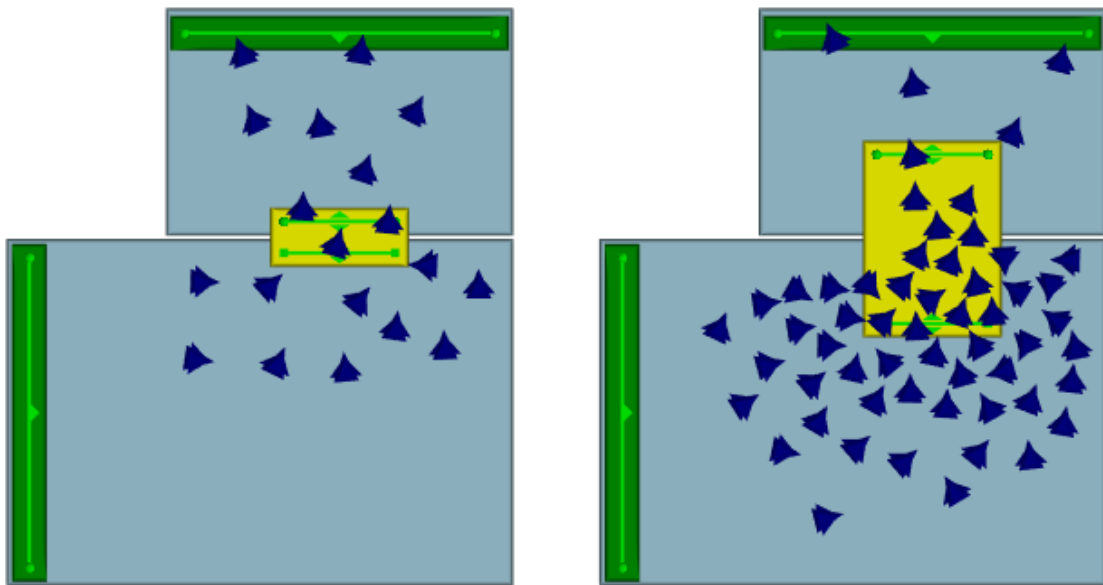
Agents walking unobstructed on a floor will always move at the same horizontal speed regardless of the slope of the geometry under them. In the images above the agents on both the flat and sloped configurations will take the same time to reach the exit. For this reason it is recommended that floors be constructed with geometry that is as close to flat as possible. Agents will adjust their vertical position by tracking top surface of a walkable object, but this has no impact on their horizontal speed.

#### 3.4.1.2.2 Connection Objects

Connection objects are used to provide access between pairs of floors in the form of doorways, stairs, ramps, etc. 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](#).

As described in [Connecting Objects Together](#) there are a number of geometric limitations on link objects. As shown below links must overlap the floors that they connect to but care should be taken to minimize the amount of overlap to the minimum. In the left hand example the overlaps have been minimized enabling a smooth approach flow through the link. In the right hand example the overlaps are far too generous leading to agents approaching the goal line from both the front and back sides of the goal line and interrupting the flow of agents. In both cases the same number of agents are created and the inefficiency of the right hand configuration has a noticeable impact on the results.





All connection objects share a set of common properties:

#### **Direction**

Connection objects may allow agents to enter from either direction or only a single direction. [Escalators](#) only allow agents to travel in one direction.

#### **Gates**

All connection objects can be configured to be gated. Gated connections 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**

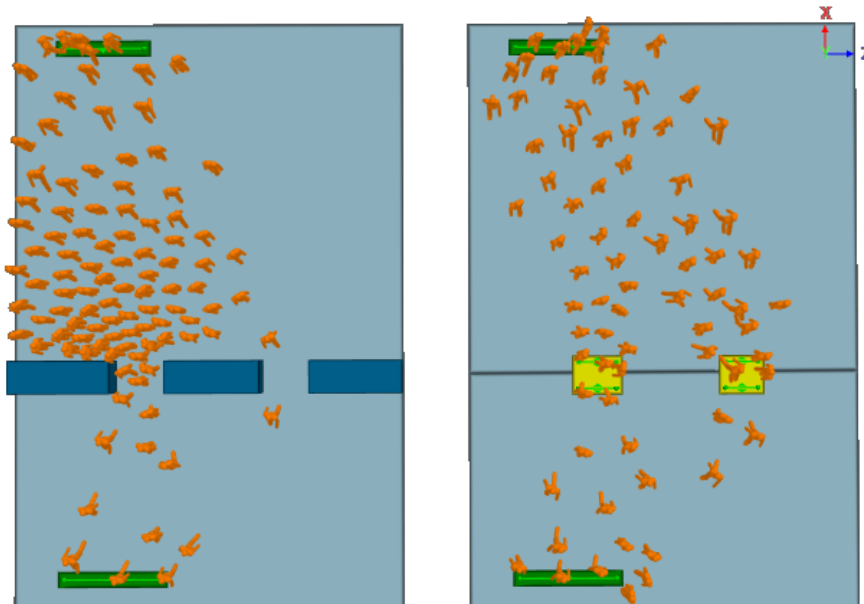
Each connection object can be added to a single [bank](#) and/or a number of [perimeters](#). Banks and perimeters control how the link is used by the agent when navigating the network.

3.4.1.2.3 Complex Spaces

**Determining when to split up floors**

When a person navigates the network, they are constantly choosing a path from a set of possible routes. These routes can be represented as a series of goal lines. It describes all possible routes through the scene with each scene object representing a node or choice point along the route. Agents are continuously evaluating the cost of available nodes when determining the optimal route to a goal. Costs are calculated from factors such as horizontal distance, vertical distance, local queuing, and oncoming traffic.

In the image below, the same layout is modeled in two different ways. On the left, two narrow channels are created by placing barriers on a single floor. There is no network and so no navigation required. The navigation system will choose the portal as destination and rely on the movement system to carry it across the floor. The agent movement system is aware of the resulting congestion, keeping agents from colliding with one another, but the navigation system is unaware so all agents will take the shortest path to the portal. On the right, the single floor has been split in two with links representing the narrow channels. The navigation system is aware of congestion as it builds up around each link, and will factor that in when choosing between the two routes.



**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.**

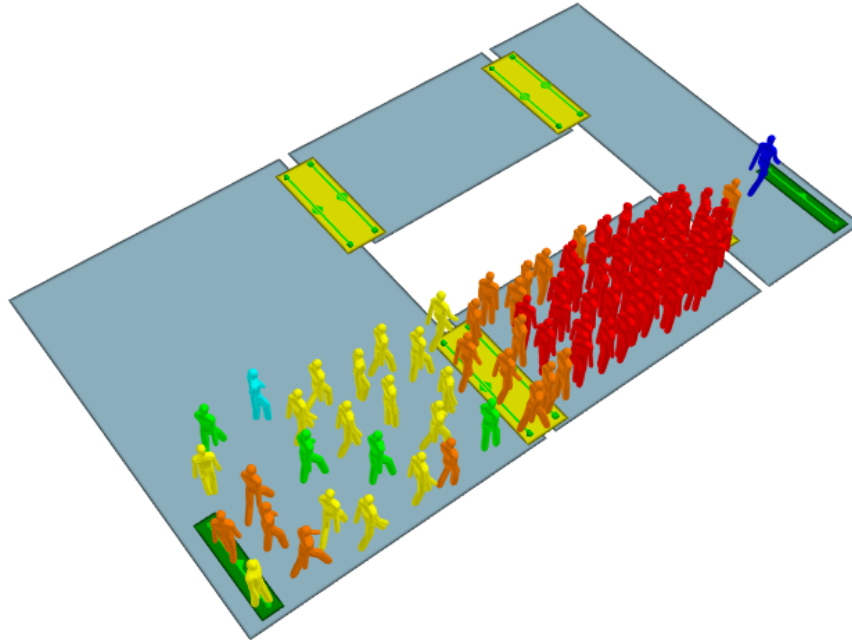
This process of splitting floors to ensure agents fully consider route implications is useful in a number of situations including:

- 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)

**Considerations**

While splitting floors up and introducing additional links to enhance network choice can be beneficial, there are counterbalancing factors to consider. Agents are only able to perceive

congestion associated with links that are connected to their current floor. 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 is by design as in most real world cases people are not able to see downstream congestion but this should be considered when deciding when and where to split up complex spaces.



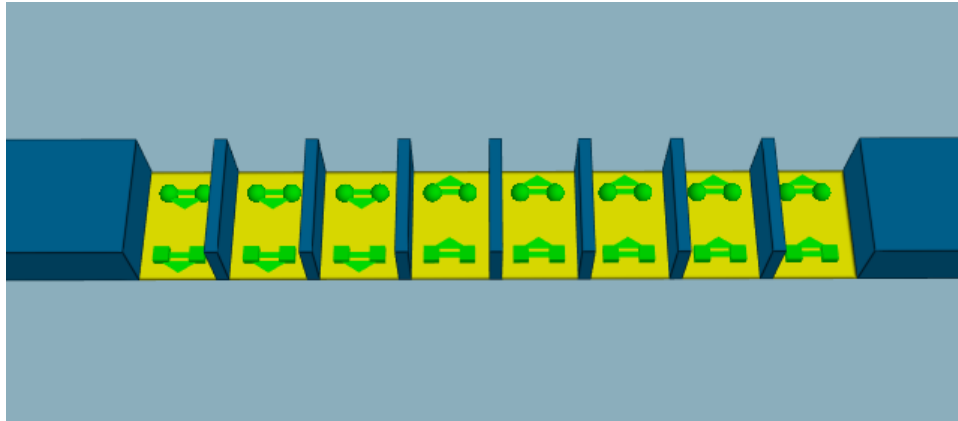
#### 3.4.1.2.4 Turnstiles

MassMotion is used extensively for analyzing transit stations, and the modelling of turnstile geometry and related agent behaviour is a common requirement. There are a number of factors which should be considered when modelling turnstiles.

##### **Geometry**

The geometry of the turnstiles will be similar to [Doors](#). 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

### Object Properties

If the turnstiles are to be used in one direction only, set that direction in the link 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) or if the turnstile is not aligned with the scene axes, it may be necessary to also 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.

### 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, links in the same direction should be banked. Please see [Banks](#) for more information.

#### 3.4.1.2.5 Stairs and Escalators

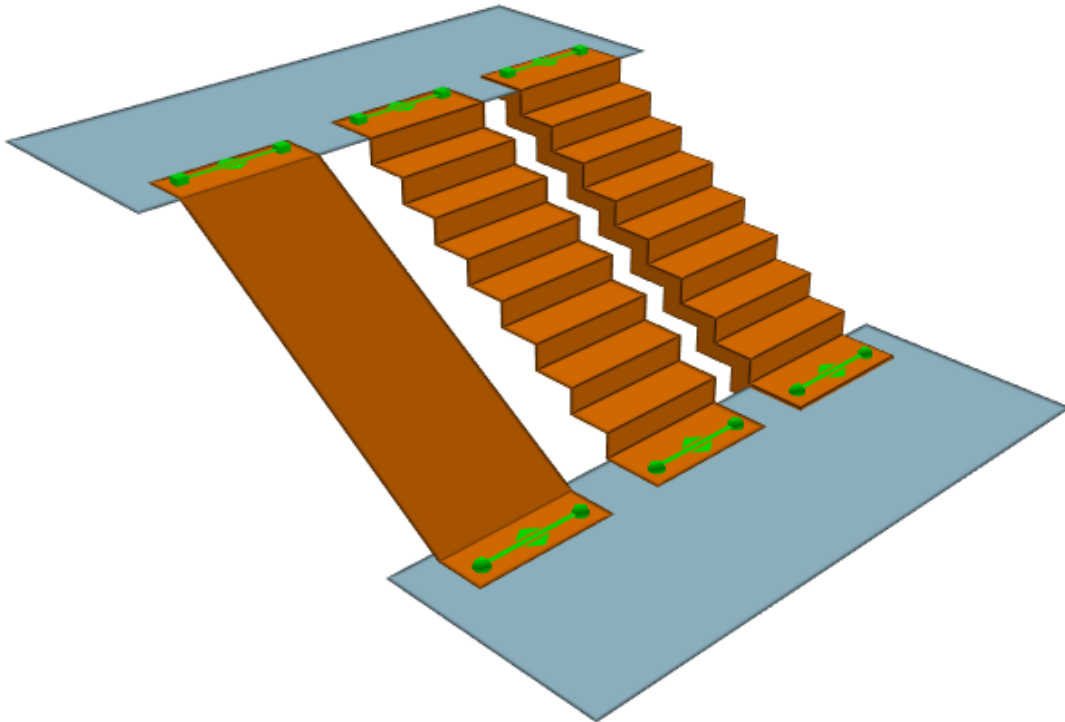
Stairs, ramps, and escalators (vertical circulation elements or VCEs) are link objects that connect floors of different heights using a sloped vertical transition. They have much the same best practice guidelines that are described in the [Doors](#) guide to simple links and in the [Connecting Objects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

- The slope of VCEs must be between 0 and 50 degrees from level. Steeper slopes may result in agents being unable to track the surface during simulation
- On stairs agents will reduce their horizontal speed proportional to the degree of the slope
- On ramps agents traveling upwards will reduce their horizontal speed proportional to the degree of the upward slope. Agents traveling downwards will ignore ramp slope and not reduce their speed
- On escalators, agents will travel at the rate specified in the Escalator Average Speed parameter of the object's property page

- Escalators must be unidirectional

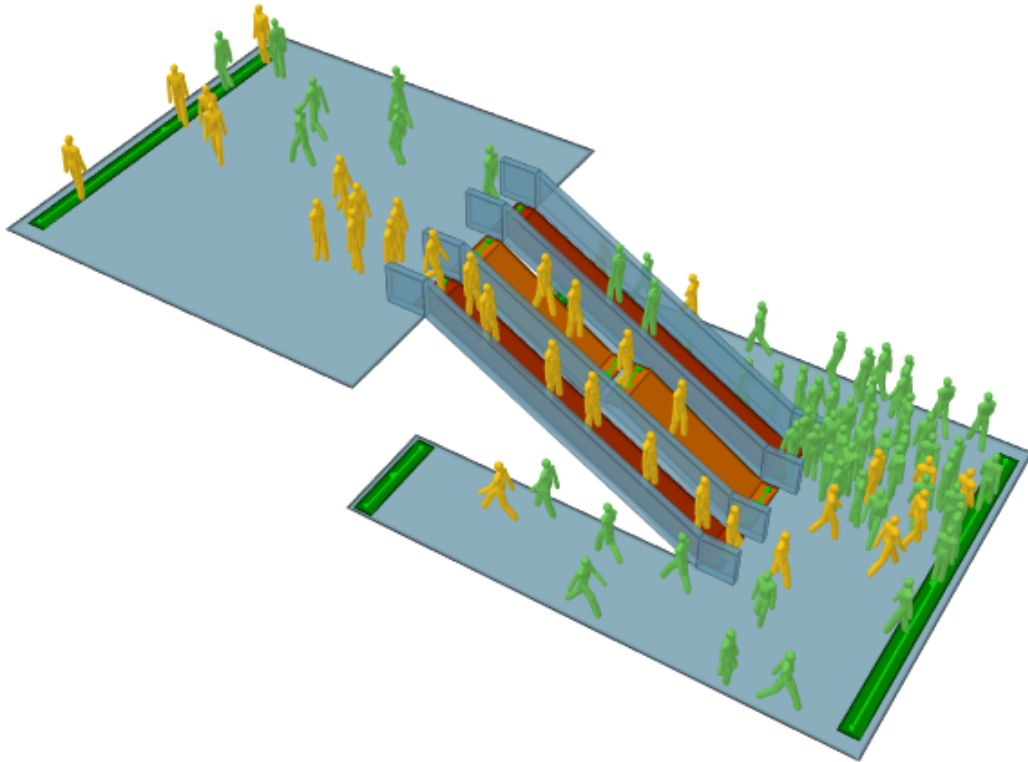
### Geometry

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. The only restriction is that they may not overlap themselves when projected vertically as described in [Floors](#) topic.



### 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. It is recommended to construct the banisters for the escalators as they would appear in the real world (i.e. extending past the escalator landings) to encourage agents to approach the VCE objects in an aligned and organized manner. This leads to more realistic queuing behaviour and flow rates on the VCE objects. 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.



### Escalators

To ensure that the escalators are able to process pedestrians at the expected rate, it is important to follow these guidelines:

- Ensure the geometry is well formed as described above.
- The modeled width of a 1m (double channel) escalator should be the width between the handrails (e.g., 1.2m) as this accounts for the total diameter of a person which is required to achieve "doubling up" of agents on MassMotion escalators
- Escalator speed should be set appropriately for the expected flow rate
- Traversal type should be set to the default (for VCEs) "Ignore Barriers"

### Banks of Vertical Circulation Elements

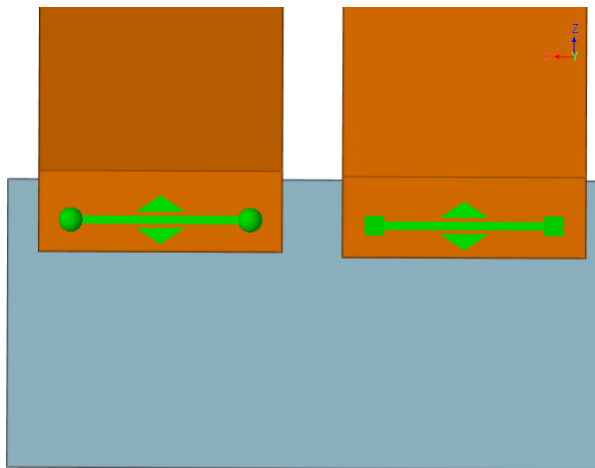
When multiple stairs, escalators, or ramps are positioned beside one another, have the same directionality, and bridge the same two floors, they should be banked to ensure agents make use of all available elements. In Figure 2, the two escalators are in opposite directions, while the stair is bidirectional. 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.

#### 3.4.1.2.6 Switchback Stairs

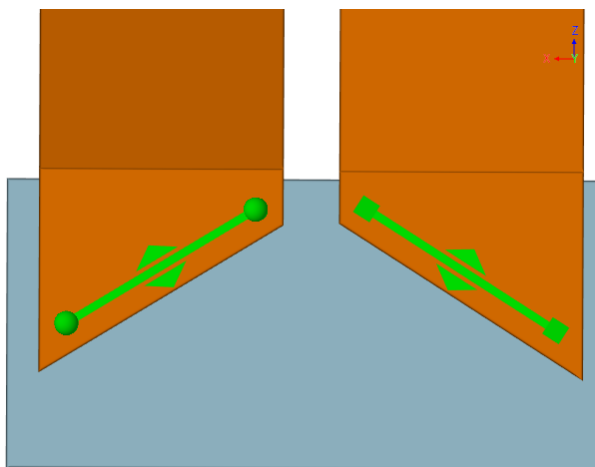
Switchback stairs such as those commonly found in the evacuation cores of buildings present a set of unique challenges. These stairs are generally narrow, with constrained landing areas where agents must make abrupt changes in direction.

### 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.



Initial switchback stair landing



Modified switchback stair landing

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

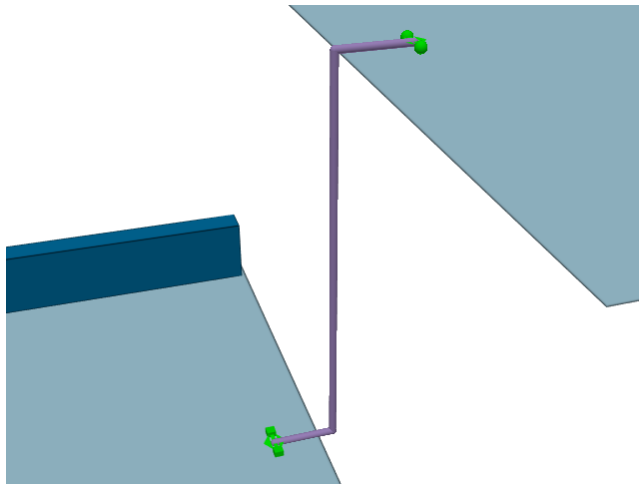
Given a demand at or below level of service D, switchback stairs process agents at a rate roughly equal to similarly sized straight stairs. For level of service E or F, there can be a 10% drop in processing rates despite the geometric adjustments described above. To achieve the same rates as straight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

#### 3.4.1.2.7 Ladders

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

### Geometry

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



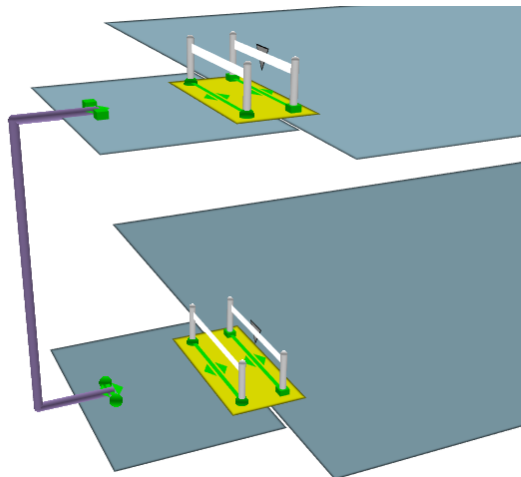
A ladder modelled using a path.

**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.

3.4.1.2.8 Elevators

There is no built in object type for elevators, but elevators can be approximated using a combination of gated links, floors, and paths. Note that elevator approximations will not model true car capacity or proper car timing, but will rely on controlled flow and set door opening times.



An elevator can be modelled using a combination of floors, gated links, a path, and a gate event.

**Floors, Links, and Path**

Create a small floor for each level at which the elevator will stop. Connect these elevator floors to their levels with gated links. Connect each elevator floor to the floors above and below using a path. The paths will allow agents to travel between levels while the gated links will control access to the elevator, opening when the elevator is at a particular floor. Create a separate gate event to control each elevator link.

**Properties**

Ensure the path directionality is set to bidirectional.



The elevator links must be gated. Set priority access to give priority to agents leaving the elevator. Use controlled access to limit the number of people that can enter the link to simulate a fixed car capacity.

For each gate event, add the corresponding gated elevator link, enable cycling, set the on duration to be the time the elevators doors will remain open, and the off duration to be the time it takes the elevator to leave the floor and return.

#### 3.4.1.2.9 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. For the entrance property of a journey, only portals would be used from the specified collection and all other members would be ignored. A single collection could contain all of the floors, door links, and portals associated with a train car. The same collection could then be specified for the gates in a gate event and the portal origins in a journey event.

When a particular type of member is expected from a collection, the collection name is appended with that member type name: MyCollection.Portals, MyCollection.Areas, MyCollection.Gates.

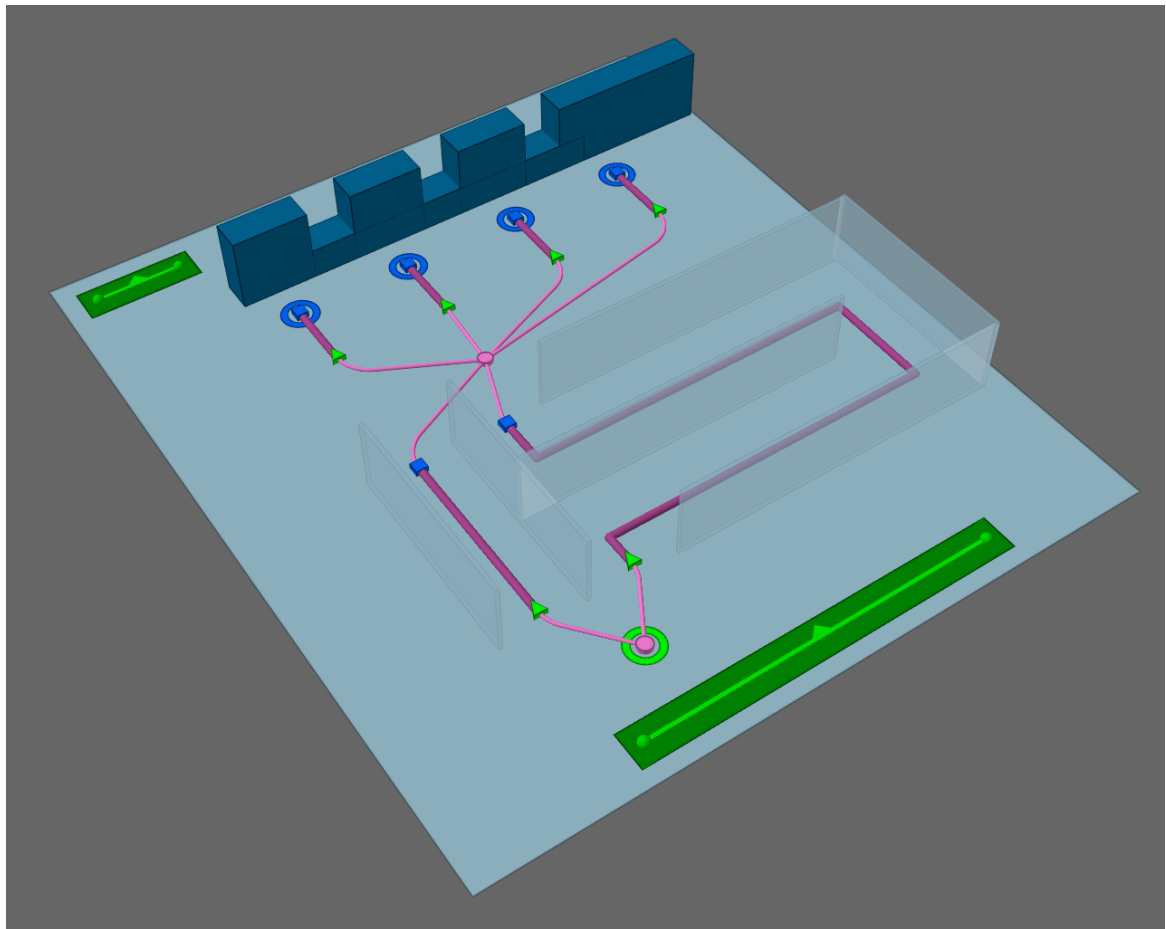
Please see [Collections](#) for more information on referencing collections from other objects.

Collection Type	
<b>Bank</b>	A bank 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> ) 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="#">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>	<p>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.</p>
-------------	---

3.4.1.2.10 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 feed into 4 service positions. The two accumulator queue entrance points should be connected together such that agents can be sent to a single dispatch entry point for the entire process chain.

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 take 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.

### 3.4.1.3 Creating and Controlling Agents

#### 3.4.1.3.1 Scheduling & Events

[Events](#) are objects which modify the scene during a simulation. Events fire at a specified start time but may be cyclical or repeating. Many events insert agents into the scene.

#### Creating Agents Using Scheduling Events

All agent related events will place agents in the scene at specified portal locations. Some events fire only once, distributing agents over a specified interval, while other events will repeat a set of number of times, creating agents over each iteration. 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 burst 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.
<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, each with a set of agents created in the scene and moving to depart on the vehicle, and with a set of agents departing into the scene from the vehicle.

#### General Events

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

General Events	
<a href="#">Execute Action</a>	Applies an <a href="#">action</a> to agents at a specified time.
<a href="#">Open Gate</a>	Opens <a href="#">connection objects</a> which have been configured as gates.
<a href="#">Reference 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.

#### 3.4.1.3.2 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.

##### Definition

A process chain is a series of one or more servers connected together and 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.

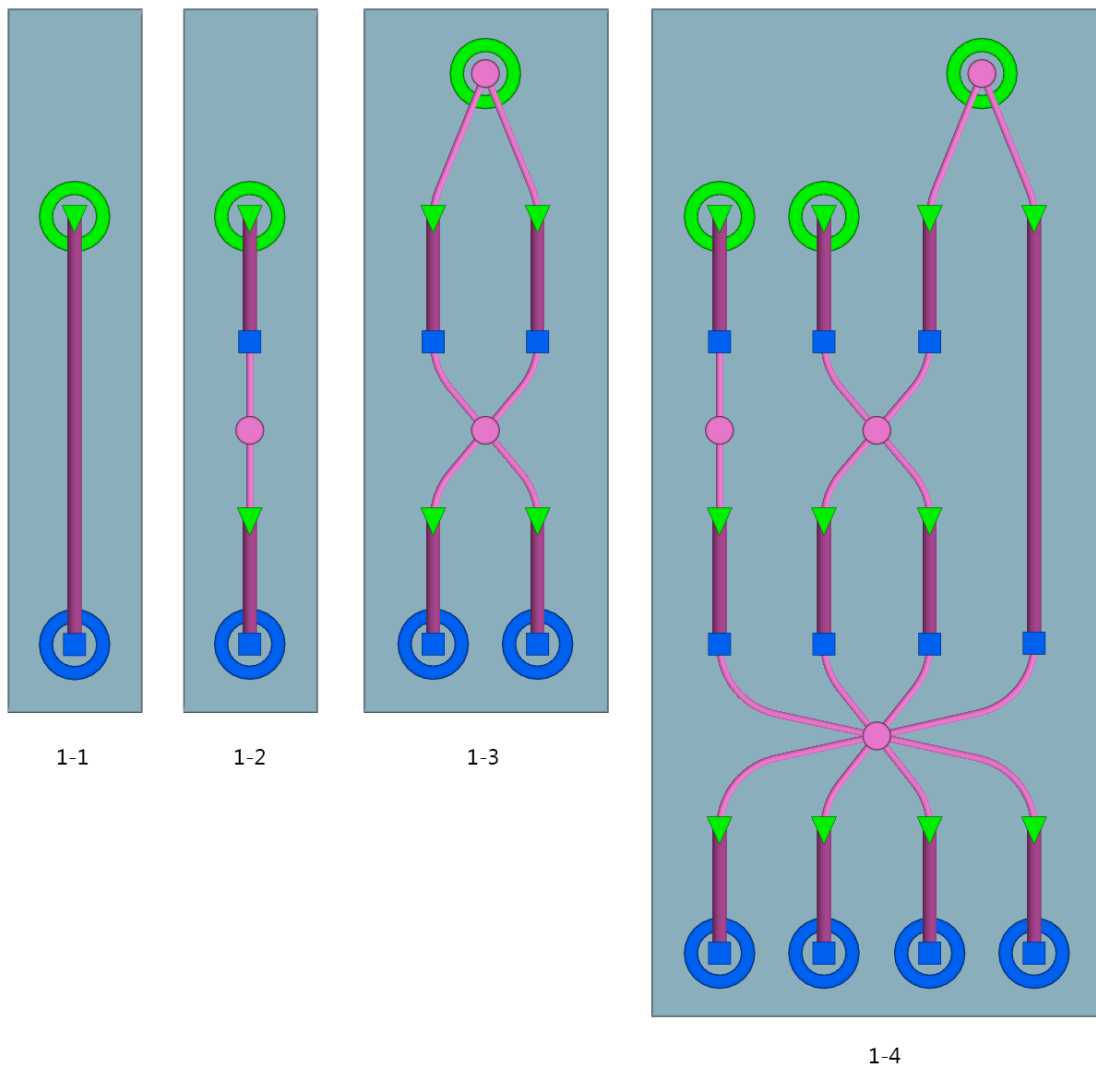


Figure 1

### 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 through the dispatch. Within the context of a dispatch, 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](#).

### Creating Process Chains

Creating process chains is done 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. These yellow circles may be connected together by selecting the circles and clicking 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". For a detailed walk through of creating process chains, please watch [this tutorial video](#).

### 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.

### 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.

#### 3.4.1.3.3 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 time triggered action 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](#).

### 3.4.2 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](#)). Both methods make use of all available processors on the machine. 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.

#### 3.4.2.1 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 use 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 a copy of the project used as input to 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.

### **3.4.3 Analysis**

Each time a simulation is executed the results are recorded in a [simulation run](#). Tools are provided in MassMotion for playing back the simulation or extracting data from the run in the form of a query. Queries can be customized for a particular project, saved, and re-applied to additional runs.

#### **Playback**

The results from a simulation run can be reviewed through playback. It is possible to view agents from multiple simulation runs at the same time. For more information see [Playback](#).

#### **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.

#### **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. For information see [Reporting](#).

#### **3.4.3.1 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. These populations are presented over top of one another. This can be very useful when comparing populations from runs with slightly different setups.

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.

### 3.4.3.2 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.

## 3.5 Troubleshooting

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

### 3.5.1 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.

### 3.5.2 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](#).

A project is automatically validated when launching a new simulation. The project can 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.

### 3.5.3 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.

The basic observer window is available through the right-click menu on a playback agent. If trying to track down deleted agents, review the deletion error message from the simulation window and note the agent ID. Open the agent observer window by right clicking on the simulation run, and enter the agent ID directly.

### 3.5.4 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, or 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.

### 3.5.5 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 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.

### 3.5.6 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.

# 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 the [Objects](#) subsection, 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 Project Settings

The project settings includes 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 a set 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.

Custom Views tab	
Main window	This window shows the list of all custom views. Custom views are created from the <a href="#">3D scene view</a> .
Show	Shows the selected custom view in the main viewport.
Rename	Renames the selected custom view.
Delete	Deletes the selected custom view.

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.

### 4.1.2 Merging Projects

MassMotion includes functionality for merging data from another project file into the currently open project, through the File -> Merge menu entry. 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.

#### 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 there is a conflict with that object, and the user must resolve the conflict manually.

#### 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.

### 4.1.3 Importing and Exporting Objects

Objects can be imported and exported using the File menu or buttons in the 'Project' tab of the main window's ribbon, and can also be exported while [editing object properties](#). Exported objects include both geometry and properties. References between objects are maintained provided all referenced objects are either included in the export, or are already present in the destination project (for instance, when exporting/importing between different versions of the same project).

When an imported .mmxml 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.

## 4.1.4 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> .  The file can be queried as an sqlite database for information about agents or used to recover a project by opening it as if it were a mm project file. See <a href="#">Simulation Data</a> for more information.
.mmxml	Exported MassMotion objects. See <a href="#">Importing and Exporting Objects</a> for more information.
.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.

## 4.1.5 Upgrading Older Projects

Projects saved using an older version of MassMotion will automatically be upgraded to version 8 when opened. No user intervention is required. Projects saved in version 8 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 8.0.

### Converting Workbench Object(s)

Some objects created in the MassMotion 7 Softimage workbench cannot be authored in MassMotion 8. These objects will import into a version 8 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 8 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 Object	MassMotion Object	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	Execute Action	<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.

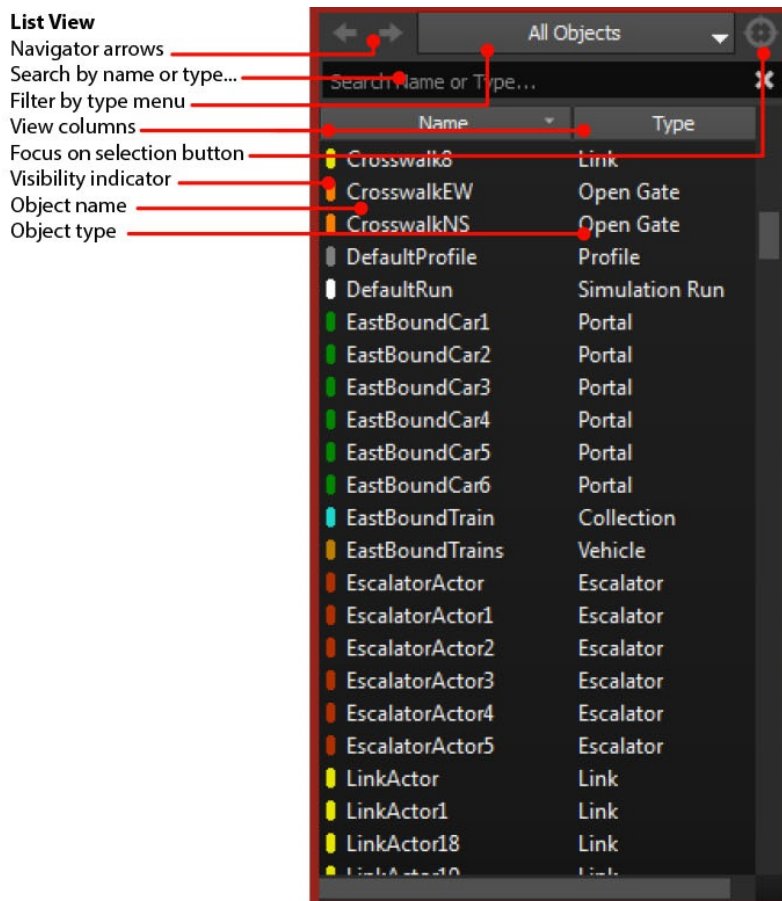


## 4.2 User Interface

### 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.

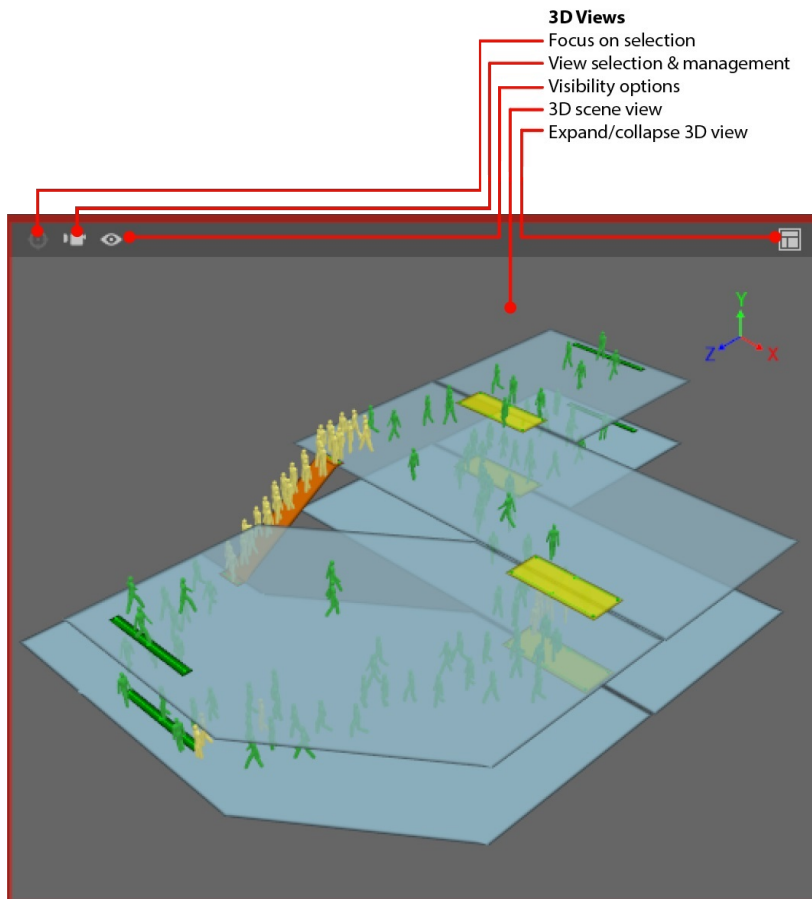
UI Element	Description
<b>Navigator Arrows</b>	Go backwards and forwards through the history of displayed objects.
<b>Search by name or type...</b>	Search objects by name or type. Objects will only be shown if either the object's name or its type contains the given text.
<b>Filter by type menu</b>	Filters 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](#) as well as analysis [cordons](#) and [regions](#). 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 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).

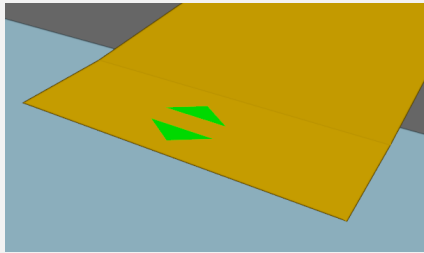
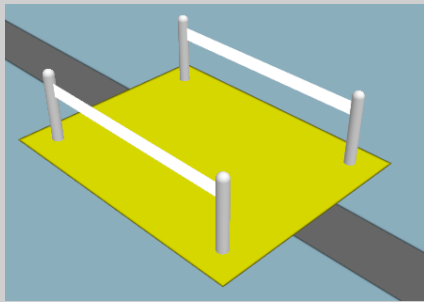
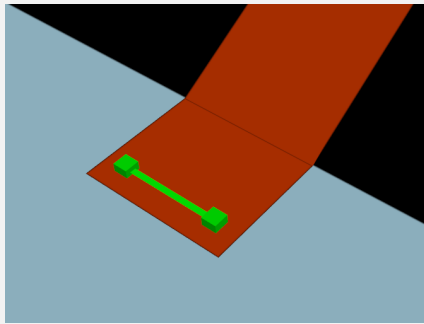
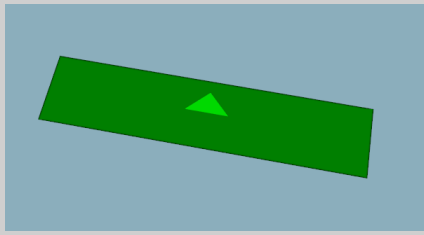
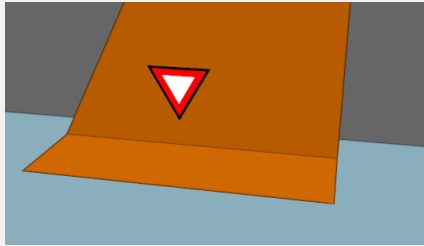
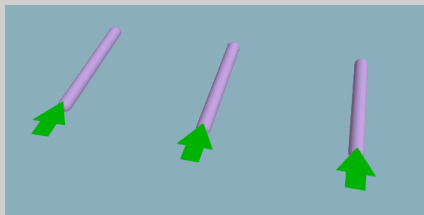
<b>View selection &amp; management</b>	Change the camera between perspective and various orthographic views. Custom views can be saved and applied in this menu as well.
<b>Visibility options</b>	<p>Change the appearance of objects in the scene view. These settings are very useful when setting up the scene to <a href="#">export a movie or image</a>.</p> <p><b>Agents:</b> Change the appearance of agents. See below for a description of agent appearance options.</p> <p><b>Decoration:</b> Toggle the appearance of decorations in the scene view. See below for a description of all available decorations.</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>Render Type:</b> Switch between shaded and wireframe representations.</p> <p><b>Draw Geometry Edges:</b> Toggle the highlighting of geometry edges.</p> <p><b>Hide Agents On Hidden Floors:</b> Agents on hidden <a href="#">floors</a> and other geometry will not be shown.</p>
<b>3D scene view</b>	A graphical representation of all visible scene objects.
<b>Expand/collapse view</b>	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.

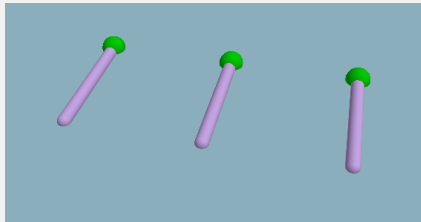
<b>Agent Appearance</b>	
<b>Animated</b>	The default animated avatar with articulated arms and legs.
<b>Avatars</b>	For a project containing advanced features, agents can be assigned <a href="#">custom geometry</a> . This option displays agents using that geometry. If the project does not contain custom agent geometry, the low resolution pegs are used.
<b>Debug</b>	A low resolution simple peg. The size of the peg will change with the radius of the agent.

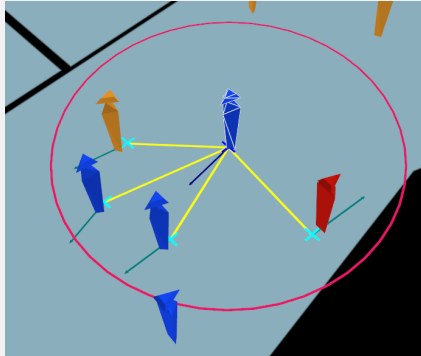
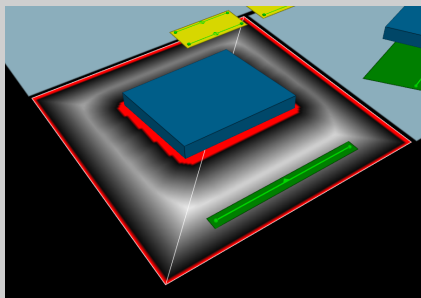
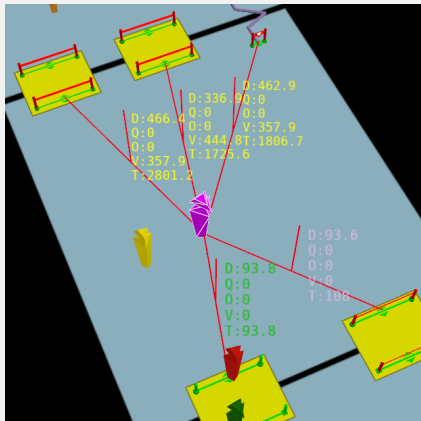
### Decorations

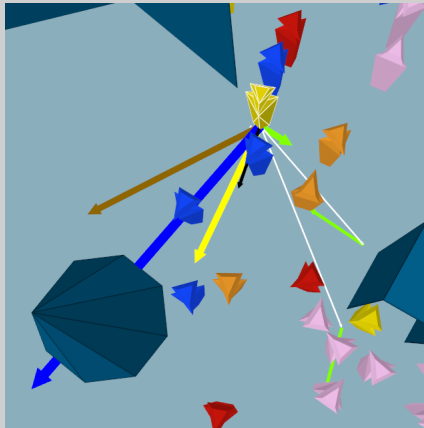
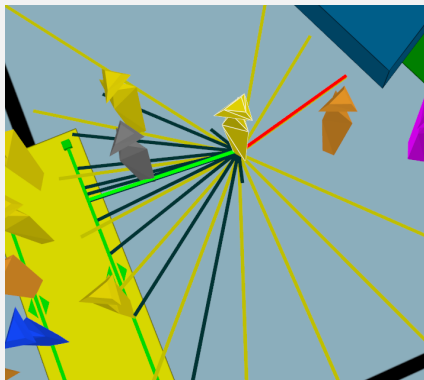
Decorations can be turned on or off for specific objects by right clicking on an object and using the 'Display' menu. All decorations of a given type can be enabled or disabled globally through the visibility options of the scene view.

<b>General Decorations</b>	
----------------------------	--

<p><b>Direction Arrows</b></p>		<p>Displays whether a connection object is unidirectional or bidirectional, allowing agents to cross in each direction. 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>.</p>

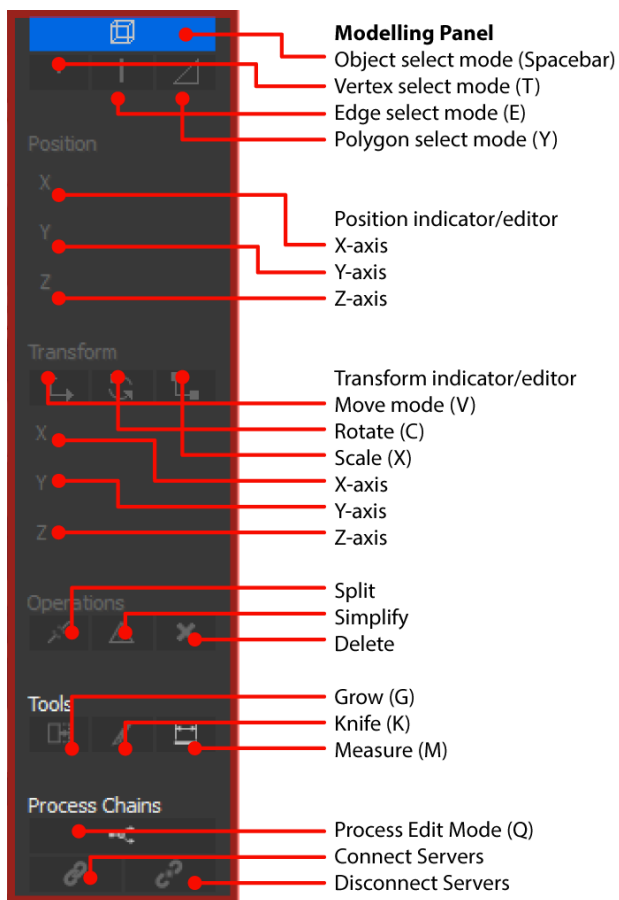
<p><b>Server End Point</b></p>		<p>Displays the end point of a <a href="#">server</a>.</p>
--------------------------------	---	--

<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>Obstacle Map</b></p>		<p>Displays the obstacle map for surfaces 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>

<p><b>Social Forces</b></p>		<p>Displays the forces which affect an agent's movement in debug simulation. See the <a href="#">Simulation Scene View</a> for more information.</p>
<p><b>Surface Probe</b></p>		<p>Displays an agent's awareness of its immediate surrounding in debug simulation. See the <a href="#">Simulation Scene View</a> for more information.</p>

### 4.2.1.3 Model Panel

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

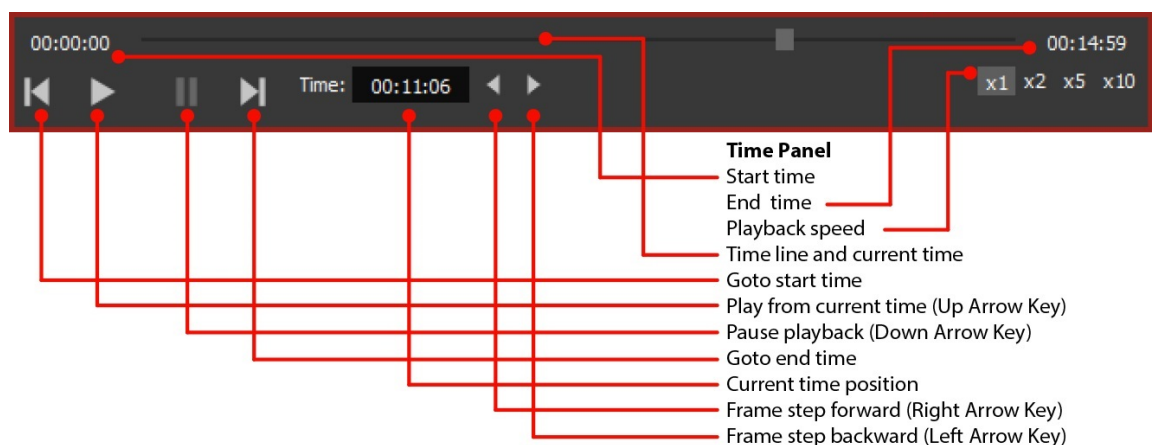


UI Element	Description
<b>Selection mode</b>	The selection mode can switch between Object, Vertex, Edge and Face/ Polygon. See <a href="#">Editing Geometry</a> for more information.
<b>Position indicator/ editor</b>	The position fields display the location of the current selection. In the case of vertices, the values represent the location of the selected vertex. In the case of objects, faces, or edges, the values display the location of the centroid. In all cases, modifying the values will move the current selection(s) to the new location. If more than one object or component is selected and they have different values for X, Y, or Z, a * will appear instead of a value.
<b>Transform indicator/ editor</b>	Translation, rotation and scaling to geometry and <a href="#">scene objects</a> can be activated here. In addition to using a manipulator in the <a href="#">scene view</a> , the precise values of a transformation in each axis can be directly edited. See <a href="#">Editing Geometry</a> for more information.
<b>Operations: Split</b>	Subdivide a selected edge or face. See <a href="#">Editing Geometry</a> for more information.

<b>Operations: Simplify</b>	Merge edges and faces to reduce geometric complexity without changing the object shape. See <a href="#">Editing Geometry</a> for more information.
<b>Operations: Delete</b>	Delete selected objects or components.
<b>Tools: Grow</b>	Extrude a selected edge or face. See <a href="#">Editing Geometry</a> for more information.
<b>Tools: Knife</b>	Cut selected objects along a cutting plane. See <a href="#">Editing Geometry</a> for more information.
<b>Tools: Measure</b>	The measure tool is used to find the distances between points in the scene. Clicking any two points will display the distance between them as well as the X, Y,Z values. By holding shift, multiple points can be selected. The total distance between selected points is displayed in the <a href="#">main status bar</a> . Switching to vertex, edge and face selection modes will snap points to the available components.
<b>Process Chains: Edit</b>	Enable or disable process chain editing mode. See <a href="#">Process Chains</a> for details.
<b>Process Chains: Connect</b>	Connect servers and/or dispatches within a process chain. See <a href="#">Process Chains</a> for details.
<b>Process Chains: Disconnect</b>	Disconnect servers and/or dispatches within a process chain. See <a href="#">Process Chains</a> for details.

#### 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>	How quickly playback runs. 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 current time (Up arrow)</b>	Plays back simulations from the current time. MassMotion simulations are calculated and recorded in 0.2second 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 menubar at the top of the MassMotion window.

Menu Bar		File Edit View Display Simulation Analysis Help	Oasys MassMotion Flow
<b>File</b>			
<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>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> <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>	Export: <ul style="list-style-type: none"> <li>• Objects for import into other MassMotion projects (.mmxml) (see <a href="#">Importing and Exporting 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>Settings</b>	Edit the <a href="#">Project Settings</a> .
<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>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="#">Model Panel</a> for details.
<b>Other Windows</b>	Toggle the display of additional windows:  <b>Console:</b> Show a console window with MassMotion diagnostic output. <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 <b>Observe Agents:</b> Display information about a particular agent from a particular run (see <a href="#">Agent Observer</a> ).
<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)

<b>Help</b>	
<b>Show Help...</b>	Display the MassMotion user guide.
<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.  In all cases the number is in metres.
<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 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 (useful for then finding them in the scene or choosing them elsewhere in the user interface).
<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 <a href="#">table</a> and <a href="#">graph</a> queries which generate data, the arrow will show or hide the results.

#### 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.

#### 4.2.2.1 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 all 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.

### 4.2.3 Choosing Objects

Working with MassMotion frequently involves the choosing of one or more objects such as the [simulation run](#) used by a given query or the set of origin [portals](#) for a [circulate event](#). In both cases MassMotion provides a variety of ways to choose the object(s).

#### 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

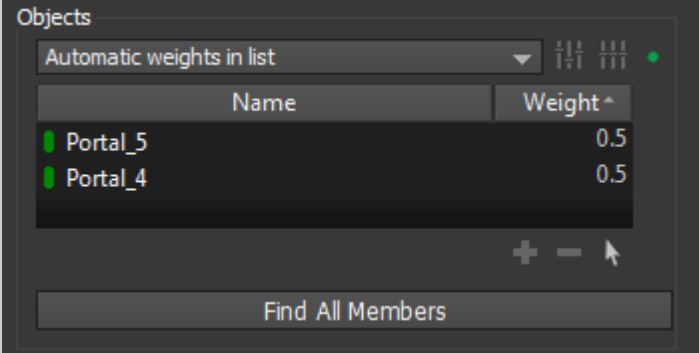
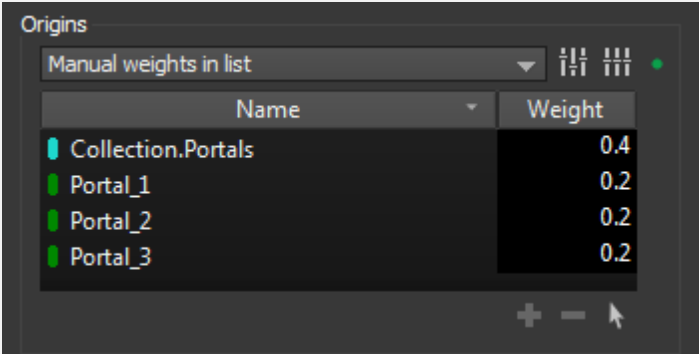
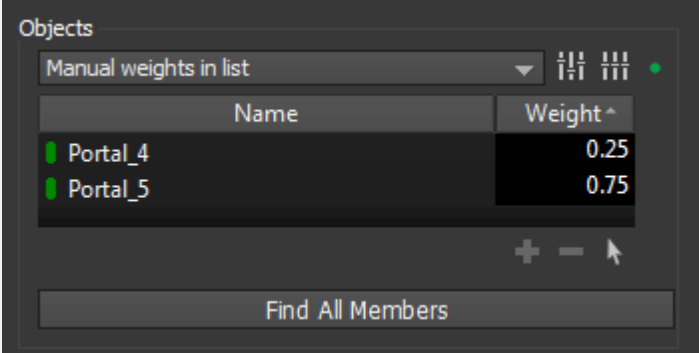
Similar to single objects, clicking on the mouse cursor icon or in an empty multi-object field will bring up a separate selection dialog. The two arrows in this dialog can be used to move valid objects between the **Available** and **Selected** columns. Alternatively, objects that are currently selected 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.

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.

Finally, 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:

Description	Meaning
<b>Ignore weights in list (and collections)</b>	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.

Description	Meaning										
	<div data-bbox="544 300 1246 656" 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><span style="color: cyan;">■</span> Collection.Portals</li> <li><span style="color: green;">■</span> Portal_1</li> <li><span style="color: green;">■</span> Portal_2</li> <li><span style="color: green;">■</span> Portal_3</li> </ul> <p style="text-align: right;">+ - ↶</p> </div> <p style="text-align: center;"><b>Journey origins ignoring weights</b></p> <div data-bbox="544 692 1246 1048" style="border: 1px solid black; padding: 5px;"> <p>Objects</p> <p>Ignore weights in list (and collections) <span style="float: right;">       </span></p> <ul style="list-style-type: none"> <li><span style="color: green;">■</span> Portal_4</li> <li><span style="color: green;">■</span> 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>										
<p><b>Automatic weights in list</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> <div data-bbox="544 1330 1246 1686" style="border: 1px solid black; padding: 5px;"> <p>Origins</p> <p>Automatic weights in list <span style="float: right;">       </span></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Name</th> <th style="text-align: right;">Weight</th> </tr> </thead> <tbody> <tr> <td><span style="color: cyan;">■</span> Collection.Portals</td> <td style="text-align: right;">0.25</td> </tr> <tr> <td><span style="color: green;">■</span> Portal_1</td> <td style="text-align: right;">0.25</td> </tr> <tr> <td><span style="color: green;">■</span> Portal_2</td> <td style="text-align: right;">0.25</td> </tr> <tr> <td><span style="color: green;">■</span> Portal_3</td> <td style="text-align: right;">0.25</td> </tr> </tbody> </table> <p style="text-align: right;">+ - ↶</p> </div> <p style="text-align: center;"><b>Journey origins with automatic weights</b></p>	Name	Weight	<span style="color: cyan;">■</span> Collection.Portals	0.25	<span style="color: green;">■</span> Portal_1	0.25	<span style="color: green;">■</span> Portal_2	0.25	<span style="color: green;">■</span> Portal_3	0.25
Name	Weight										
<span style="color: cyan;">■</span> Collection.Portals	0.25										
<span style="color: green;">■</span> Portal_1	0.25										
<span style="color: green;">■</span> Portal_2	0.25										
<span style="color: green;">■</span> Portal_3	0.25										

Description	Meaning
	 <p style="text-align: center;"><b>Collection members with automatic weights</b></p>
<p><b>Manual weights in list</b></p>	<p>Each item in the list of origins is assigned a manual weight; the two 'equalizer' buttons can be used to quickly set all weights equal or normalize them (force them 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 style="text-align: center;"><b>Journey origins with manual weights</b></p>  <p style="text-align: center;"><b>Collection members with manual weights</b></p>



## 4.2.4 Working with Colours

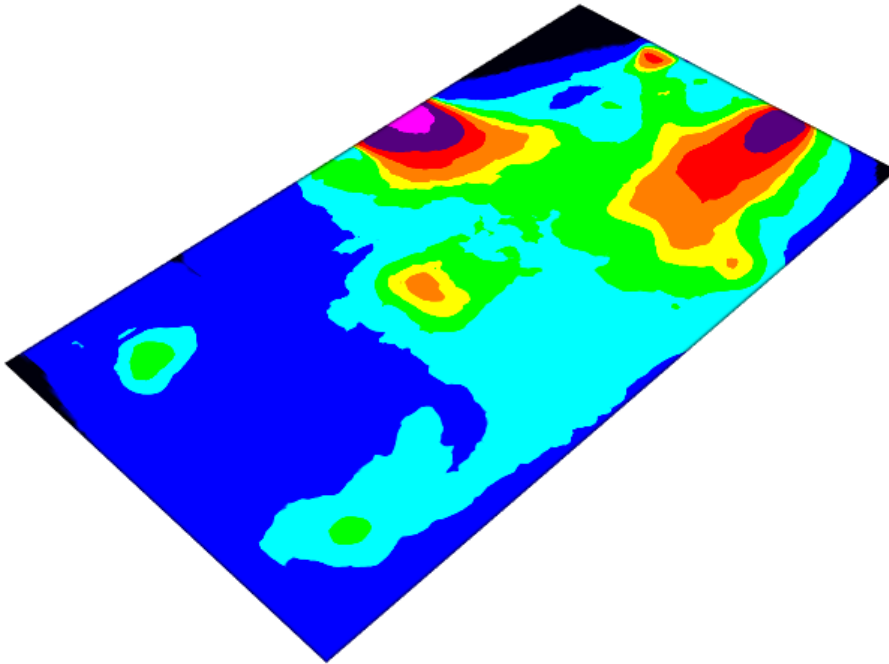
### Individual Colours

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. Note that transparency can be set if desired 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).

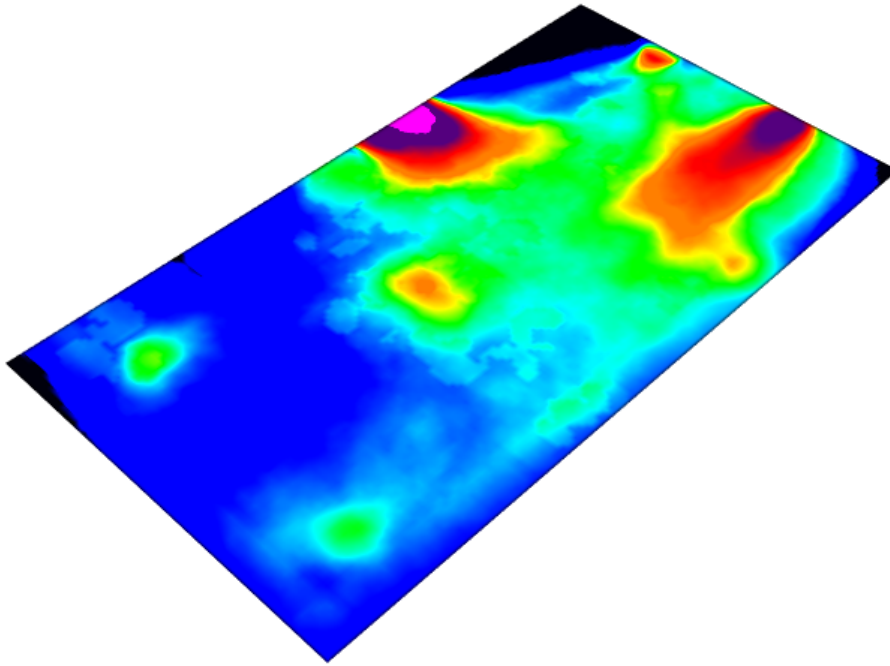
Alternatively, the swatch can be right-clicked to select from a small number of built-in colours or reset the colour to its default value (e.g., light blue for a [floor](#)).

### 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 cutoff values in between colours. In most cases, these cutoff 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 cutoff values. The first colour is used for all values less than the first cutoff value and the last colour is used for all values greater than the last cutoff value. If sharp contours are used, the second colour is used for all values between the first and second cutoff value, the third colour is used for all values between the second and third cutoff values, and so on. If a gradient is used, colour values are smoothly interpolated across each cutoff value except for the first and last, which remain as sharp cutoffs. For example, the images below show the relationship between numerical value and colour for cutoff 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 cutoff will not be visible since it typically has an associated cutoff value of 0.0, and often no values are less than or equal to 0.0.)



**Contour colouring**



**Gradient colouring**

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 cutoff 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 cutoff value and associated colour; the cutoff value chosen should

be one that you do not expect to ever reach. In the above example, a cutoff 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 cutoff value of 8.0; however, if the measured value (density, distance, time, cost etc.) ever exceeds 12.0, a sharp cutoff 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.

#### 4.2.5 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
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

Description	Meaning
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; an absolute time of 09:00:00 should be used in this case instead.
Event start	A specified offset from the start of a specified event (negative offset values can be used if necessary).
Simulation start	A specified offset from the start time of the simulation.
Simulation end	A specified offset from the end time of the simulation (in this case, a negative offset value should probably be used).

**Time Reference Ranges**

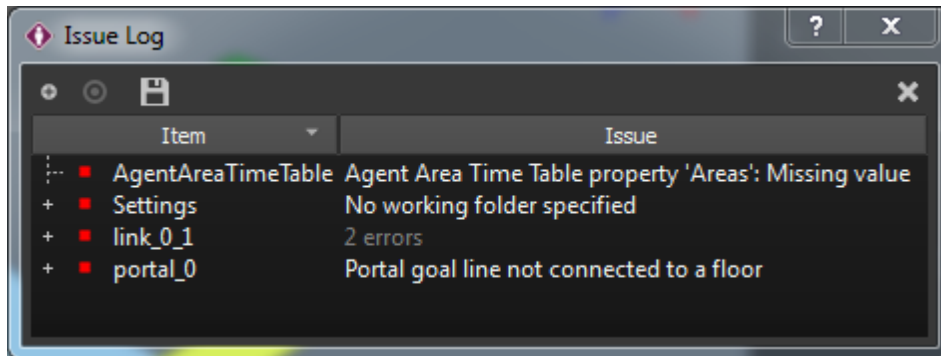
In some places, time ranges can be specified using time references (as above) instead of simple times. In these cases the following options are possible:

Description	Meaning
All after	From the specified time reference to the end of the simulation
All before	From the beginning of the simulation to the specified time reference
All simulation	The entire simulation
Between times	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

## 4.2.6 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 "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" bullseye button. The contents of the issue window can be saved to an external CSV file.

## 4.2.7 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 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>	Press and hold the 'S' key, then press and hold the left mouse button while moving the mouse.  Alternative: Hold the middle mouse button while moving the mouse.
<b>Rotate</b>	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 the object and press the 'F' key.  Alternative: Hold the shift key and middle mouse button while moving the mouse.

Camera Manipulation Controls	
<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 zoom to frame within the view 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>	Selects an object or agents in the scene. Click and drag to "box select" multiple 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>Q</b>	Enter or exit process chain edit mode for editing the connections between servers .
<b>Space</b>	Return to object selection mode. Any <a href="#">manipulations</a> will also be stopped.

Object Selection Controls	
<b>Escape</b>	Deselect all and return to object selection mode. Any <a href="#">manipulations</a> will also be stopped.

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 components have a separate context menu specific to the type of component.
<b>X, C, V</b>	Starts scaling, rotating and translation manipulations respectively. See <a href="#">Editing Geometry</a> for more information.
<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> for more information.
<b>M</b>	Opens the measure tool. See <a href="#">Model Panel</a> tools for more information.
<b>K</b>	Opens the knife tool. See <a href="#">Editing Geometry</a> for more information.
<b>Delete</b>	Delete selected objects or components.

## 4.2.8 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 clipping objects, at the cost of rendering accuracy (objects will be more likely to appear to overlap each other).
<b>Scene View: Far clip distance</b>	Maximum distance from the camera at which objects are visible. Increase as necessary for very large scenes, at the cost of rendering accuracy (objects will be more likely to appear to overlap each other).
<b>Reset</b>	Reset all general to MassMotion defaults.

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: Reference axes</b>	Whether the set of reference axes should be displayed by default during movie/image export.
<b>Reset</b>	Reset all movie/image preferences to MassMotion defaults.

## 4.3 Objects

As described in the [Objects](#) page of the user guide, a MassMotion project consists primarily of a set of objects. Objects can be classified as:

- [Scene](#): physical objects such as floors, which have a physical presence in the scene
- [Activities](#): create agents, modify agents, or perform operations on the scene
- [Analysis](#): queries and related objects that are used to analyze the results of a simulation

This section describes the properties of each individual type of object in detail.

### 4.3.1 Scene Objects

Scene Object Type	Description
<a href="#">Collections</a>	<p>Collections are containers of other objects. Some collections are used during the simulation and others are for modeling and analysis.</p> <p><b>Bank</b>: Represents a set of nearby connections between two objects. Affects agents in simulation.</p> <p><b>Collection</b>: A general multi-purpose collection that can store any non-collection object.</p> <p><b>Perimeter</b>: A set of connections which cannot be crossed more than once. Affects agents in simulation.</p> <p><b>Zone</b>: A conceptual area in the simulation.</p> <p><b>Reference Model</b>: Created automatically to contain the <a href="#">reference geometry</a> objects from an imported geometry file.</p>
<a href="#">Barrier</a>	Obstacles that block agents from passing through.
<a href="#">Escalator</a>	A vertical <a href="#">connection object</a> representing an escalator. Escalators must be unidirectional.



<a href="#">Floor</a>	Floors represent any flat area in which agents may walk.
<a href="#">Link</a>	A <a href="#">connection object</a> which connects floors horizontally. May represent a doorway or turnstile.
<a href="#">Path</a>	A <a href="#">connection object</a> constructed from a curve segment. Does not need to be horizontal.
<a href="#">Portal</a>	Represents entrances into the simulation as well as agent destinations.
<a href="#">Ramp</a>	A vertical <a href="#">connection object</a> which represents a ramp.
<a href="#">Reference Geometry</a>	<p>Imported geometry from external sources. Geometry must be used to generate other MassMotion scene objects before they can be used.</p> <p>The following files can be imported:</p> <ul style="list-style-type: none"> <li>• 3DS Max (.3ds)</li> <li>• Collada (.dae)</li> <li>• AutoCAD Drawing Exchange Format (.dxf)</li> <li>• FBX (.fbx)</li> <li>• Industry Foundation Classes (.ifc)</li> <li>• Wavefront (.obj)</li> </ul>
<a href="#">Server</a>	Used to model queues and more complex agent behaviour.
<a href="#">Stair</a>	A vertical <a href="#">connection object</a> which represents a stair.
<a href="#">Visual</a>	Cosmetic objects that enhance the look of the scene.

#### 4.3.1.1 Collections

Collections are a versatile way of using and interacting with 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.

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.

##### 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="#">analysis regions</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="#">Analysis 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>Regions</b>	<a href="#">Analysis Regions</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>Walkables</b>	Walkable objects ( <a href="#">floors</a> , <a href="#">links</a> , etc.).

**Types of Collections**

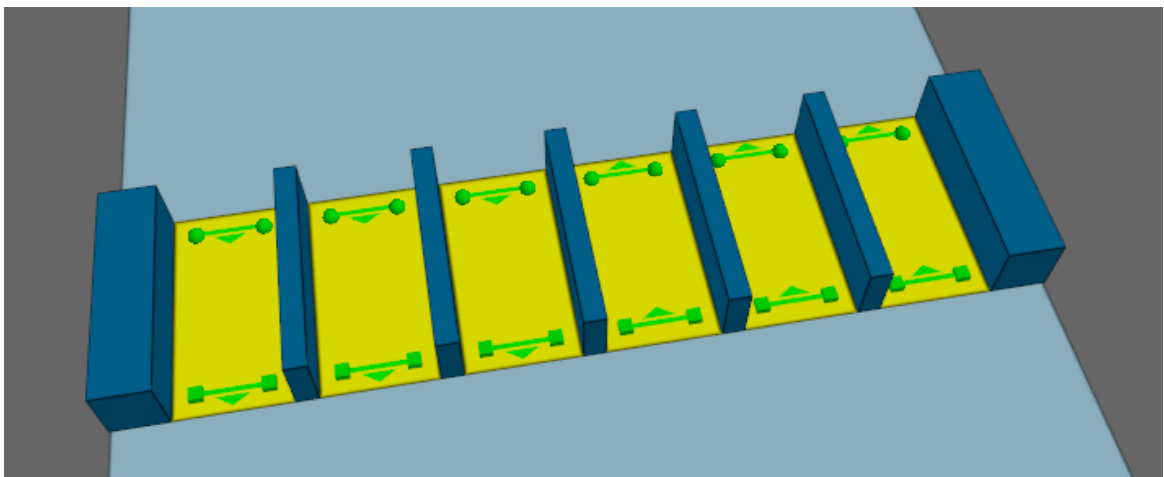
<b>Collection Type</b>	<b>Description</b>
<a href="#">Bank</a>	A collection of connection objects which connect the same two objects. Banks help agents navigate clusters of similar and closely placed connections.
<a href="#">General Collection</a>	A collection of any type of object except other collections.
<a href="#">Perimeter</a>	A collection of connection objects encircling a conceptual area of interest. Agents cannot cross the same perimeter twice.
<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.
<a href="#">Reference Model</a>	A collection of <a href="#">reference geometry</a> objects which were imported from the same file. These are created automatically by importing geometry.

## 4.3.1.1.1 Bank

A 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 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> ).

4.3.1.1.2 Collection

A general collection can store objects of any type except other [collections](#). They can be used as conceptual or spatial groupings of other objects. Objects can belong to more than one collection.

Collection Properties	
<b>Name</b>	The name of the collection.
<b>Objects</b>	Members of the collection. These may be any non-collections object. The weights of objects can be set here if set to manual. See <a href="#">Choosing Objects</a> for more details.

4.3.1.1.3 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 hidden from 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.

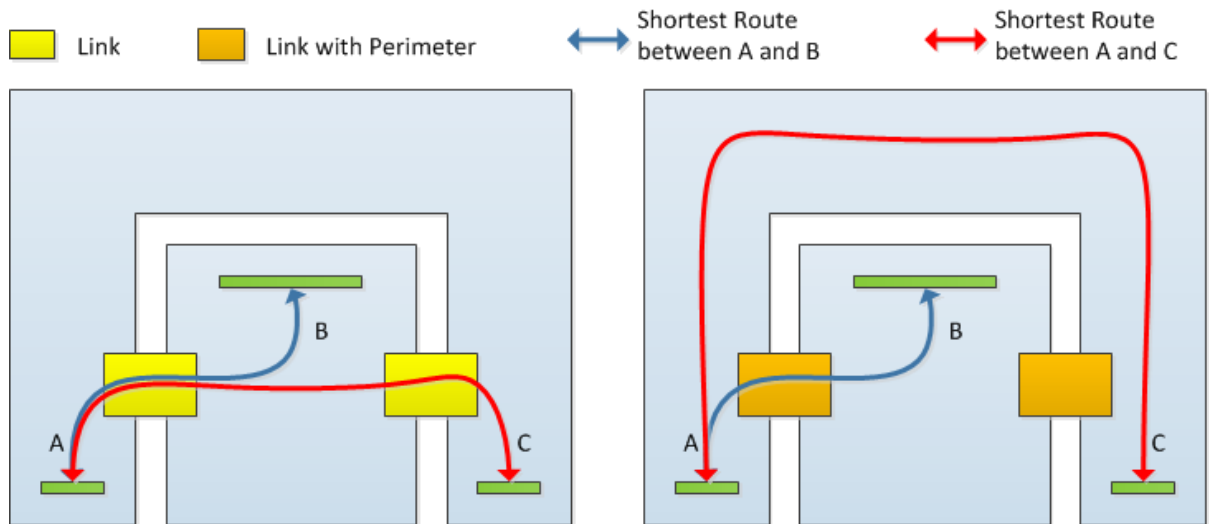


Figure 1: Example floor plan with and without a perimeter.

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.

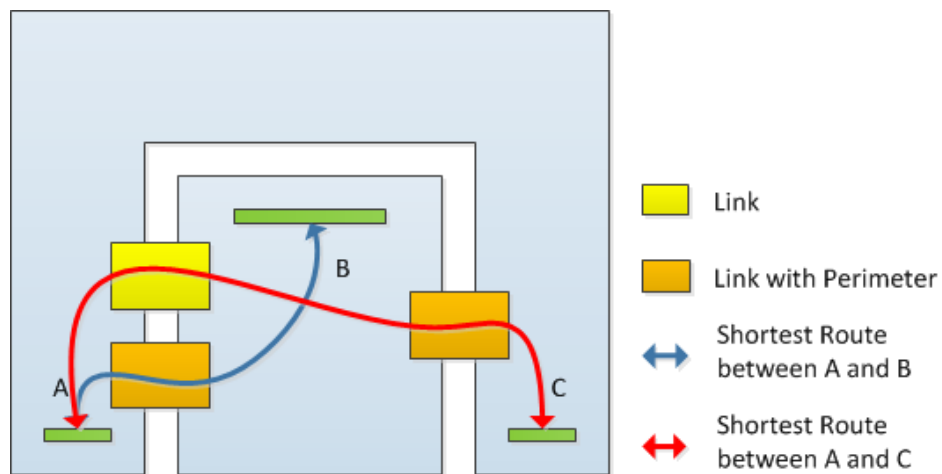


Figure 2: Example floor plan with incomplete perimeter.

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> (ie. <a href="#">links</a> , <a href="#">stairs</a> , <a href="#">escalators</a> , <a href="#">ramps</a> and <a href="#">paths</a> ).

4.3.1.1.4 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 connections 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 connections 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>

4.3.1.1.5 Reference Model

Reference models are automatically generated when importing geometry and will contain all imported [reference geometry](#) objects from a single file.

[Reference geometry](#) objects cannot be directly edited or modified and so reference model objects provide a way to reposition them to fit into the scene. All objects from a single file can be scaled, rotated and translated in bulk using the reference model.

Deleting a reference model will delete all associated [reference geometry](#) objects from the scene. Reference geometry objects can also be deleted individually.

Reference Model Properties	
<b>Scale, Rotation, Translation</b>	<p>These can be modified to rescale and move reference geometry objects to better fit the scene. Scale will be applied first, then rotation, then translation. Note that this means that the translation values will be in metres, the MassMotion unit of measurement (rather than the original file's</p>

	units) since the translation values are applied after the model has already been scaled from its original units into metres.
<b>Reference Geometry Objects</b>	The reference geometry objects included from the file.

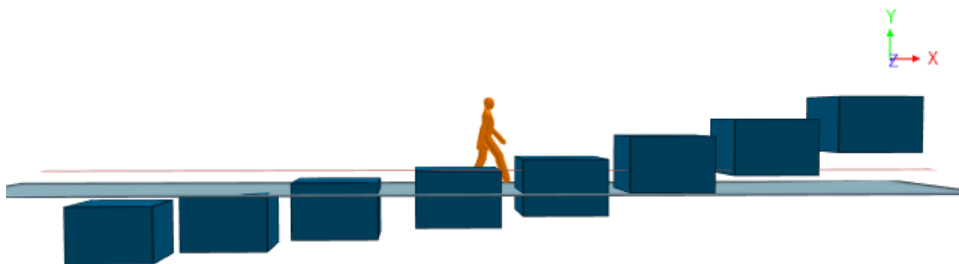
#### 4.3.1.2 Barrier

A barrier describes a region through which agents cannot pass. 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 to 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 removed by error during simulation. If dividing the floor into unconnected areas is necessary, use separate floors for each area instead.

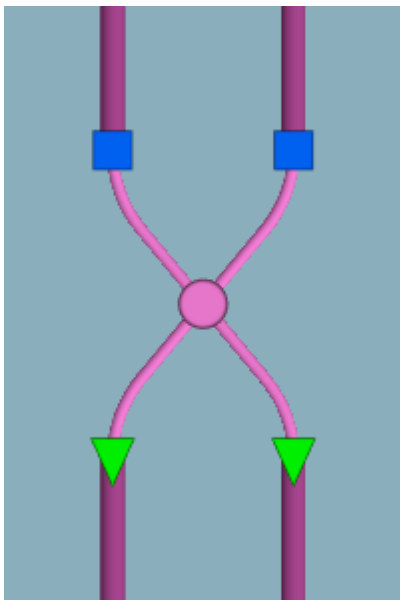
**Properties**

There are no properties for objects of this type.

**4.3.1.3 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 to downstream servers:</p> <p><b>Assign to smallest queue:</b> agents will be sent to the server with the smallest queue (absolute count, not proportional) in the downstream set</p> <p><b>Assign randomly:</b> agents will be sent to a randomly determined server in the downstream set. Only servers with unused capacity will be considered for selection.</p> <p>Note that token based restrictions on server access will be respected for all</p>

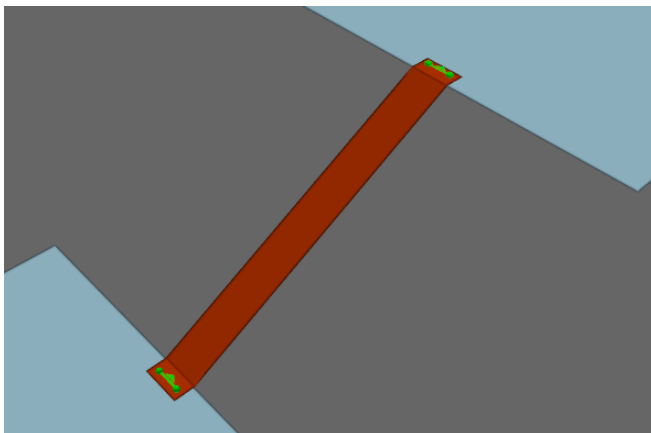


	assignment methods.
<b>Sources</b>	The list of connected upstream servers which feed into this dispatch. Sources can be added or removed using the multi-object chooser in the Dispatch properties page or via the connect and disconnect functions in the "Edit process chains" mode in the main application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

#### 4.3.1.4 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 1m wide, with 0.5m landings at each end where the ball connection will be at the bottom and the box connection will be at the top.

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](#) objects.
- 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 average speed as specified in the escalator properties.

**Properties**

General Tab	
<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>	Defines any distance penalty or modified cost to be added to the escalator. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information.
<b>Costs: Queue multiplier</b>	Increases the penalty for queuing. A higher number will increase the perceived time cost of queuing making agents more likely to seek alternate options when there is queuing. See <a href="#">Agent Navigation</a> for more information.
<b>Physical: Map Resolution</b>	Determines the resolution of surface maps created for agent navigation. Smaller values describe escalator and barrier edges more accurately resulting in more precise <a href="#">walkable area definitions</a> , but increase memory and processing requirements during simulation.
<b>Physical: Effective Width</b>	Defines how the width of the escalator's connections is calculated. The width is used to calculate queue times (how long it will take for a queue of agents to clear) and controlled flow if using people/minute/metre.  <b>Calculate from Geometry:</b> The default setting where the width (m) is calculated from the vertices of the escalator edges. <b>Specify Manually:</b> Allows the effective width (m) of both the box and ball connections to be set to specific values.
<b>Tread Speed</b>	Defines the speed of the escalator along the incline, measured in metres per second.

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate, closing the objects to all agents unless explicitly opened through events (see <a href="#">Open Gate Event</a> ).
<b>Gate: Wait style:</b>	Sets the behaviour of agents waiting for the escalator when the gate is closed.  <b>Stand Still:</b> Agents will stop moving after stepping onto a floor connected to the escalator. <b>Spread Out:</b> Agents will spread out on the floor after stepping onto a floor connected to the escalator. <b>Focus On Target:</b> Agents will move to and wait by the escalator.
<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>	Sets whether the flow rate is calculated as people/minute or people/minute/metre.  If using people/minute/metre, the effective width (general tab property) is used.
<b>Limit Flow: Max rate</b>	Sets the maximum flow rate. Flow rate will be measured according to the rate type shown above.

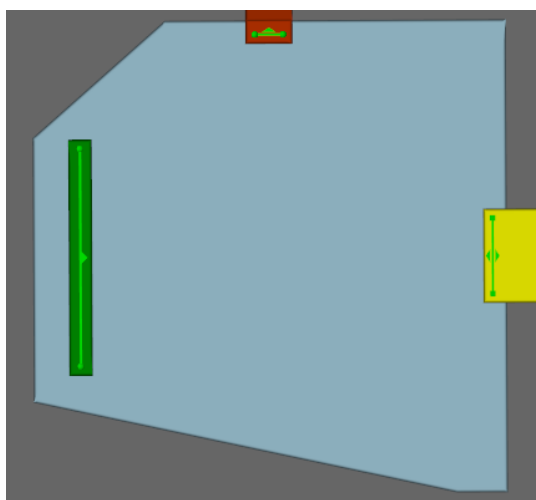
<b>Agent Behaviour tab</b>	
<b>Delay On Enter</b>	Sets how long an agent will be delayed before stepping onto an escalator. This can be set as a distribution in seconds, see <a href="#">Single Value Distributions</a> for more information.  If an escalator is gated and closes before the delay finishes, the agent will wait until the next time the escalator opens before attempting to step on again.
<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: Walk type</b>	Defines the manner in which agents will cross the escalator.  <b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories. <b>Ignore Barriers:</b> As above with the exception that agents will ignore any barriers on the surface. This is the default for escalators. <b>Virtual:</b> Agents will move instantly across the surface to their intended goal, ignoring all obstructions on the surface. Commonly used for areas of a simulation environment where the agent route choices are important but crowding characteristics are not important.
<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.
<b>Body Radius: Radius</b>	Sets the agent radius in metres.

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

### 4.3.1.5 Floor

Floors are the most fundamental object types in a MassMotion environment. 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) but 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 is not required to be flat, but elevation 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>	Defines any distance penalty or modified cost to be added to the floor. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres.
<b>Physical: Map resolution</b>	Determines the resolution of surface maps created for agent navigation. Smaller values describe floor and barrier edges more accurately resulting in more precise <a href="#">walkable area definitions</a> , but increase memory and processing requirements during simulation.

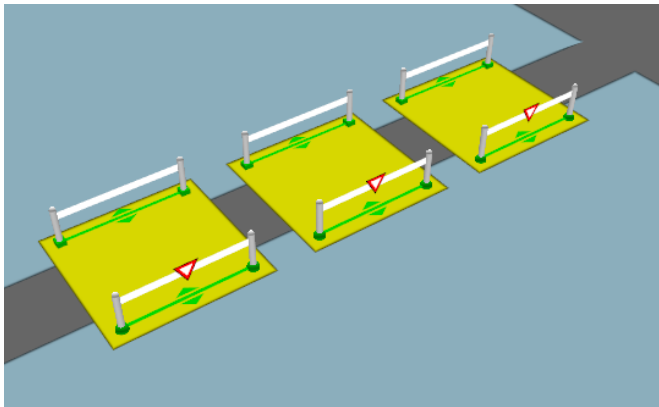
Agent Behaviour Tab	
<b>Traversal: Walk type</b>	<p>Defines the manner in which agents will cross the floor.</p> <p><b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories. This is the default for floors.</p> <p><b>Ignore Barriers:</b> As above with the exception that agents will ignore any barriers on the surface. Used mainly for connection objects (links, stairs, escalators, ramps).</p> <p><b>Virtual:</b> Agents will move instantly across the surface to their intended goal, ignoring all obstructions on the surface. Commonly used for areas of a simulation environment where the agent route choices are important but crowding characteristics are not important.</p>
<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 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>Body Radius: Radius</b>	Sets the agent radius in metres.
<b>Direction Bias: Set agent 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>Direction Bias: Direction</b>	Sets which direction agents will prefer to move.
<b>Speed Limit: Cap agent speed</b>	Enables a maximum threshold for agent speed. Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed Limit: Maximum</b>	Sets the maximum speed in metres per second.

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

#### 4.3.1.6 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

In order to support properly formed goal lines, geometry used for links should be very close to flat and approximately quadrilateral (a rectangle, trapezoid or parallelogram) although they may consist of a large number of individual triangles. 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 floors must be different (a link cannot be used to connect two portions of the same floor). Links must be at least 0.4m wide to allow proper generation of goal lines.

#### 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>	<p>Sets the direction in which agents can traverse the link.</p> <p><b>Two way:</b> Agents may cross in both directions  <b>Unidirectional (ie. Ball to Box or Box to Ball):</b> Agents may only cross in the specified direction.</p>
<b>Costs: Distance added</b>	<p>Defines any distance penalty or modified cost to be added to the link. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information.</p>

<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. See <a href="#">Agent Navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of surface maps created for agent navigation. Smaller values describe link and barrier edges more accurately resulting in more precise <a href="#">walkable area definitions</a> , but increase memory and processing requirements during simulation.
<b>Physical: Effective width</b>	Defines how the width of the link's connections is calculated. The width is used to calculate queue times (how long it will take for a queue of agents to clear) and controlled flow if using people/minute/metre.  <b>Calculate from Geometry:</b> The default setting where the width (m) is calculated from the vertices of the link edges. <b>Specify Manually:</b> Allows the effective width (m) of both the box and ball connections to be set to specific values.

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate, closing the objects to all agents unless explicitly opened through events (see <a href="#">Open Gate Event</a> ).
<b>Gate: Wait style:</b>	Sets the behaviour of agents waiting for the link when the gate is closed.  <b>Stand Still:</b> Agents will stop moving after stepping onto a floor connected to the link. <b>Spread Out:</b> Agents will spread out on the floor after stepping onto a floor connected to the link. <b>Focus On Target:</b> Agents will move to and wait by the link.
<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>	Sets whether the flow rate is calculated as people/minute or people/minute/metre.  If using people/minute/metre, the effective width (general tab property) is used.
<b>Limit Flow: Max rate</b>	Sets the maximum flow rate. Flow rate will be measured according to the rate type shown above.
<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.

Agent Behaviour tab	
<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: Walk type</b>	<p>Defines the manner in which agents will cross the link.</p> <p><b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories.</p> <p><b>Ignore Barriers:</b> As above with the exception that agents will ignore any barriers on the surface. Used mainly for connection objects (links, stairs, escalators, ramps).</p> <p><b>Virtual:</b> Agents will move instantly across the surface to their intended goal, ignoring all obstructions on the surface. Commonly used for areas of a simulation environment where the agent route choices are important but crowding characteristics are not important.</p>



<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 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>Body Radius: Radius</b>	Sets the agent radius in metres.
<b>Direction Bias: Set agent 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>Direction Bias: Direction</b>	Sets which direction agents will move.
<b>Speed Limit: Cap agent speed</b>	Enables a maximum threshold for agent speed. Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed Limit: Maximum</b>	Sets the maximum speed in metres per second.

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

#### 4.3.1.7 Path

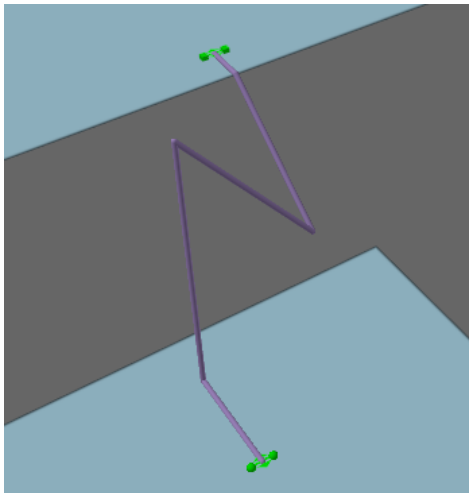
Paths are [connection objects](#) consisting of a simple curve. There are no restrictions on the shape of the curve, and they may be vertical or even intersect themselves and other geometry.

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 [Editing Geometry](#) for more information on editing paths.

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 will maintain a "Queue Spacing" (set in Agent Behaviour) between the

surfaces of the cylinders.



A path

**Restrictions on Geometry:**

- A path must contain a single curve object.
- Each end point of the curve must be positioned just *above* a separate floor.
- There are no restrictions on the path described by the curve provided that it is continuous.
- 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.

**Properties**

General Tab	
<b>Direction</b>	<p>Sets the direction in which agents can traverse the path.</p> <p><b>Two way:</b> Agents may cross in both directions  <b>Unidirectional (ie. Ball to Box or Box to Ball):</b> Agents may only cross in the specified direction.</p>
<b>Costs: Distance added</b>	<p>Defines any distance penalty or modified cost to be added to the path. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information.</p>
<b>Costs: Queue multiplier</b>	<p>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. See <a href="#">Agent Navigation</a> for more information.</p>
<b>Physical: Effective width</b>	<p>Paths are represented as lines, but the goal lines onto and off of the lines still have width.</p> <p>Defines how the width of the path's connections is calculated. The width is used to calculate queue times (how long it will take for a queue of agents</p>

	<p>to clear) and controlled flow if using people/minute/metre.</p> <p><b>Calculate from Geometry:</b> The default setting where the width (m) is calculated from the goal line width.</p> <p><b>Specify Manually:</b> Allows the effective width (m) of both the box and ball connections to be set to specific values.</p>
--	---

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate, closing the objects to all agents unless explicitly opened through events (see <a href="#">Open Gate Event</a> ).
<b>Gate: Wait style:</b>	<p>Sets the behaviour of agents waiting for the path when the gate is closed.</p> <p><b>Stand Still:</b> Agents will stop moving after stepping onto a floor connected to the path.</p> <p><b>Spread Out:</b> Agents will spread out on the floor after stepping onto a floor connected to the path.</p> <p><b>Focus On Target:</b> Agents will move to and wait by the path.</p>
<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>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</p>

	<p>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>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: Walk type</b>	<p>Defines the manner in which agents will cross the path.</p> <p><b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories.</p> <p><b>Ignore Barriers:</b> Identical to 'Standard' for paths, as agents always ignore barriers when traversing paths.</p> <p><b>Virtual:</b> Agents will move instantly to the other end of the path. When a path is virtual, route distance calculations ignore the internal geometry of the path and measure the straight line distance between the two path end points.</p>
<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 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

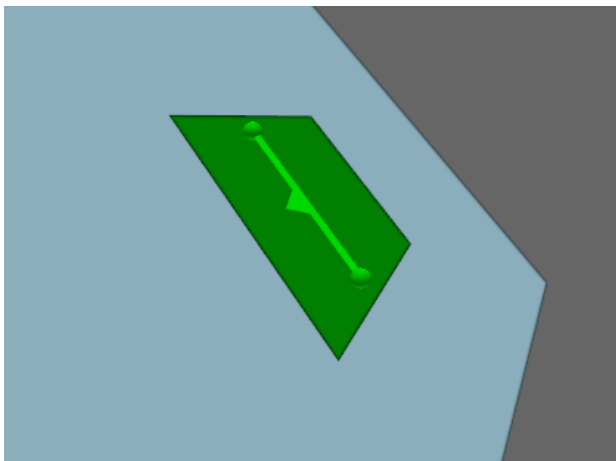
	of their profile settings, and b) agent movement has only been validated for radius values close to 0.25m.
<b>Body Radius: Radius</b>	Sets the agent radius in metres.  Agents on a path will see each other as cylinders with a radius equal to their body radius and will maintain a space between each other set by their "Queue Spacing".
<b>Speed Limit: Cap agent speed</b>	Enables a maximum threshold for agent speed.
<b>Speed Limit: Maximum</b>	Sets the maximum speed in metres per second. Agents traveling above the specified maximum will have their speed reduced to the maximum.

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

#### 4.3.1.8 Portal

Portals act as both entrances and destinations for agents in the simulation. All portals can serve as an entrance, specified through agent creation events. Portals are by default also enabled as destinations, allowing agents to seek them as intermediate destinations or final simulation exits.

Destination portals require additional time to initialize during simulations. If simulation initialization is too time consuming, consider changing some portals into "Entrance Only".



A portal

#### Restrictions on Geometry

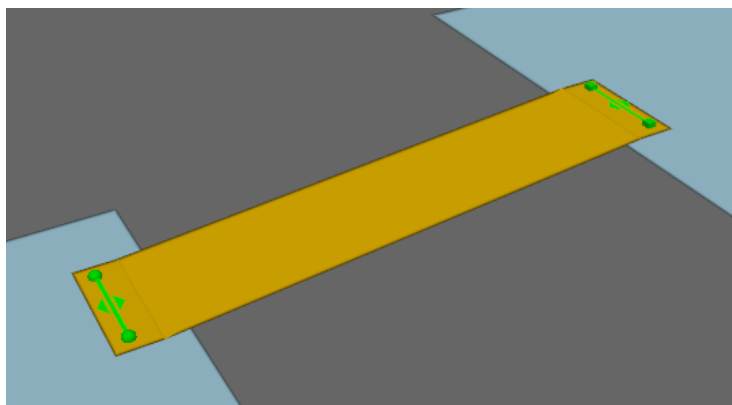
- Portals must be flat.
- Portals must be approximately rectangular or trapezoidal, to allow proper construction of the goal line.
- Portals must be positioned between 0.01m and 0.2m above a floor and no closer than 0.2m to a floor edge.

**Properties**

General Tab	
<b>General: Type</b>	<p>How events and agents will interact with the portal.</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.</p> <p><b>Along Spawn Line:</b> Agents appear randomly distributed on the longest midline of the planar, rectangular portal geometry.</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 boundary of the floor beneath the portal. Agents will only appear in valid unobstructed regions of the floor, but may appear on top of existing agents in areas of high density.</p>
<b>Agent Placement: Start Angle</b>	<p>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.</p>

**4.3.1.9 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](#) 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 (ie. Ball to Box or Box to Ball):</b> Agents may only cross in the specified direction.
<b>Costs: Distance added</b>	Defines any distance penalty or modified cost to be added to the ramp. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information.
<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. See <a href="#">Agent Navigation</a> for more information.
<b>Physical: Map resolution</b>	Determines the resolution of surface maps created for agent navigation. Smaller values describe ramp and barrier edges more accurately resulting in more precise <a href="#">walkable area definitions</a> , but increase memory and processing requirements during simulation.
<b>Physical: Effective width</b>	Defines how the width of the stair's connections is calculated. The width is used to calculate queue times (how long it will take for a queue of agents to clear) and controlled flow if using people/minute/metre.  <b>Calculate from Geometry:</b> The default setting where the width (m) is

	<p>calculated from the vertices of the ramp edges.</p> <p><b>Specify Manually:</b> Allows the effective width (m) of both the box and ball connections to be set to specific values.</p>
--	--

Access Tab	
<b>Gate: Enable use as gate</b>	Configures the object as a gate, closing the objects to all agents unless explicitly opened through events (see <a href="#">Open Gate Event</a> ).
<b>Gate: Wait style:</b>	<p>Sets the behaviour of agents waiting for the ramp when the gate is closed.</p> <p><b>Stand Still:</b> Agents will stop moving after stepping onto a floor connected to the ramp.</p> <p><b>Spread Out:</b> Agents will spread out on the floor after stepping onto a floor connected to the ramp.</p> <p><b>Focus On Target:</b> Agents will move to and wait by the ramp.</p>
<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>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>



	<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.
<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.

Agent Behaviour tab	
<b>Delay On Enter</b>	<p>Sets how long an agent will be delayed before stepping onto a 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: Walk type</b>	<p>Defines the manner in which agents will cross the ramp.</p> <p><b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories. Most like the real world.</p> <p><b>Ignore Barriers:</b> As above with the exception that agents will ignore any barriers on the surface. Used mainly for connection objects (links, stairs, escalators, ramps).</p> <p><b>Virtual:</b> Agents will move instantly across the surface to their intended goal, ignoring all obstructions on the surface. Commonly used for areas of a simulation environment where the agent route choices are important but crowding characteristics are not important.</p>
<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 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>Body Radius:</b>	Sets the agent radius in metres.

<b>Radius</b>	
<b>Direction Bias: Set agent 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>Direction Bias: Direction</b>	Sets which direction agents will move.
<b>Speed Limit: Cap agent speed</b>	Enables a maximum threshold for agent speed.
<b>Speed Limit: Up</b>	Sets the maximum upward speed in metres per second.
<b>Speed Limit: Down</b>	Sets the maximum downward speed in metres per second.

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

#### 4.3.1.10 Reference Geometry

When a geometry file is imported using the 'Scene' tab of the main window's ribbon, each individual object within the file is placed in the scene as a reference geometry object. These objects have no impact on simulation and other types of scene objects must be created from them before use. The "Use to Generate" sub-menu available when right clicking reference geometry can be used to do this. Using a reference geometry object to generate another object will hide the reference geometry and mark it as used. Unused reference geometry can be found via the [list view](#) by filtering for "Unused Imports".

Alternatively, sub-sections of reference geometry can be used to generate separate scene objects. This can be done by entering face selection mode, selecting the desired faces and right clicking them to create new objects. In this case, reference geometry should be marked as used manually.

Reference geometry cannot be edited, although individual reference geometry objects can be deleted after importing a file. Instead, a [reference model](#) object is created on import which contains all of a file's reference geometry. The reference model can be scaled, rotated and translated to match the MassMotion model's coordinates.

#### Supported File Formats

The following files can be imported:

- 3DStudio (.3ds)
- Collada (.dae)
- AutoCAD Drawing Exchange Format (.dxf)
- FBX (.fbx)
- Industry Foundation Classes (.ifc)

- Wavefront (.obj)

IFC files contain additional information about their geometry and create IFC specific types of reference geometry. These can be used to automatically generate a corresponding scene object shown in the table below.

All other file formats produce "Generic Reference Geometry".

IFC Type	Automatically Generated MassMotion Type
Wall, Wall Standard Case, Railing, Column, Furnishing Element	<a href="#">Barrier</a>
Escalator, Moving Walkway	<a href="#">Escalator</a>
Space, Landing Slab	<a href="#">Floor</a>
Door	<a href="#">Link</a>
Stair, Stair Flight	<a href="#">Stair</a>
Element, Floor Slab, Base Slab, Unrecognized Slab, Elevator, Unrecognized Transport Element, Plate, Building Element Proxy	None - must be explicitly specified

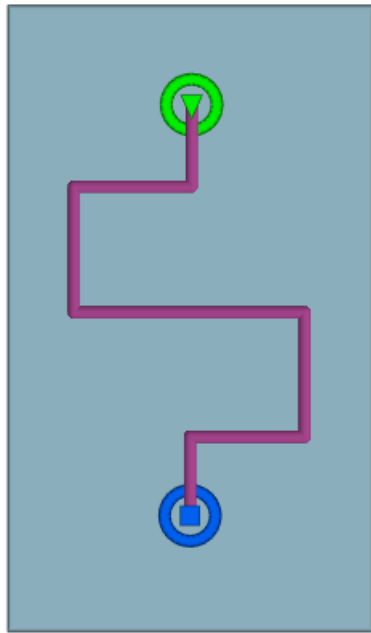
### Properties

There are no properties for objects of this type.

#### 4.3.1.11 Server

Servers model queuing behaviour and more complex interactions between agents and their environments.

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. Like [Paths](#) 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.

#### Approach

Agents will only approach a server that has available capacity as defined by the input buffer capacity. If an agent has chosen to use 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 buffer line) and approaches the processor according to the approach type (see the "General" properties below). 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, freeing up space for agents that would like to approach the server. 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 by token. If an agent holds one of the given tokens the agent will use the corresponding distribution instead of the common distribution. See the "General" properties below 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:

*Server is the end of a process chain* The agent finishes the 'Seek Process' task and continues with its next task.

*Server is in the middle of a process chain and there is downstream capacity* The agent approaches the next server in the chain.

*Server is in the middle of a process chain and there is no downstream capacity* The agent is held in place until there is available downstream capacity (possibly blocking access to the current processor).

### Filtering by Token

A server can be configured to give exclusive or preferential access to agents based on whether or not they hold a specific token. Servers that require a specific token will only accept agents that hold the token. Servers that prefer a specific token will accept any agent, but when given a choice will always choose an agent that holds the preferred token.

If an agent is sent to a group of servers that require tokens that the agent does not have, the agent will be removed from the simulation and the error logged.

If two servers are grouped and the first server is configured to ignore tokens while the second is configured to prefer or require a token, all agents holding that token will be sent to the second server, regardless of available capacity in the first. Only once the second server has reached capacity will agents with the preferred or required token ever consider the first server.

*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.

### 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.

### Properties

General Tab	
<b>Approach Type</b>	<b>Use Line: Enter At Start:</b> Agents approaching the server will move across the floor towards the beginning of the buffer line (identified by the entrance arrow) and then move along the line until they reach the processing end. Agents will move in single file along the curve following behind the agent in front. Only the agent at the front of the line can be processed. In the case where the line become full, new agents will continuously try to board the line, clustering around the line entrance.

	<p><b>Ignore Line:</b> Agents will ignore the buffer line and head across the floor directly to the processing end of the server (identified by the box). They will begin processing as soon as they reach the box, assuming there is available processing capacity.</p>
<b>Capacity</b>	<p>Defines the maximum number of agents accepted by the server (not counting those being processed).</p> <p><b>Infinite:</b> there is no constraint placed on the number of agents accepted by the server. All agents are free to approach.</p> <p><b>Finite:</b> The server will only accept a certain number of new agents based on the input buffer's available capacity. Agents will not be permitted to approach the server until there is available capacity. The available capacity is calculated as:</p> <p style="text-align: center;"><i>"Available Capacity" = "Max Capacity" - "Count of Agents Waiting to be Processed"</i></p>
<b>Agent Spacing Distribution</b>	<p>The minimum distance between two agents on the path. Each agent will be given a target minimum distance according to this distribution, and will always stay at least that distance back from the agent in front.</p>
<b>Token Access</b>	<p><b>Ignore Token:</b> Server accepts all agents.</p> <p><b>Prefer Token:</b> Server accepts all agents, but given a choice will always choose the agent holding the specified token.</p> <p><b>Require Token:</b> Server will only accept agents holding the specified token.</p> <p>Consume token check box indicates if the token will be removed from the agent after entering the server object.</p>
<b>Contact Time</b>	<p>Defines the length of time taken to process an agent. This is the default contact time that is applied to all agents that do not hold one of the tokens specified by the optional contact time by token entries.</p>
<b>Optional Contact Times By Token</b>	<p>Additional specialized contact times. If an agent holds one of the specified tokens, the corresponding contact time is used. If the agent does not hold any of the tokens or no tokens are specified, the default contact time is used. In the case where an agent holds multiple defined tokens, one of the token based contact times is used, but the process of selection is undefined.</p>

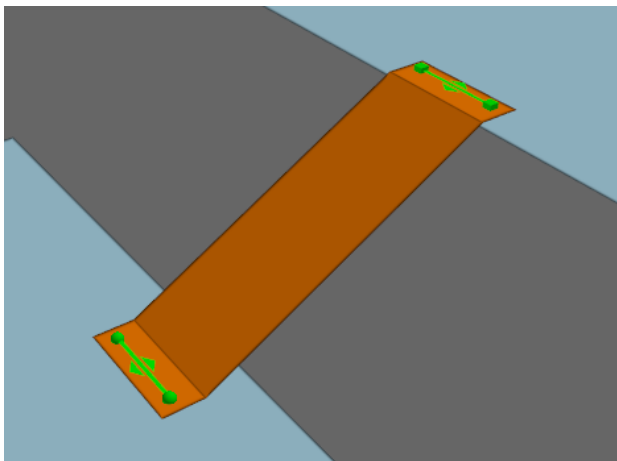
<b>Actions Tab</b>	
<b>Finished Processing</b>	<p>The specified action is applied to each agent immediately after it has finished being processed. In the case where the server is connected to other downstream servers and the agent is held in place for lack of downstream capacity, the action is still applied after processing (contact</p>

	time) regardless of whether or not the agent is free to move on to the next server.
--	---

Connections Tab	
<b>Source</b>	If the server is connected to an upstream Dispatch 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 Dispatch 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.

#### 4.3.1.12 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](#) 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>	<p>Sets the direction in which agents can traverse the stair.</p> <p><b>Two way:</b> Agents may cross in both directions  <b>Unidirectional (ie. Ball to Box or Box to Ball):</b> Agents may only cross in the specified direction.</p>
<b>Costs: Distance added</b>	<p>Defines any distance penalty or modified cost to be added to the stair. A positive value will add a distance (and hence make the route less desirable for agents), and a negative value will remove a distance. Distance is measured in metres. See <a href="#">Agent Navigation</a> for more information.</p>
<b>Costs: Queue multiplier</b>	<p>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. See <a href="#">Agent Navigation</a> for more information.</p>
<b>Physical: Map resolution</b>	<p>Determines the resolution of surface maps created for agent navigation. Smaller values describe stair and barrier edges more accurately resulting in more precise <a href="#">walkable area definitions</a>, but increase memory and processing requirements during simulation.</p>
<b>Physical: Effective width</b>	<p>Defines how the width of the stair's connections is calculated. The width is used to calculate queue times (how long it will take for a queue of agents to clear) and controlled flow if using people/minute/metre.</p> <p><b>Calculate from Geometry:</b> The default setting where the width (m) is calculated from the vertices of the stair edges.  <b>Specify Manually:</b> Allows the effective width (m) of both the box and ball connections to be set to specific values.</p>

**Access Tab**



<b>Gate: Enable use as gate</b>	Configures the object as a gate, closing the objects to all agents unless explicitly opened through events (see <a href="#">Open Gate Event</a> ).
<b>Gate: Wait style:</b>	<p>Sets the behaviour of agents waiting for the stair when the gate is closed.</p> <p><b>Stand Still:</b> Agents will stop moving after stepping onto a floor connected to the stair.</p> <p><b>Spread Out:</b> Agents will spread out on the floor after stepping onto a floor connected to the stair.</p> <p><b>Focus On Target:</b> Agents will move to and wait by the stair.</p>
<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>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</b>	When enabled, agents in the counter-flow direction will move aside to

<b>aside</b>	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>Delay On Enter</b>	<p>Sets how long an agent will be delayed before stepping onto a 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: Walk type</b>	<p>Defines the manner in which agents will cross the stair.</p> <p><b>Standard:</b> Agents will consider other agents, barriers, and personal speed characteristics to calculate their trajectories. Most like the real world.</p> <p><b>Ignore Barriers:</b> As above with the exception that agents will ignore any barriers on the surface. Used mainly for connection objects (links, stairs, escalators, ramps).</p> <p><b>Virtual:</b> Agents will move instantly across the surface to their intended goal, ignoring all obstructions on the surface. Commonly used for areas of a simulation environment where the agent route choices are important but crowding characteristics are not important.</p>
<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 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.
<b>Body Radius: Radius</b>	Sets the agent radius in metres.
<b>Direction Bias: Set agent 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>Direction Bias: Direction</b>	Sets which direction agents will move.

<b>Speed Limit: Cap agent speed</b>	Enables a maximum threshold for agent speed. Agents traveling above the specified maximum will have their speed reduced to the maximum.
<b>Speed Limit: Up</b>	Sets the maximum upward speed in metres per second.
<b>Speed Limit: Down</b>	Sets the maximum downward speed in metres per second.

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

#### 4.3.1.13 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.

### 4.3.2 Activity Objects

Activity objects have to do with creating agents, defining agent characteristics and behaviour, and controlling elements of the scene.

#### Defining Agents

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.

#### Creating Agents

Object Type	Description
<a href="#">Journey</a>	An event to create agents moving from origin portals to destination portals.
<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 subway car.
<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="#">Execute Action</a>	An event which applies an <a href="#">agent action</a> to change the behaviour of agents.
<a href="#">Open Gate</a>	An event which opens gated <a href="#">connection objects</a> .
<a href="#">Reference 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.

**Other Objects**

Object Type	Description
<a href="#">Agent Actions</a>	An operation that can be applied to an agent to modify its properties or assign it new tasks.
<a href="#">Agent Tests</a>	A condition which when applied to an agent returns true or false.
<a href="#">Token</a>	Identifying markers held by agents.

**4.3.2.1 Agent 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 when they 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	
<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 combine tests and other actions.

Compound Actions	
<b>If/Then</b>	If the <a href="#">test</a> evaluates to true, the 'Then' action is applied to the agent.

<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.
<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>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>

Modifying actions are applied immediately to the agent as the action is executed. Tests performed later in the action will notice property modifications enacted earlier in the action.

<b>Modifying Actions</b>	
<b>Give tokens</b>	Immediately give the specified tokens 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>Remove tokens</b>	If the agent is currently holding any of the specified tokens, immediately remove them from the agent.
<b>Assign avatar</b>	Immediately change the avatar of the agent.
<b>Reset avatar</b>	Immediately return the agent to its original or first assigned avatar. The first avatar assigned to an agent (whether through action or profile) is considered the agent's original avatar. Reset 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>Assign colour</b>	Immediately change the colour of the agent.
<b>Reset colour</b>	Immediately return the agent to its original colouring.
<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. Those tasks will still be given to the agent after the action has finished executing.

<b>Clear route 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.
----------------------------	---

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.

<b>Task Giving Actions</b>	
<b>Evacuate zone</b>	<p>The agent will navigate the scene, following the best cost route through the zone to a floor that is not a member of the specified zone. Best cost includes all standard route costing components such as horizontal distance and queuing. The task is complete once the agent has reached a floor 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.
<b>Goto best waypoint</b>	<p>The agent will attempt to reach and transition to the specified adjacent floor, portal, link, stair, ramp, path, or escalator. The waypoint must be connected to the current floor. If multiple waypoints are specified the agent will attempt to reach the 'best' or lowest cost. The agent will ignore any waypoint objects not connected to the current floor. The task is complete once the agent has transitioned to one of the waypoint objects.</p> <p>As soon as the agent has reached the waypoint it will move on to its next task. This may result in the agent turning around and going back the way they came in pursuit of the next task. To define a chain of routes to be followed in succession, use a 'List of actions' and define multiple 'Goto best waypoint' actions one after the other.</p>
<b>Seek best portal</b>	<p>The agent will navigate the scene, following the best cost route to one of the specified portals. The agent will consider all of the portals specified and select the portal with the best cost route. The task is complete once the agent has reached one of the given portals.</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>Seek origin</b>	The agent will navigate the scene, following the best cost route to the portal

	<p>through which the agent entered the simulation. The task is complete once the agent has reached its origin portal.</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>Seek process start</b>	<p>The agent will navigate the scene, following the best cost route to the specified process chain start. Best cost includes all standard route costing components such as horizontal distance and queuing. Note that only designated servers or process chain start points are possible destinations for this task (see <a href="#">Process Chains &amp; Servers</a>). If an agent is to be sent to any of a number of similar servers, those servers should be connected together by a single dispatch, and the dispatch specified in the 'Seek process start' action. In this case the dispatch will distribute agents across the connected servers.</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>
<b>Wait for duration</b>	<p>The agent will stand around for the specified number of seconds, using the wait style specified. The wait timer starts when the agent begins executing the task. The task is complete once the specified number of seconds have elapsed.</p>
<b>Wait until time</b>	<p>The agent will stand around until the simulation reaches the specified time. The task is complete once the given time has been reached. The task will be completed instantaneously if executed after the specified time.</p>

#### 4.3.2.1.1 Where to Use Actions

Actions can be applied in a number of ways. The following table lists the various locations within a project where user created custom actions can be referenced and so applied. In cases where multiple actions are applied, it is important to understand the [order of action execution](#).

Action Triggers		
Object	Property	Description

Profile	On birth	Applied to an agent newly created using the profile.
Journey, Circulate, Evacuate, Vehicle	On birth	Applied to an agent newly created by the event.
Execute Action	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	When finished	Applied to an agent once it has been processed by the server (depending on downstream availability in the process chain, this might be applied before the agent is released from the 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.
------	---------	--

#### 4.3.2.1.2 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.

Order of Action Application	
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'</li> <li>4. Timed execute 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. Timed execute 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. Timed execute 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. Timed execute action(s): 'Action'</li> <li>2. Server: 'When finished'</li> </ol>

#### 4.3.2.2 Agent Tests

Agent tests operate on a single agent at an instant in time, and return either true or false. Agent tests are used by [agent actions](#) to customize how the action is applied to an agent.

Special Tests	
<b>Always true</b>	Always returns true.
<b>Named test</b>	Returns the result of executing the specified agent test object.

Compound tests combine the results of other tests.

Compound Actions	
<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.

Static tests will always return the same result for a given agent and do not vary with time.

Static Tests	
<b>Created by</b>	Returns true if the agent was created by any of the specified events.
<b>Entered at</b>	Returns true if the agent entered the simulation at any of the specified portals.
<b>Initial Goal</b>	Returns true if the agent was created with any of the specified portals as its initial goal. The initial goal refers to the single portal destination assigned to the agent by the creating event. It does not consider any goals applied via actions, even if the action is applied by the creating event. Not all agents will have an initial goal.

<b>Uses Profile</b>	Returns true if the agent was created with any of the specified profiles.
---------------------	---

Dynamic tests can produce different results at different times. They depend on temporal values or agent properties that can change from one frame to the next.

<b>Dynamic Tests</b>	
<b>Agent Age</b>	Returns true if the agent is currently younger/older than the specified age.
<b>Has Tokens</b>	Returns true if the agent is currently holding all/any/none of the specified tokens.
<b>Insize Zone</b>	Returns true if the agent is currently on an object that is inside the zone.
<b>Random Chance</b>	Returns true if a random number chosen between 0 and 100 is less than or equal to the given value.
<b>Simulation Time</b>	Returns true if the current simulation time is before/after the specified time.

#### 4.3.2.3 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.

#### 4.3.2.4 Circulate

Circulate events create agents that will 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.

##### Properties

Arrive Tab	
<b>Timing: Start Time</b>	The reference time when the event starts. See <a href="#">Working with Time</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 alter the distribution of profiles amongst the population.
<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> Complex agent arrival is described by a series of rows. Each row contains the duration of an interval and the number of 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.</p>
<b>Origins</b>	The <a href="#">portals</a> and collections of portals at which agents will be created. Portals can be weighted to create more agents than others. See <a href="#">Choosing Objects</a> for more details.
<b>Destinations</b>	The <a href="#">portals</a> and collections of portals to which agents will travel and exit once they have finished circulating. Agent goals can be set to either travel to a portal assigned by weights (see <a href="#">Choosing Objects</a> ), or to the lowest cost (often closest) portal available.

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>	How agents should position themselves as they wait at a circulation portal.  <b>Spread Out:</b> Spread out over the floor, near the portal. <b>Stand Still:</b> Stop moving as soon as the portal is reached.
<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 bottom distribution is used.

Colours Tab	
<b>Colours</b>	<b>Event colour:</b> Agents are assigned the same colour as the event object. <b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour. <b>Darken event colour:</b> Agents are assigned a darker version of the event object colour. <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. <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. 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.

**4.3.2.5 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.

**Properties**

Agents Tab	
<b>Timing: Start Time</b>	The reference time when the event starts. See <a href="#">Working with Time</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 alter the distribution of profiles amongst the population.
<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> Complex agent arrival is described by a series of rows. Each row contains the duration of an interval and the number of 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.</p>
<b>Origins</b>	The <a href="#">portals</a> and collections of portals at which agents will be created. Portals can be weighted to create more agents than others. See <a href="#">Choosing Objects</a> for more details.

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>	How the agents will position themselves before evacuating.  <b>Stand Still:</b> The agents will stand by the portal at which they were created. <b>Spread Out:</b> The agents will spread out over the floor on which they were created.
<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.
<b>Exits</b>	The <a href="#">portals</a> and collections of portals to which agents will ultimately travel. Once agents finish evacuating zones, they will head to the lowest cost (often closest) exit portal.

Colours Tab	
<b>Colours</b>	<b>Event colour:</b> Agents are assigned the same colour as the event object. <b>Lighten event colour:</b> Agents are assigned a lighter version of the event object colour. <b>Darken event colour:</b> Agents are assigned a darker version of the event object colour. <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. <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. 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 "Exits" properties. The "Origins" collections can be weighted, changing the distribution of agents created at those portals.

#### 4.3.2.6 Execute Action

The action 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 holds a given token 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 only applied to a given agent once, even if properties of that agent change while the event is still active.

If a target token is specified then only agents holding the token receive the action.

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 entry.

Time	
<b>Time range</b>	Time over which the event is active. When the event starts, it will be applied to all targeted agents in the simulation. While active, it will apply to all targeted agents as they enter the simulation.
<b>Target: Agents holding token</b>	If set, the action will only be applied to agents holding the specified token. Note that the test for token possession is performed only when the event is applied. Agents will not spontaneously respond to the event upon acquiring the token even if the event remains active.
<b>Target: Agents in zone</b>	If set, then only agents inside the specified zone will respond to the event. When the event first fires it is applied to all agents in the zone. It is also applied to each agent that enters the zone, for as long as the event remains 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.

#### 4.3.2.7 Journey

Journey events are the most direct way to populate a scene. They create agents at one of the origin portals and instruct them to seek one of the destination portals then exit the simulation.

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.

#### Properties

Properties	
<b>Timing: Start Time</b>	The reference time when the event starts. See <a href="#">Working with Time</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 alter the distribution of profiles amongst the population.
<b>Population: Demand</b>	The number of agents the event will create and when they will be created relative to the event start time.



	<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> Complex agent arrival is described by a series of rows. Each row contains the duration of an interval and the number of 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.</p>
<b>Origins</b>	The <a href="#">portals</a> and collections of portals at which agents will be created. Portals can be weighted to create more agents than others. See <a href="#">Choosing Objects</a> for more details.
<b>Destinations</b>	The <a href="#">portals</a> and collections of portals to which agents will travel. Agent goals can be set to either travel to a portal assigned by weights (see <a href="#">Choosing Objects</a> ), or to the best (often closest) portal available.

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 evaluates to true.</p> <p><b>Specified colour:</b> Agents are assigned the specified colour.</p>

Actions Tab	
<b>On birth</b>	The action will be applied to all agents created by the event as they enter the simulation. 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 "Destination" properties. The collections can be weighted, changing the distribution of agents going from/to various portals.

#### 4.3.2.8 Open Gate

Open gate events are used to control accessibility through various [connection objects](#). [Escalators](#), [links](#), [paths](#), [ramps](#) and [stairs](#) must have gates enabled in their "Access" properties before they can be used in an open gate event.

A gated connection object will be closed by default, preventing any agents from stepping onto it, unless an open gate event is used. A gated connection object can be used in multiple open gate

events and will be open if any of the events opens it.

A [vehicle event](#) is a convenient combination of an open gate event and a [journey event](#).

**Properties**

Properties	
<b>Timing: Time range</b>	The time reference range over which the event is active. See <a href="#">working with time</a> for more information.
<b>Timing: Cycle</b>	By default, an open gate event will open its gates for the entire time it is active. It can be set to instead cycle between open and close over its active duration.  <b>Open:</b> The gate will open for this duration after becoming active (or until the event is no longer active). <b>Closed:</b> The gate will then close for this duration before opening again.
<b>Target</b>	When enabled, the gate will only open for those agents holding the specified tokens. If an agent does not hold one of the tokens, the gate is perceived as closed.
<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.

**Collections in open 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.

**4.3.2.9 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.

Events which create agents will have a "Profile" property which determines which profile will be used.

**Profile Properties**

The physical properties of the agent define radius, speed, and general movement characteristics.

Profile Properties	
<b>Radius</b>	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.  <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 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>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.
<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>Route Cost Weights</b>	<p>When determining the overall cost for a given route, each of the component costs is multiplied by a weight which serves to make that component more or less important to a particular agent. The greater the variation in weights, 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 Distance Cost:</b> Applied to the horizontal distance cost component.</p> <p><b>Vertical Distance Cost:</b> Applied to the vertical distance cost component.</p> <p><b>Queue Cost:</b> Applied to the queue cost component.</p> <p><b>Processing Cost:</b> Applied to the closed penalty cost component. This includes all penalties from from any gate or priority access 'wait when closed' penalties, delays, or controlled flow restrictions on near floor connection objects.</p>

#### 4.3.2.10 Reference 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 need be modified to reflect operational changes. See time references in [Working with Time](#).

#### Properties

Properties	
<b>Timing: Start</b>	When the reference event is considered to have started.

#### 4.3.2.11 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.

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.

General Properties	
<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.

File Types	
<b>Reference Events</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>Agent Schedules</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>Curves</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>Locations</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>Gates</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 Events</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>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>
<b>Action Events</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>

4.3.2.11.1 Timetable Files

4.3.2.11.1.1 Timetable Reference Event File

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 Name</b>	The name of a reference event. The name must be unique both within the 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 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.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token by name. The token must exist in the project. Agents created by a schedule that refers to this event will be given the specified tokens.

**Example**

See [Timetable Reference Event Example](#).

4.3.2.11.1.2 Timetable Schedule File

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 reference event, To reference event:** The start time is taken from the From reference event 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 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).

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 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, agents will be distributed across the member portals according to the distribution defined in the file. 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 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 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.
<b>Profile (optional)</b>	The name of an existing profile in the project. Agents created by the schedule will be given the specified profile. If no profile is specified, agents will be given the default profile.
<b>Init Action (optional)</b>	The name of an existing action in the project. The action will be applied to any agent created by the schedule.
<b>Give Tokens... (optional, variable)</b>	Each subsequent column can specify a single token by name. The token must exist in the project. Agents created by the schedule will be given the specified tokens.

**Example**

See [Timetable Schedule Example](#) and [Timetable Reference Event Example](#).

4.3.2.11.1.3 Timetable Curve File

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 DuratiValues..

# 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](#).

#### 4.3.2.11.1.4 Timetable Location File

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 MassMotion 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.
<b>Portals Names... (variable)</b>	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.

**Example**

```
#
# Sample MyTimetableLocation.csv
#
Group Name, Use Closest Portal_A, Portal_B, Portal_C, Portal_D

# create a group with A and B and equal entrance and exit distribution
GroupABEven, , Y, Y

# create a group with A and B where agents are 3 times more likely to use
GroupABUneven, 0.75, 0.25

# create a group with equal entrance distribution over A, B, D, where where
GroupABD, Y, Y, Y, 0, Y

# create a group with C and D where as entrances agents are twice as likely
GroupCD, Y, , , 0.6666, 0.3333
```

Also, see [Timetable Schedule Example](#).

4.3.2.11.1.5 Timetable Gate File

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. 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 MassMotion project.

Gate groups can be referenced by the [Timetable Gate 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>Gate 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.

**Example**

See [Timetable Gates Example](#)

4.3.2.11.1.6 Timetable Gate Event File

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 names of the events must be unique both within the file and across the MassMotion project. 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 or to a gate group defined in the <a href="#">Timetable Gate File</a> .

### Example

See [Timetable Gates Example](#).

#### 4.3.2.11.1.7 Timetable Action Event File

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 names of the events must be unique both within the file and across the MassMotion project. 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 by name. The token 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.

## 4.3.2.11.1.8 Timetable Evacuation Event File

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 names of the events must be unique both within the file and across the MassMotion project. 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).

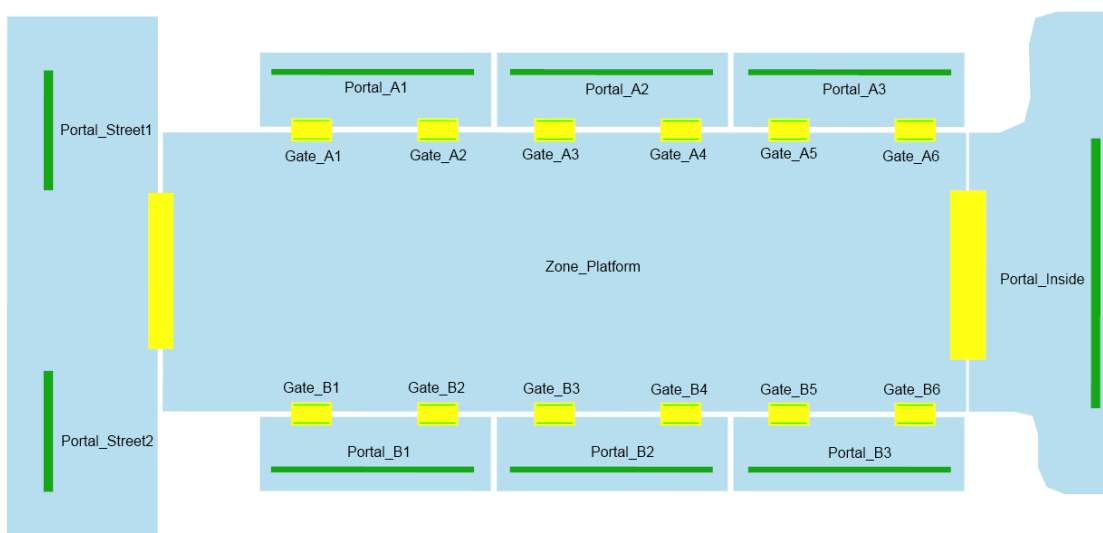
When the event fires, the evacuation command is given to all agents in the scene that meet the criteria defined by the event definition. The event then remains active for the specified duration. If a zone is specified, the active event will be applied to all agents as they enter the zone. If no zone 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.

<p><b>Location... (optional)</b></p>	<p>The name of an existing portal in the project, 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.</p>
<p><b>Give Tokens... (optional, variable)</b></p>	<p>Each subsequent column can specify a single token by name. The token must exist in the project. Each agent that is affected by the evacuation event will also be given the specified tokens.</p>

4.3.2.11.2 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 MainWindow and name it 'MyTimetable'.
3. Open the MyTimetable properties and use the File 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:

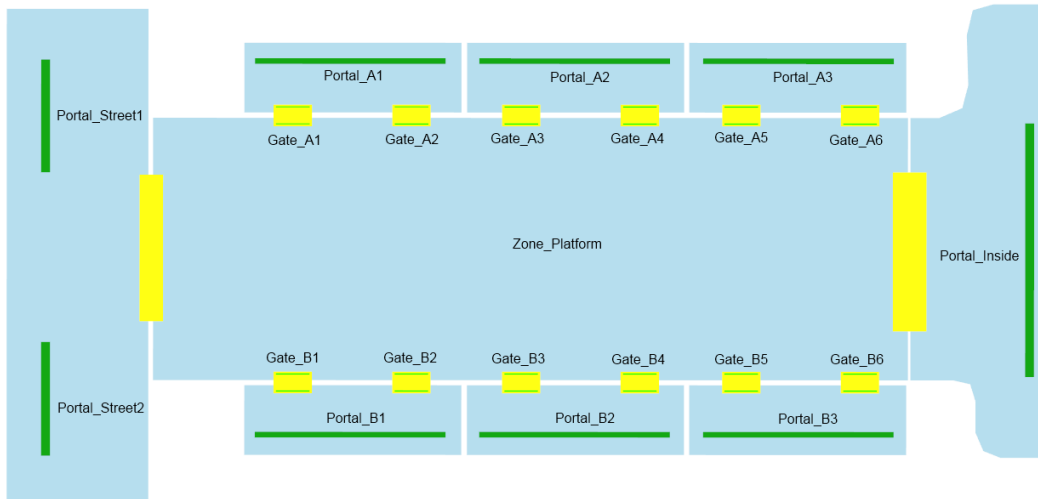
- [Timetable Schedule Example](#)
- [Timetable Gates Example](#)
- [Timetable Reference Event Example](#)

4.3.2.11.2.1 Timetable Schedule 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 secon						
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 street									
GStreet,	,	,	,	,	,	,	Y,	Y	

And finally the schedule file for generating agents

#									
#	Sample MyTimetableSchedule.csv								
#									
From,	To,	Populat	Time Off	Curve,	Avatar or C	Profi	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	CConstar	Red,	,	,	TNextDay	Ttransfe

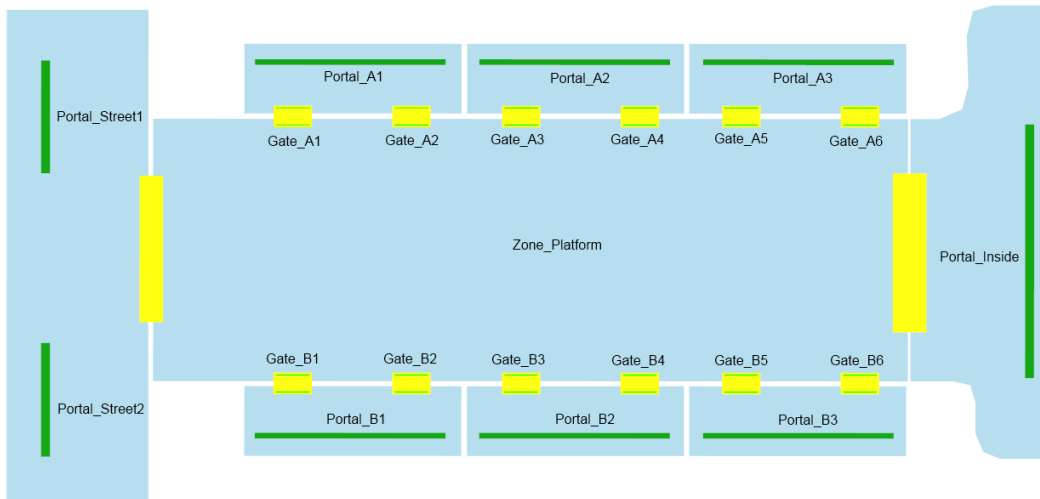
4.3.2.11.2.2 Timetable Gates Example

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,

And finally the gate event file for opening and closing gates.

#													
#	Sample MyTimetableGateEvent.csv												
#													
Reference	Time Offset,	Duration,	Apply On	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												

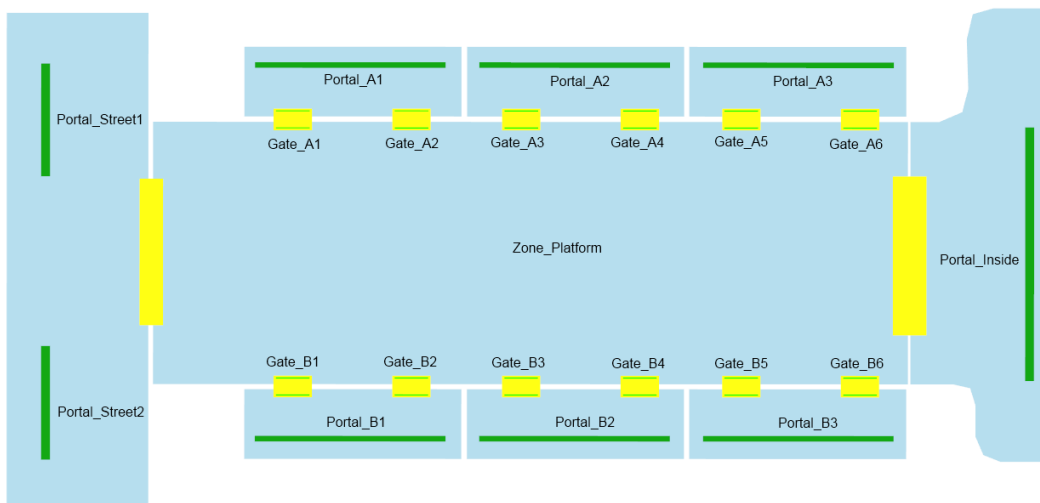
,	32:30:00,	01:00:00,	,	,	Gate_B4		
---	-----------	-----------	---	---	---------	--	--

4.3.2.11.2.3 Timetable Reference Event Example

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_	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_	GTrackA,	100	14:15:00	CConstar	Red				
GStreet,	GTrackB,	200,	14:15:00	CConstar	Blue				
Portal_	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_A	Festival,	100,	-00:45:	CConstar	Yellow				
GStreet,	Festival,	57,	-00:45:	CConstar	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 Name	Gate_A1,	Gate_A2,	Gate_A3,	Gate_A4,	Gate_A5,	Gate_A6,	Gate_A7,	Gate_A8,	Gate_A9,	Gate_A10,	Gate_A11,	Gate_A12,	Gate_A13,
# Combine all gates for track A into a single group for easy refer													
Gates_A,	Y,	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		

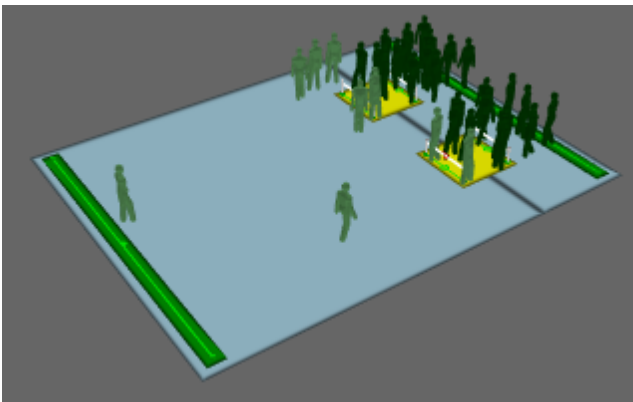
#### 4.3.2.12 Token

Tokens are objects held by agents. They can be used to identify certain agents or to limit access at [connection objects](#). Tokens are given or removed using [Agent Actions](#).

#### 4.3.2.13 Vehicle

Vehicle events are used to simulate the alighting and boarding of passengers from/to vehicles at regular intervals, such as at a subway station. Gated links can be used to simulate the opening of vehicle doors.

Vehicles "arrive" at controllable intervals and open their doors (ie. gated links) for as long as they "dwell". Alighting agents will be created automatically slightly before the doors open, but boarding agents can have more finely tuned arrival times. Origin and destinations of boarding and alighting agents do not need to be the same, or have the same weights.



**Darker agents alighting from a vehicle event with lighter agents boarding.**

#### 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.

Gates do not necessarily need to be near either the origins or destinations of agents, nor do they need to separate their origins and destinations. Additional gates can be included, allowing the vehicle event to control the flow of agents beyond those boarding and alighting.

#### Properties

Vehicle Tab	
<b>Arrivals: First Arrival</b>	When the first vehicle "arrives". If a vehicle arrives too close to the simulation start, no alighting agents will be created.
<b>Arrivals: Repeat</b>	How many times a vehicle will arrive.  <b>Until End:</b> An unlimited number of vehicles will arrive. <b>Specified Count:</b> A specified number of vehicles will arrive
<b>Timing: Headway</b>	The time between vehicle arrivals. This can be specified as a <a href="#">distribution</a> .
<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>Vehicle 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.

Alighting Tab	
<b>Create alighting agents</b>	Enable the creation of alighting agents.
<b>Population: Agent count</b>	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.
<b>Population: 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>Vehicle Origins</b>	The <a href="#">portals</a> and collections of portals at which alighting agents will be created. Portals can be weighted to create more agents than others. See <a href="#">Choosing Objects</a> for more details.
<b>Vehicle Destinations</b>	The <a href="#">portals</a> and collections of portals to which alighting agents will travel. Agent goals can be set to either travel to a portal assigned by weights, or to the nearest portal available.  If using "Assigned" agent goals, the portals can be weighted to be a destination for more agents. See <a href="#">Choosing Objects</a> for more details.

Boarding Tab	
<b>Create boarding agents</b>	Enable the creation of boarding agents.
<b>Timing: Before</b>	The time before vehicle arrival that boarding agents start being created.

<b>arrival</b>	
<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: 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 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="#">Agent Start Distributions</a> for information on arrival distributions.</p> <p><b>Table:</b> Complex agent arrival is described by a series of rows. Each row contains the duration of an interval and the number of 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.</p>
<b>Vehicle Origins</b>	The <a href="#">portals</a> and collections of portals at which boarding agents will be created. Portals can be weighted to create more agents than others. See <a href="#">Choosing Objects</a> for more details.
<b>Vehicle Destinations</b>	<p>The <a href="#">portals</a> and collections of portals to which boarding agents will travel. Agent goals can be set to either travel to a portal assigned by weights, or to the nearest portal available.</p> <p>If using "Assigned" agent goals, the portals can be weighted to be a destination for more agents. See <a href="#">Choosing Objects</a> for more details.</p>

#### Colours Tab

<b>Alighting / Boarding: Colours</b>	<p>Defines the colour scheme for alighting /boarding 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>Alighting / Boarding: On birth</b>	The action will be applied to all boarding/alighting agents created by the event as they enter the simulation. For a description of the order in which actions are applied see <a href="#">Action Order of Execution</a> .
---------------------------------------	--

**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.

They can also be used in "Origins" and "Destination" properties for alighting and boarding agents. The collections can be weighted, changing the distribution of agents going from/to various portals.

**4.3.3 Analysis Objects**

Analysis Object Type	Description
<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>	A reusable analysis object. These allow queries to target agents meeting specific criteria, including location, density, etc.
<a href="#">Cordon</a>	A reusable analysis object. These objects represent boundaries in space that agents can cross. Despite appearing in the scene, they do not impact simulations.
<a href="#">Region</a>	A reusable analysis object. These objects represent volumes in space that may contain agents. Despite appearing in the scene, they do not impact simulations.
<a href="#">Trip</a>	A reusable analysis object. These objects represent an ordered series of locations an agent may traverse over its lifetime.
<a href="#">Graphs</a>	A query analysis object that produces graphs.  <a href="#">Agent Density</a> : Counts the number of agents at various levels of crowd density. <a href="#">Agent Speed Ratio</a> : Counts the number of agents with different ratios of actual to desired speed.. <a href="#">Composite</a> : Fully customizable graph which can combine any number of simulation runs and values. <a href="#">Flow Count</a> : Counts the number of agents crossing various <a href="#">transitions</a> . <a href="#">Performance Graph</a> : Compares performance of <a href="#">simulation runs</a> . <a href="#">Population Count</a> : Counts the number of agents in different <a href="#">areas</a> . <a href="#">Region Density</a> : Measures the average density over one or more regions. <a href="#">Server Population Count</a> : Counts the number of agents at various servers.
<a href="#">Maps</a>	A query analysis object. These queries paint objects in the scene based on the behaviour of agents. With the exception of the Vision Time Map, all maps may only be applied to <a href="#">walkable objects</a> .  <a href="#">Agent Count</a> : Counts the number of unique agents that cross each point. <a href="#">Agent Path</a> : Traces the route taken by each agent.



	<p><b><a href="#">Agent Time to Exit</a></b>: The time required for an agent to exit from each point.</p> <p><b><a href="#">Average Density</a></b>: The average density at each point.</p> <p><b><a href="#">Average Non-Zero Density</a></b>: The average non-zero density at each point.</p> <p><b><a href="#">Dynamic Path</a></b>: The average non-zero density at each point.</p> <p><b><a href="#">Experienced Density</a></b>: The experienced density at each point.</p> <p><b><a href="#">Instantaneous Density</a></b>: The current density at each point. This map is "live" and updates with simulation playback.</p> <p><b><a href="#">Maximum Density</a></b>: The maximum density at each point.</p> <p><b><a href="#">Static Cost</a></b>: The cost perceived by agents to reach various destination portals. This map does not require a simulation run.</p> <p><b><a href="#">Static Distance</a></b>: The measured distance to various destination portals. This map does not require a simulation run.</p> <p><b><a href="#">Time Above Density</a></b>: The time agents spend above a specified density at each point.</p> <p><b><a href="#">Time Occupied</a></b>: The time for which each point is occupied by an agent.</p> <p><b><a href="#">Time Until Clear</a></b>: The time before the last agent leaves each point.</p> <p><b><a href="#">Vision Time</a></b>: The time agents spend looking at a surface. This map can be applied to barriers.</p>
<b><a href="#">Tables</a></b>	<p>A query analysis object that produces tables.</p> <p><b><a href="#">Agent Area Time</a></b>: Lists the amount of time agents spend in given areas.</p> <p><b><a href="#">Agent LOS Time</a></b>: Lists the amount of time agents spend at each level of service.</p> <p><b><a href="#">Agent Process Chain Time</a></b>: Lists the amount of time agents spend in a process chain.</p> <p><b><a href="#">Agent Summary</a></b>: Provides general information about each agent.</p> <p><b><a href="#">Agent Timetable Summary</a></b>: Provides a summary of agents produced by a given timetable</p> <p><b><a href="#">Agent Token Time</a></b>: Lists the amount of time agents carry given tokens.</p> <p><b><a href="#">Agent Trip Time</a></b>: Lists the time required for agents to complete a given trip.</p> <p><b><a href="#">Origin/Destination</a></b>: Counts the number of agents entering and exiting at each portal</p> <p><b><a href="#">Performance Table</a></b>: Provides diagnostics for simulation runs</p> <p><b><a href="#">Server Summary</a></b>: Lists queue times and other information for servers.</p>

#### 4.3.3.1 Simulation Run

A simulation run represents a single iteration of a MassMotion 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](#), with results written to the specified database file (see [Simulation Data](#)). An empty simulation run can also be created from the main window's analysis ribbon. An empty simulation run can be used to connect to an existing database file by editing the simulation run properties. This is useful when comparing results from different projects; a simulation run object can be created in one project that is then set to point at the results generated from a different project so that they can be compared.

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 by using the run's agent filter). However, only one simulation may be executed at once. During simulation any other simulation runs which point to the same database will be prevented from reading the database until the simulation is complete. Similarly, when using the [movie and image export tools](#), a lock is placed on the simulation runs used and they may not be used for simulation until the export window is closed.

### 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.

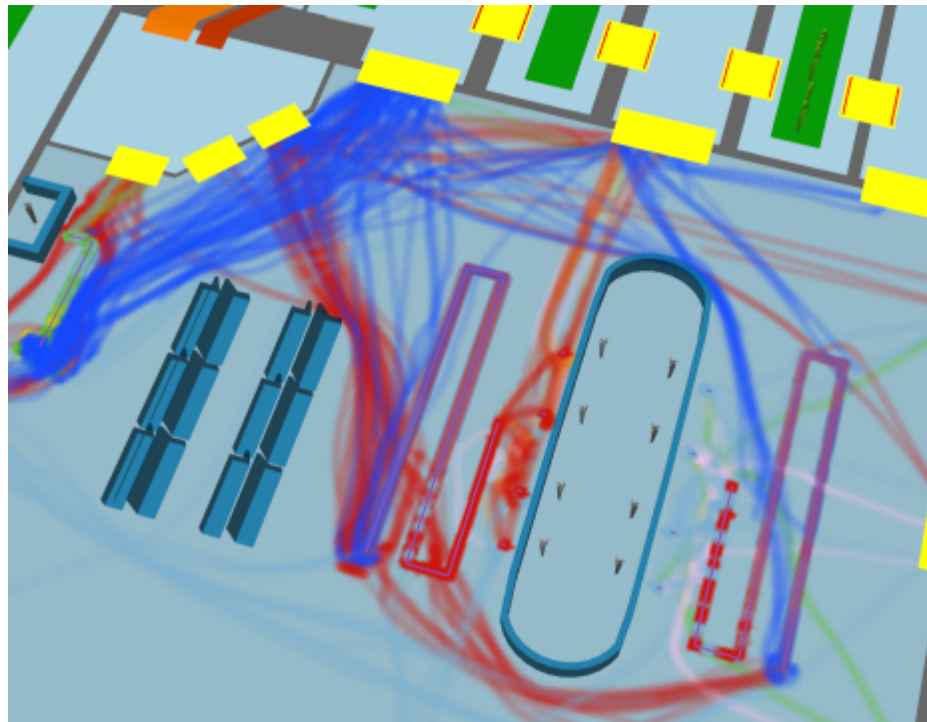
Properties	
<b>Database File</b>	The file to which the object is connected. Simulation execution writes to this file while playback and analysis read from this file.
<b>Status</b>	The status of the simulation run's connection to the database. Users can refresh the connection by pressing the refresh button.
<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>	<p>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> <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>

### 4.3.3.2 Agent Filter

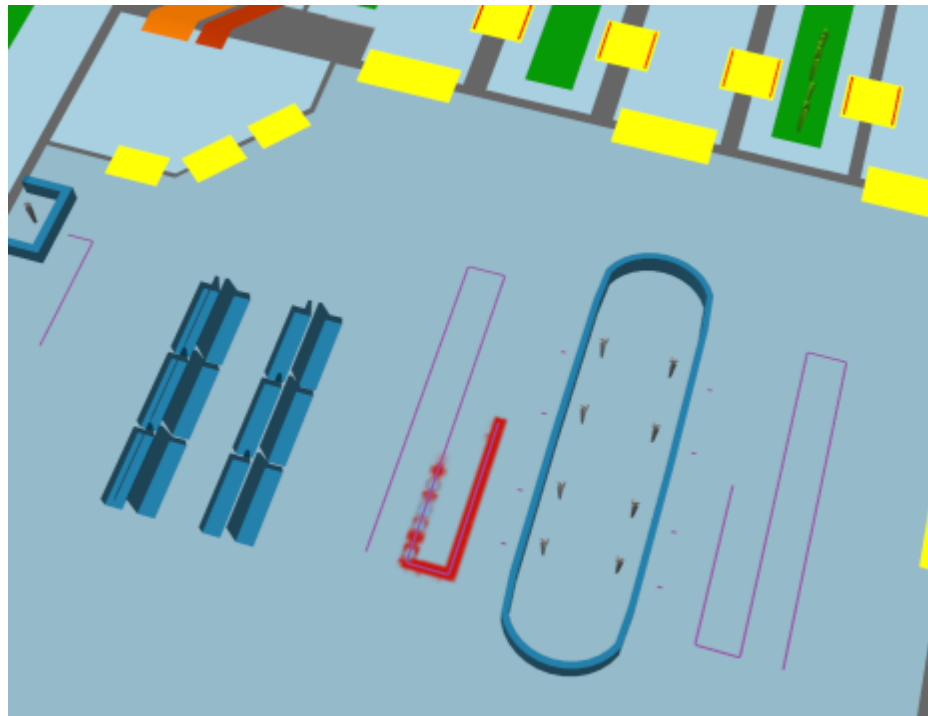
Agent filters provide a powerful way to set up queries (graphs, tables, maps) that operate on a specific (possibly time-varying) set of agents. Refer to the various [Graph](#), [Table](#) and [Map](#) types for descriptions of how filters can be used in different types of queries.

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.

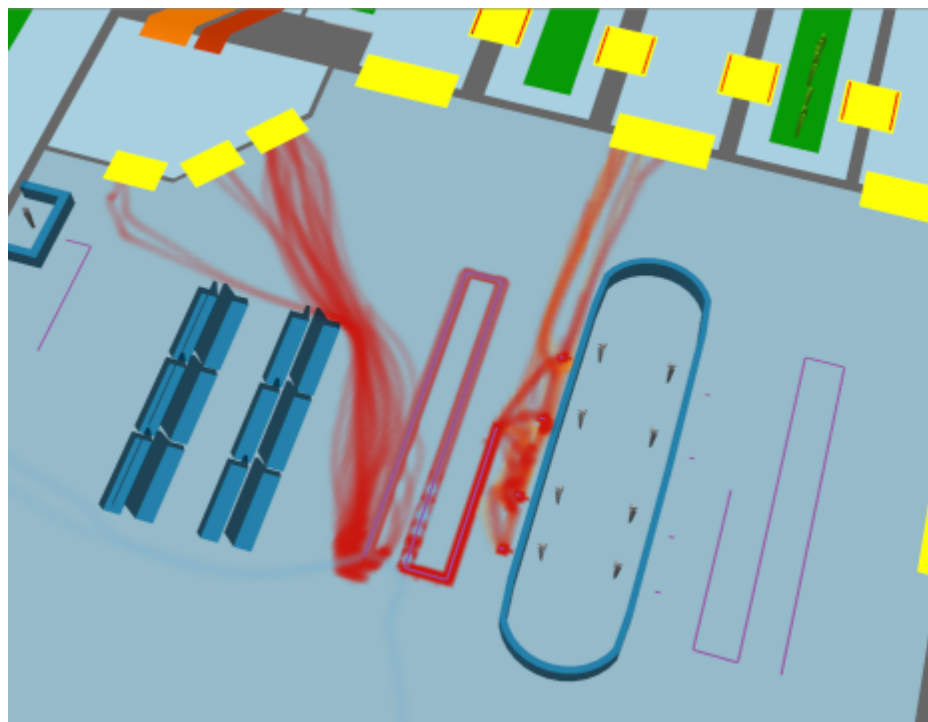
One important type of transforming filter is an 'are ever' filter. This filter can be thought of as a qualifier on another filter; instead of agents only being included in the generated list **while** they satisfy some criterion (and therefore potentially dropping in and out of the generated list over 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' &gt; 'At server'

### Filter Debugging

An excellent way to determine if a filter is working as expected is to set a particular simulation to colour agents differently based on whether or not they satisfy the filter (see [Simulation Run](#) for details). For instance, a simulation run can be set to colour all agents blue unless they currently satisfy a given filter, in which case they should be coloured green. By then using the various playback controls in the [time panel](#) (such as dragging the timeline slider), it is easy to see which agents satisfy the filter at which times.

### Named Filters

Agent filters can be created inline within a query, for use within that query. Filters can also be created as independent, named objects by clicking on the 'Agent Filter' button in the Analysis toolbar; these can then be referred to within other filters and queries, as discussed below.

### 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.

### Collections in agent filters

[Collections](#) can be used to define agent filters and allow for the creation of more complex filters. It is also more convenient to use a single "in area" filter with a collection, as opposed to an "any of filter" followed by several individual "in area" filters.

Filter Type	References	Agents Included
<b>All agents</b>	None	All agents.
<b>Selected agents</b>	None	Agents that are currently selected. Note that this filter is 'live', and refers that whatever agents are selected when the filter is actually used, not when it is created. For instance, if an <a href="#">Agent Path</a> map was created using this filter, whatever agents were selected when the 'Evaluate' button was pressed would have their paths shown. If later on a different set of agents was selected and the map was re-evaluated, the map would 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.

Filter Type	References	Agents Included
<b>None of:</b>	One or more filters	Agents included by none of the referenced filters at a given instant.
<b>Created by:</b>	Timetable or schedule	Agents that were initially created by the referenced timetable or schedule.
<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 are included.
<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 are included.
<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 are included.
<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 collection are included.
<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 are included.
<b>In area:</b>	<a href="#">Area</a>	Agents currently in the given area. With collections: Agents currently in any of the areas in the collection are included.
<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.

#### 4.3.3.3 Cordon

Analysis cordon objects are used when performing analysis but have no impact on the simulation itself. Agents will ignore objects of this type. It is possible to create and use cordon objects for analysis on previously generated simulation runs without having to re-execute the run.

##### 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).
- During analysis, agents will be considered 'at' the cordon when the centre point between their feet crosses any surface of the cordon object.

##### Properties

There are no properties for objects of this type.

#### 4.3.3.4 Region

Analysis region objects are used when performing analysis but have no impact on the simulation itself. Agents will ignore objects of this type. It is possible to create and use region objects for analysis on previously generated simulation runs without having to re-execute the run.

Agents are considered inside a region when their feet are inside the region.

##### Notes on geometry

- Must be a single polygon mesh object or a collection of polygon mesh objects.
- During analysis, agents will be considered 'in' the region 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.

#### 4.3.3.5 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 complete a trip if they 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'). Trips created using the toolbar button can later be referenced by queries or filters.

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.

End when	
<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 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 analysis 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.

**4.3.3.6 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 as an image or a CSV file of the plotted values. The following image formats are supported: \*.png, \*.svg, \*.pdf

**Graph Styles**

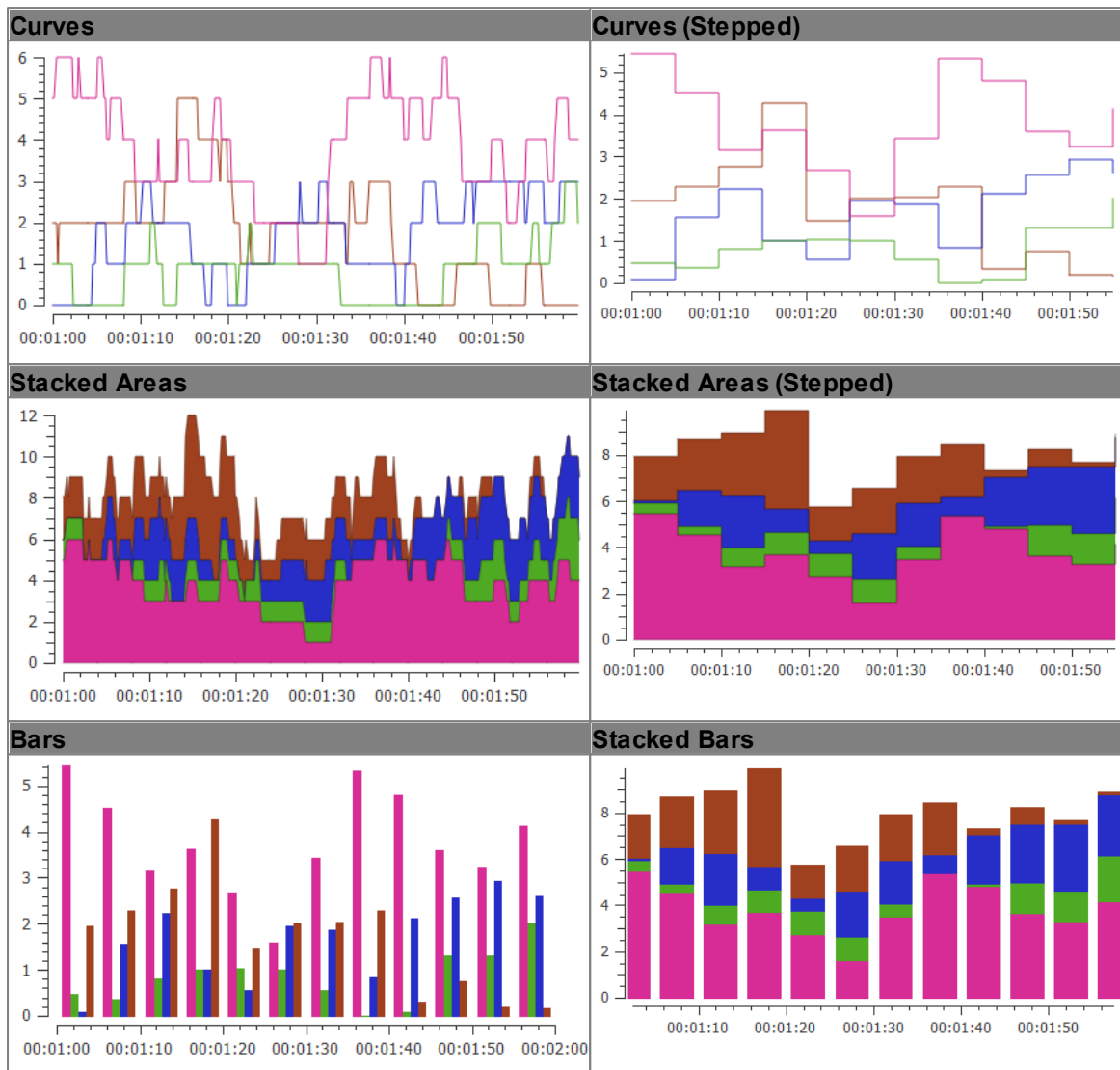
The title and axis labels of graphs can be set for each graph, as well as the inclusion of a legend. The colour of datasets is set by the groups in the graph structure (see below).

Several styles of graph are available, though not all types of graphs may support each one. Stepped graphs and bar graphs require a "bucket size" which is the duration which each section of the graph lasts.

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.



**Navigating Graphs**

Once a graph is generated, subsections of the graph can be examined in further detail by clicking and dragging to define an area to zoom to in the graph. The view can be reset by middle clicking the graph.

**Graph Structure**

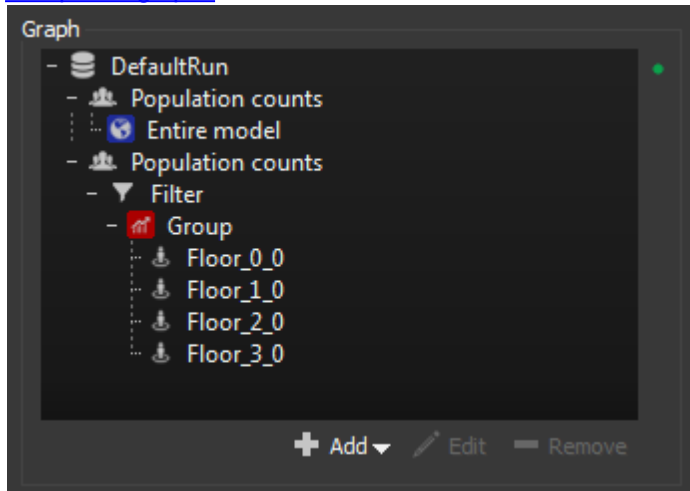
Graphs use a tree structure to organize their source data into datasets. Multiple items can be added at each level; for instance, multiple groups can be placed under a filter.

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.

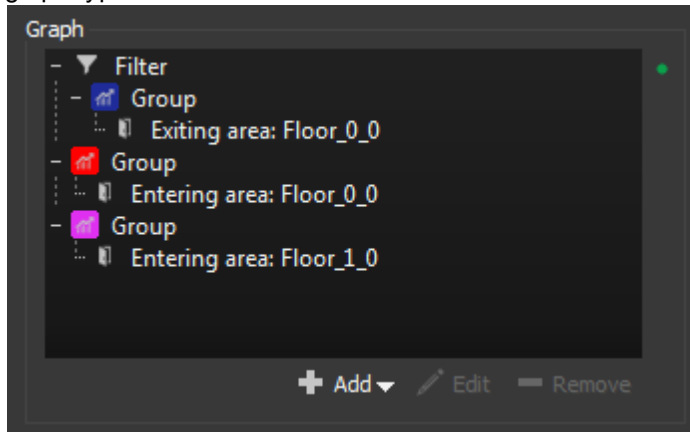
<b>2. Graph Type</b>	The type of data to graph. Composite graphs can combine different <a href="#">Flow Counts</a> , <a href="#">Population Counts</a> or <a href="#">Server Population Counts</a> .
<b>3. Filter (optional)</b>	An <a href="#">agent filter</a> that limits which agents are counted.
<b>4. 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="#">Agent Density</a>	Counts the number of agents at various levels of crowd density.
<a href="#">Agent Speed</a>	Counts the number of agents with different ratios of actual to desired speed.

<a href="#">Ratio</a>	
<a href="#">Composite</a>	Fully customizable graph which can combine any number of simulation runs and values.
<a href="#">Flow Count</a>	Counts the number of agents crossing various <a href="#">transitions</a> .
<a href="#">Performance Graph</a>	Compares performance of <a href="#">simulation runs</a> .
<a href="#">Population Count</a>	Counts the number of agents in different <a href="#">areas</a> .
<a href="#">Region Density</a>	Computes the average density over time for one or more <a href="#">regions</a> .
<a href="#">Server Population Count</a>	Counts the number of agents at various servers.

#### 4.3.3.6.1 Agent Density

An agent density graph is a special graph type that shows the breakdown of agents within different density ranges (see [LOS Colour Mapping](#)). The graph will be shown as stacked areas, with each band corresponding to one density range.

Agent Density Parameters	
<b>Name</b>	The name of the graph.
<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 (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 included in the graph. This can be used

	to display density ranges only for agents in a given area, only for those undergoing a particular trip, only those holding a specific token, etc.
<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.

#### 4.3.3.6.2 Agent Speed Ratio

An agent speed ratio graph is a special graph type that shows the breakdown of agents within different ranges of speed ratios (ratio of actual to desired speed). The graph will be shown as stacked areas, with each band corresponding to one speed ratio range.

<b>Agent Density Parameters</b>	
<b>Name</b>	The name of the graph.
<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 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 included in the graph. This can be used to display speed ratio ranges only for agents in a given area, only for those undergoing a particular trip, only those holding a specific token, etc.
<b>Speed Ratio Ranges</b>	Specifies the speed ratio ranges and corresponding colours that should be used to build the graph. Defaults to ratios based on <a href="#">standard Fruin walking speeds</a> .
<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.
--	--

#### 4.3.3.6.3 Composite

Composite graphs are an advanced feature that 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 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 ([population count](#), [flow 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>Name</b>	The name of the graph.
<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>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>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>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.

4.3.3.6.4 Flow Count

Flow count graphs can be created to measure the number of agents performing specified [transitions](#) during particular time intervals (bins).

Groups in a flow count graph are composed of one or more [transitions](#). Each graph can be configured to produce multiple series showing the flow counts for each of the selected transitions independently, or one series showing the flow count treating all selected transitions as a single combined transition.

If a group of transitions is created under a filter, only agents satisfied by that filter will be included in the associated counts. 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 in the flow count as long as they ever held the specified token at some point during the reporting period.

Flow Count Parameters	
<b>Name</b>	The name of the graph.
<b><a href="#">Simulation run</a></b>	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.</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.
--	--

#### 4.3.3.6.5 Performance

Performance graphs can be used to compare time and operations required to process simulation steps for each simulation frame.

Performance Table Parameters	
<b>Name</b>	The name of the table.
<b><a href="#">Simulation runs</a></b>	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. <b>Operations:</b> Operations required. <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.  <ul style="list-style-type: none"> <li>• <b>Entire Frame:</b> Processing an entire frame.</li> <li>• <b>Serial Components:</b> Processing the serial components of a frame.</li> <li>• <b>Threaded Components:</b> Processing the threaded components of a frame.</li> </ul> Information from individual components can be displayed as well.
<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.

#### 4.3.3.6.6 Population Count

Population count graphs can be created to measure the total number of agents over time in specified areas.

Groups in a population count graph are composed of one or more [areas](#). Each group can be

configured to produce multiple series showing the population counts in each of the selected areas independently, or one series showing the population count treating all selected areas as a single combined area.

If a group of areas is created under a filter, only agents satisfied by that filter will be included in the associated counts. 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 population count only while they are holding the specified token. In the second case, agents will be included in the population count for the entire reporting period as long as they ever held the specified token at some point during that period.

Population Count Parameters	
<b>Name</b>	The name of the graph.
<b><a href="#">Simulation run</a></b>	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 is available. If selected, population 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 or explanation about the graph. This will be saved, but will not be exported to CSV.



## 4.3.3.6.7 Region Density

Region density graphs can be used to show how the average density (total population divided by total area) varies over time within one or more [Region](#) objects. Each region 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.

Region Density Parameters	
<b>Name</b>	The name of the graph.
<b><a href="#">Simulation run</a></b>	The simulation run for which densities should be calculated.
<b>Reporting period</b>	The time period over which densities should be calculated.
<b>Sampling period</b>	How frequently to measure each region'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>Regions</b>	Which regions to include in the graph. Each region will correspond to one graph series.
<b>Density Ranges</b>	Which colours and density ranges to use for the coloured background stripes of 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.

## 4.3.3.6.8 Server Population Count

Server population count graphs can be created to measure the total number of agents at specified 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.

Groups in a server population count graph are composed of one or more servers. Each group can be configured to produce multiple series showing the population counts for each of the selected servers independently, or one series showing the total population count for all selected servers.

If a group of servers is created under a filter, only agents satisfied by that filter will be included in the associated counts. 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 server population count only while they are holding the specified token. In the second case, agents will be included in the count for the entire reporting period as long as they ever held the specified token at some point during that period.

Server Population Count Parameters	
<b>Name</b>	The name of the graph.
<b><a href="#">Simulation run</a></b>	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> <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.</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.

### 4.3.3.7 Maps

Map queries paint objects in the scene based on the behaviour of agents.

All maps have a "Surface Objects" parameter which determines which objects will be painted. Most maps can only paint walkable surfaces, with the exception of [vision time maps](#), which can paint barriers as well.

Several maps can customize how they display values on their surfaces. Using transparent values will allow an object's colour to be displayed instead of other colours. See [Working with Colours](#) for more information.

Only one map can be displayed at any time, evaluating or showing a map will hide any other map that is shown. Maps can be hidden from the view menu or by right clicking map objects in the list view.

Maps cannot be directly exported, but the [movie and image exporter](#) can be used instead.

#### List of Maps

Map Type	Description
<a href="#">Agent Count</a>	Counts the number of unique agents that cross each point.
<a href="#">Agent Path</a>	Traces the route taken by each agent.
<a href="#">Agent Time to Exit</a>	The time required for an agent to exit from each point.
<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="#">Dynamic Path</a>	Shows dynamic 'trails' behind each agent as they move around the model.
<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="#">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.
<a href="#">Time Above Density</a>	The time agents spend above a specified density at each point.

<a href="#">Time Occupied</a>	The time for which each point is occupied by an agent.
<a href="#">Time Until Clear</a>	The time before the last agent leaves each point.
<a href="#">Vision Time</a>	The time agents spend looking at a surface. This map can paint barriers.

4.3.3.7.1 Agent Count

Agent count maps can be used to display how many different agents use different parts of a walkable object. The colour at each point will indicate the number of unique agents that ever stood at that point during the given time range.

Agent Count Map Parameters	
<b>Name</b>	The name of the map.
<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>	A simple field that can be used to save comments or explanation about the map.

4.3.3.7.2 Agent Path

Agent path maps can be used to 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 each other, 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	
<b>Name</b>	The name of the map.
<a href="#">Simulation run</a>	The simulation run for which the map should be generated. This also controls the colour of the paths; the agent colouring options set on the simulation run are used when generating the paths.
<b>Time range</b>	The time period over which paths should be generated.
<a href="#">Agent Filter</a>	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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.3 Agent Time To Exit

Agent time to exit maps can be used to measure how long it takes agents to exit the simulation from any point in their journey through the simulation. The colour at each point will indicate the longest time it took any agent to exit the simulation after standing at that point. For instance, in an evacuation scenario, points far away from the exits would have a large value since agents standing at those points would take a long time to be able to exit the simulation. Points near the exits would have small values since agents at those points would exit the simulation shortly after the evacuation begins.

Agent Time To Exit Map Parameters	
<b>Name</b>	The name of the map.
<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 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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.4 Average Density

Average density maps can be used to display what parts of an object were, on average, most crowded. The colour at each point will indicate the average density (agents per square metre) over the given time range in a circle about that point. The circles used to calculate density have a standard Fruin area of 3.25 square metres.

The average density is defined as:

$$LOS(t) = \frac{\sum_{n=1}^t density(n)}{t}$$

Average Density Map Parameters	
<b>Name</b>	The name of the map.

<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 floor 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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.5 Average Non-Zero Density

Average non-zero density maps are the same as [Average Density](#) maps except that zero density values are not included in the average. Therefore, long stretches of time with no agents will not affect the computed values.

Average Non-Zero Density Map Parameters	
<b>Name</b>	The name of the map.
<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 floor 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>	A simple field that can be used to save comments or explanation about the map.

## 4.3.3.7.6 Dynamic Path

Dynamic path maps are similar to [agent path maps](#), but the trails left behind agents fade out 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 by using the 'play' option of the [Time Panel](#). Manually dragging the playback slider will erase and reset all paths.

Dynamic Path Map Parameters	
<b>Name</b>	The name of the map.
<a href="#">Simulation run</a>	The simulation run for which the map should be generated. This also controls the colour of the paths; the agent colouring options set on the simulation run are used when generating the paths.
<a href="#">Agent Filter</a>	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 be, in seconds. For instance, setting this to 10 seconds will mean that the trail behind each agent will show approximately the last 10 seconds of movement before fading into invisibility. 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>	A simple field that can be used to save comments or explanation about the map.

## 4.3.3.7.7 Experienced Density

Experienced density maps are a measure of the average density experienced by the agents (the average of all experiences), 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.

#### Experienced Density Map Parameters

<b>Name</b>	The name of the map.
<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 floor 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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.8 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 indicates the current density (agents per square metre) in a circle about that point. The circles used to calculate density have a standard Fruin area of 3.25 square metres.

Average Density Map Parameters	
<b>Name</b>	The name of the map.
<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 floor 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>	A simple field that can be used to save comments or explanation about the map.



## 4.3.3.7.9 Maximum Density

Maximum density maps are similar to [average density maps](#), but show the maximum density ever reached at each point during the given time range.

Maximum Density Map Parameters	
<b>Name</b>	The name of the map.
<b><a href="#">Simulation run</a></b>	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 floor 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>	A simple field that can be used to save comments or explanation about the map.

## 4.3.3.7.10 Static Cost

Static Cost Maps show how agents perceive the network and the cost in seconds to arrive at given portals from each point. This map does not take crowds into account and can be evaluated before a simulation is run. **Evaluating this map rebuilds the scene network which can be time consuming and the evaluate command cannot be canceled.**

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. Distance penalties and link directionality will have an impact on the final result.

Static Cost Map Parameters	
<b>Name</b>	The name of the map.
<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 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, reflecting what occurs in simulation. A subset of the model can be used to speed up processing time or to focus the analysis.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the map.

4.3.3.7.11 Static Distance

Static Distance Maps show the distance from given portals to each point. This map does not take crowds into account and can be evaluated before a simulation is run. **Evaluating this map rebuilds the scene network which can be time consuming and the evaluate command cannot be canceled.**

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. For example, any height variation on a floor is ignored. Distance penalties and link directionality will have **no** impact on the final result.

Static Distance Map Parameters	
<b>Name</b>	The name of the map.
<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 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, reflecting what occurs in simulation. A subset of the model can be used to speed up processing time or to focus the analysis.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the map.

4.3.3.7.12 Time Above Density

Time above density maps can be used to find parts of walkable objects where the agent density exceeded a given threshold, and determine for how long this occurred. Map colours will indicate the amount of time each point on each selected object had a local density greater than a given threshold.

Time Above Density Map Parameters	
<b>Name</b>	The name of the map.
<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</b>	The density value cutoff to use. Standard Fruin walkway values can be selected,

<b>Density</b>	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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.13 Time Occupied

Time occupied maps can be used to measure how long a given location was occupied by agents during the simulation. The colour at each point will indicate the cumulative count (in seconds) that any agent stood at that point. 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	
<b>Name</b>	The name of the map.
<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.
<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>	A simple field that can be used to save comments or explanation about the map.

#### 4.3.3.7.14 Time Until Clear

Time until clear maps can be used to measure how long it took until no agents were left at various points. 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. For instance, in an evacuation scenario, points far away from the exits would have small values since those areas would be left 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	
<b>Name</b>	The name of the map.
<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

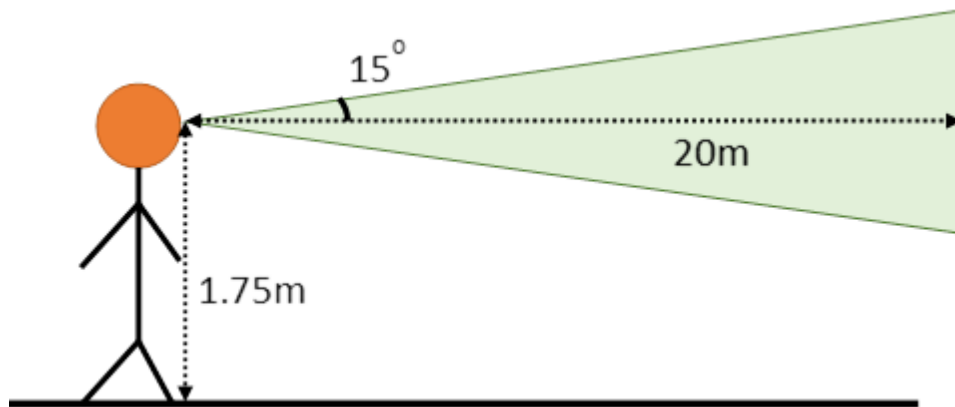
	computed from the start of the time range.
<b>Agent Filter</b>	Used to select a subset of agents to consider when generating the map.
<b>Colouring</b>	The colours that will be used in the map. Ranges are given as times in seconds. See <a href="#">Working with Colours</a> for more information.
<b>Surface Objects</b>	Which objects to apply the map to.
<b>Notes</b>	A simple field that can be used to save comments or explanation about the map.

4.3.3.7.15 Vision Time

Vision time maps are an **experimental** feature which calculates where agents are looking as they move through a simulation. Unlike other maps, vision time maps can also be applied to barriers.

Vision time maps display the amount of time a given point on an object is viewed. The time is multiplied by the number of agents, thus two agents looking at a wall for 1 second and one agent looking at a wall for 2 seconds will both yield a time value of 2 seconds.

The scene is divided into volumetric elements or "voxels". All agents will project a viewing cone ahead of them marking voxels they see. This cone is currently fixed at a 15 degree half angle, 20 metre length and 1.75m above the floor.



Agents will mark any voxels containing scene objects, including hidden objects and reference geometry and are unable to see through them. They will, however, see through any other agents in front of them and analysis [cordons](#) and [regions](#).

The voxel based approach can cause certain visual artefacts may occur as voxels obscure each other.

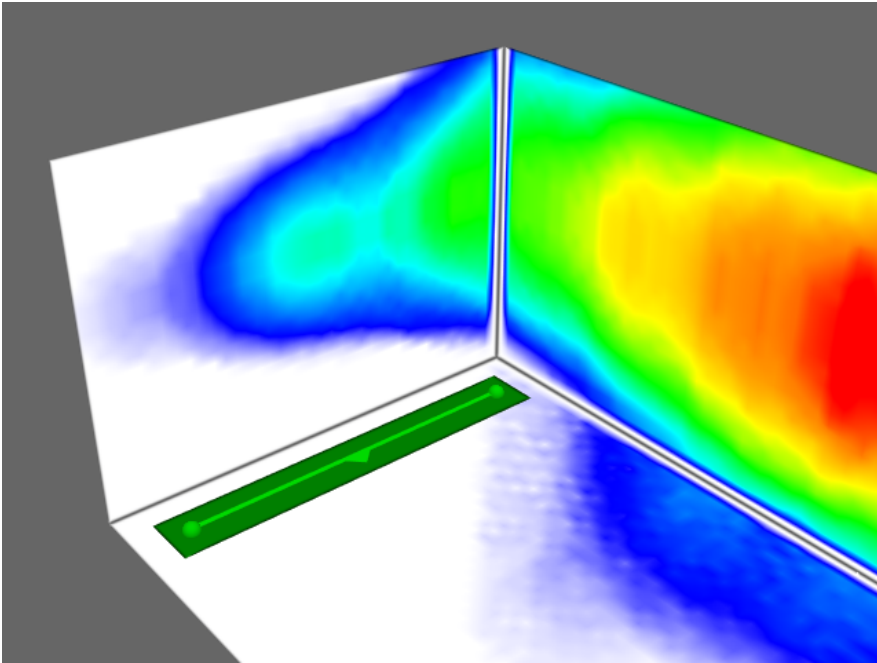


Figure 1: Voxels in corners have no set values because they are blocked by adjacent voxels.

This map is very resource intensive, consuming both CPU time and memory. Changing the sampling period is an effective way of speeding up the map computation; the amount of memory consumed can only be reduced by choosing a smaller set of objects.

Average Density Map Parameters	
<b>Name</b>	The name of the map.
<b><a href="#">Simulation run</a></b>	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. 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 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>Notes</b>	A simple field that can be used to save comments or explanation about the map.

**4.3.3.8 Tables**

Table queries all produce tabular data in different forms. Once evaluated, tables can be exported to a CSV file. Additionally, right-clicking on any table column allows the generation of a histogram of the values in that column, which can itself be exported to CSV.

**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 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 Trip Time</a>	Lists the time required for agents to complete a given trip.
<a href="#">Origin/Destination</a>	Counts the number of agents entering and exiting at each portal.
<a href="#">Performance Table</a>	Provides diagnostics for simulation runs.
<a href="#">Server Summary</a>	Lists queue times and other information for servers.

4.3.3.8.1 Agent Area Time

Agent area time tables can be used to determine how long different agents spend in different [areas](#).

Agent Region Time Parameters	
<b>Name</b>	The name of the table.
<a href="#">Simulation run</a>	The simulation run for which region times should be calculated.
<b>Time range</b>	The time period over which times should be calculated.

<b>Agent Filter</b>	Used to select when agents should be included in the area times; see the column descriptions below for details.
<b>Areas</b>	Which areas should 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, and are ever in any of the given areas during that time range.

Agent Region Time Columns	
<b>Agent ID</b>	Internal agent ID.
<b>Names (variable)</b>	One column per selected <a href="#">areas</a> , showing how long the agent spent in that area 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 in various areas while undergoing a particular <a href="#">trip</a> .

#### 4.3.3.8.2 Agent LOS Time

Agent LOS time tables can be used to determine how long different agents spend at different levels of service.

Agent LOS Time Parameters	
<b>Name</b>	The name of the table.
<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><a href="#">Agent Filter</a></b>	Used to select when agents should be included in the LOS times; see the column descriptions below for details.
<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 ever satisfy the given filter during the given time range.

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,</b>	Total amount of time that the agent spent at each level of service while

<b>D,E,F] Duration</b>	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.
------------------------	--

4.3.3.8.3 Agent Process Chain Time

Agent process chain time tables can be used to produce agent summary information about a given process.

Agent Process Chain Time Parameters	
<b>Name</b>	The name of the table.
<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 afterwards will not be included.
<a href="#">Agent Filter</a>	Can be used to further restrict which agents are included in the table.
<b>Servers</b>	Specifies lists of start and end servers defining the process. 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>	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's processing time. Agents will only be included in the table if they satisfy the given filter during the given time range.

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.

#### 4.3.3.8.4 Agent Summary

Agent summary tables can be used to 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 the number of agents in various speed ranges.
- Right-click on an agent Start Time value to focus the 3D view and timeline on the agent at the time in entered the simulation.

<b>Agent Summary Parameters</b>	
<b>Name</b>	The name of the table.
<b><a href="#">Simulation run</a></b>	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.
<b><a href="#">Agent Filter</a></b>	Can be used to further restrict which agents are 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.

<b>Agent Summary Columns</b>	
<b>Agent ID</b>	Internal agent ID.

<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 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.

4.3.3.8.5 Agent Timetable Summary

Agent timetable summary tables can be used to display information for a set of agents associated with a set of reference events from a particular [Timetable](#) object. Reference events are not visible to any object outside of the Timetable object in which they are defined. The timetable summary table allows for queries that target agents generated by a Timetable schedule that were based on one or more 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.

<b>Agent Timetable Summary Parameters</b>	
<b>Name</b>	The name of the table.
<b><a href="#">Simulation run</a></b>	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.
<b><a href="#">Agent Filter</a></b>	Can be used to further restrict which agents are included in the table.
<b>Timetable</b>	The timetable object the table should refer to. 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>	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.

<b>Agent Timetable Summary Columns</b>	
<b>Agent ID</b>	Internal agent ID.
<b>From Reference Event</b>	Reference event from which the agent was created (blank if the schedule did not specify a 'From' reference event).
<b>To Reference Event</b>	Reference event the agent was sent to (blank if the 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> .

#### 4.3.3.8.6 Agent Token Time

Agent token time tables can be used to determine how long different agents spend holding different [tokens](#).

<b>Agent Token Time Parameters</b>	
<b>Name</b>	The name of the table.
<b><a href="#">Simulation run</a></b>	The simulation run for which token times should be calculated.
<b>Time range</b>	The time period over which token times should be calculated.
<b><a href="#">Agent Filter</a></b>	Used to select when agents should be included in the token times; see the

	column descriptions below for details.
<b>Tokens</b>	Which <a href="#">tokens</a> should 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, and ever possess any of the given tokens during that time range.

Agent Token Time Columns	
<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 <a href="#">trip</a> .

4.3.3.8.7 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	
<b>Name</b>	The name of the table.
<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>	Used to select a subset of agents to include in the table.
<a href="#">Trip</a>	The trip the table will refer to.
<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'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.

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).

#### 4.3.3.8.8 Origin/Destination

Origin/destination tables (matrices) can be used to check how many agents entered and exited the simulation at different portals during a particular time period.

Origin/Destination Parameters	
<b>Name</b>	The name of the table.
<a href="#">Simulation run</a>	The simulation run for which counts should be calculated.
<b>Time range</b>	The time period over which counts should be calculated.
<a href="#">Agent Filter</a>	Used to select a subset of agents to include 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.

The resulting table will contain one row for every entrance portal in the model, and one column for every exit portal. Each cell in the table corresponds to an entrance portal/exit portal pair. The value of the cell is the count of agents who, during the given time range, entered the simulation at the corresponding entrance portal and exited at the corresponding exit portal.

An additional row is included for agents that are "Present at Start" of the time range. Two additional columns are added for agents "Present at End" of the time range and for agents "Removed in Transit" via [actions](#) or an error. Thus, agents that do not completely enter and exit the simulation during a time range can also be accounted for.

Agents will only be included in the counts if they satisfied the given filter during the given time range.

#### 4.3.3.8.9 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	
<b>Name</b>	The name of the table.
<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>	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 simulation run.

<b>Agent Token Time Columns</b>	
<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> <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>

## 4.3.3.8.10 Server Summary

Server summary tables can be used to measure 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).

Server Summary Parameters	
<b>Name</b>	The name of the table.
<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="#">Agent Filter</a></b>	Used to select a subset of agents to include in the table.
<b>Servers</b>	Which servers the table will refer to.
<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 server, aggregated across the given runs using the given aggregation type. Only agents that satisfy the given filter will contribute to the computed values.

Server Summary Columns	
<b>Server Name</b>	Server name.
<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.

## 4.4 Simulation

### 4.4.1 Running 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>	This will 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>	This will run a <a href="#">Debug Simulation</a> , which will run more slowly than a console simulation but has 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>	This will 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

In all cases, 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. As a result, a simulation that is stopped early can still be used for playback and analysis, although care must be taken when interpreting results.

## 4.4.2 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, or is stopped by the user midway.

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.

## 4.4.3 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](#) for details. A properties pane is also 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.

### 4.4.3.1 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.

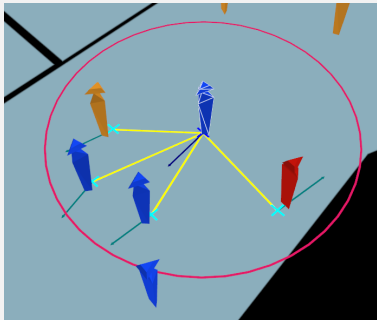
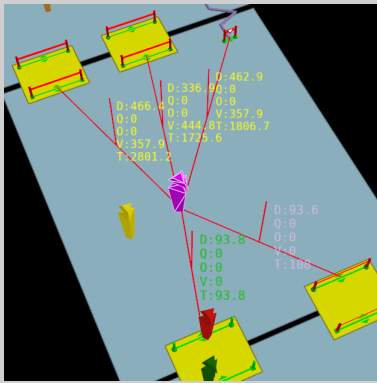
#### Simulation Control

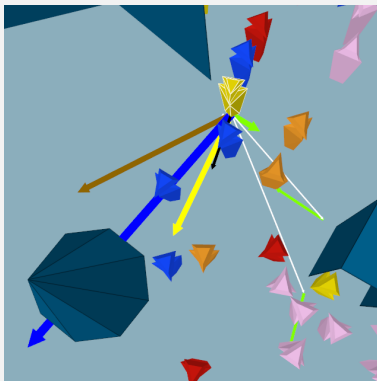
Simulation execution can be paused and controlled using the keyboard.

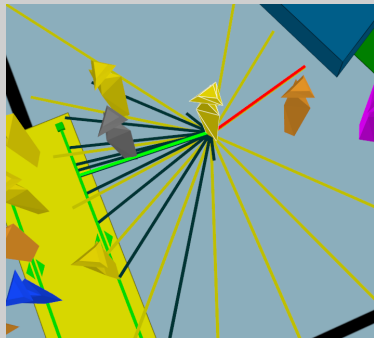
Simulation Control	
<b>Space Bar/ Up Arrow Key</b>	Toggles the simulation paused state.
<b>Left Arrow Key</b>	Advances the simulation by one frame at a time. This feature is only available when the simulation is paused.

### 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:

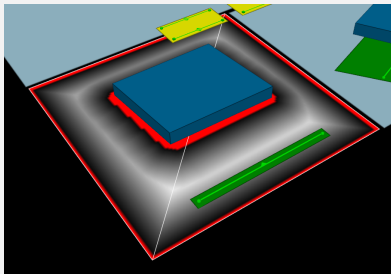
Agent Display Options		
<b>Neighbourhood</b>		Displays the local area vision bounds and indicates neighbouring agents within the selected agent's awareness. Note: The information displayed is for the previous simulation frame.
	<b>Yellow Line</b>	Lines from central agent to surrounding neighbours.
	<b>Pink Ring</b>	Displays geometric bounds of agent's awareness.
	<b>Turquoise Arrow</b>	Velocity of neighbouring agent.
	<b>Dark Blue Arrow</b>	Velocity of central agent.
<b>Route Costing</b>		Displays instantaneous costs (distance, queue, vertical, total) that the agent is subjected to. Red lines connect selected agents to other waypoints on the floor.
	<b>D Distance cost</b>	Total distance cost
	<b>Q Queue cost</b>	Total queue cost (proportional to size of queue)
	<b>O 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)

Agent Display Options																										
		<table border="1" style="width: 100%;"> <tr> <td style="width: 10%;"><b>T</b></td> <td style="width: 20%;"><b>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>Agents can be presented with multiple choices (e.g. 3 doors), meaning more than one exit on a floor can have an associated route cost. Different colours of text indicate different messages.</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 10%;"><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.</td> </tr> <tr> <td><b>Yellow</b></td> <td>This object has already been crossed.</td> </tr> <tr> <td><b>White</b></td> <td>Backtracking costs may be applied.</td> </tr> <tr> <td><b>White/pale</b></td> <td>This route has not been chosen to travel. Slightly different pale colours are used for each route to allow them to be distinguished from each other.</td> </tr> </table>	<b>T</b>	<b>Total cost</b>	Cumulative cost *	<b>Green</b>	This route has been chosen as the target.	<b>Red</b>	This route is not available.	<b>Yellow</b>	This object has already been crossed.	<b>White</b>	Backtracking costs may be applied.	<b>White/pale</b>	This route has not been chosen to travel. Slightly different pale colours are used for each route to allow them to be distinguished from each other.											
<b>T</b>	<b>Total cost</b>	Cumulative cost *																								
<b>Green</b>	This route has been chosen as the target.																									
<b>Red</b>	This route is not available.																									
<b>Yellow</b>	This object has already been crossed.																									
<b>White</b>	Backtracking costs may be applied.																									
<b>White/pale</b>	This route has not been chosen to travel. Slightly different pale colours are used for each route to allow them to be distinguished from each other.																									
<b>Social Forces</b>		<p>Displays the "social forces" to which the agent is subjected.</p> <table border="1" style="width: 100%;"> <tr> <td style="width: 10%;"><b>Bright Green</b></td> <td style="width: 20%;">Goal force</td> <td>Pulls neighbour towards desired destination</td> </tr> <tr> <td><b>Bright Yellow</b></td> <td>Obstacle constrained neighbour force</td> <td>Repels from surrounding neighbours</td> </tr> <tr> <td><b>Purple</b></td> <td>Drift force</td> <td>Drifts an agent in direction of bias when interacting with oncoming crowd</td> </tr> <tr> <td><b>Turquoise</b></td> <td>Collision veer force</td> <td>Veers agent towards direction bias in a head-on collision</td> </tr> <tr> <td><b>Orange</b></td> <td>Collision yield force</td> <td>Slows down and torques agent to avoid perpendicular collision</td> </tr> <tr> <td><b>White</b></td> <td>Cohesion force</td> <td>Attracts agents together within a crowd.</td> </tr> <tr> <td><b>Grey</b></td> <td>Orderly Queuing Force</td> <td>Pushes agents towards the front of the goal to which they are targeted, helping to ensure a straight and orderly queue.</td> </tr> <tr> <td><b>Brown</b></td> <td>Corner Force</td> <td>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)</td> </tr> </table>	<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>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)																								

Agent Display Options											
		<table border="1"> <tr> <td><b>Blue</b></td> <td>Obstacle constrained net force</td> <td>Resulting net force</td> </tr> <tr> <td><b>Pink</b></td> <td>Panic force</td> <td>Strong force pulling agent back to walkable surface when an agent has lost track of its surface</td> </tr> <tr> <td><b>Black</b></td> <td>Obstacle constrained velocity</td> <td>Resulting velocity</td> </tr> </table>	<b>Blue</b>	Obstacle constrained net force	Resulting net force	<b>Pink</b>	Panic force	Strong force pulling agent back to walkable surface when an agent has lost track of its surface	<b>Black</b>	Obstacle constrained velocity	Resulting velocity
<b>Blue</b>	Obstacle constrained net force	Resulting net force									
<b>Pink</b>	Panic force	Strong force pulling agent back to walkable surface when an agent has lost track of its surface									
<b>Black</b>	Obstacle constrained velocity	Resulting velocity									
<b>Surface Probe</b>		<p>Displays agent awareness of surrounding space as it relates to obstacle avoidance and the direction of goal.</p> <table border="1"> <tr> <td><b>Black</b></td> <td>Displays directions of potential forward routes</td> </tr> <tr> <td><b>Green</b></td> <td>Displays direction of best forward route</td> </tr> <tr> <td><b>Gold</b></td> <td>Points to nearby obstacles/boundaries</td> </tr> <tr> <td><b>Red</b></td> <td>Points to closest obstacle/boundary</td> </tr> </table>	<b>Black</b>	Displays directions of potential forward routes	<b>Green</b>	Displays direction of best forward route	<b>Gold</b>	Points to nearby obstacles/boundaries	<b>Red</b>	Points to closest obstacle/boundary	
<b>Black</b>	Displays directions of potential forward routes										
<b>Green</b>	Displays direction of best forward route										
<b>Gold</b>	Points to nearby obstacles/boundaries										
<b>Red</b>	Points to closest obstacle/boundary										

### Scene Object Display Options

The right-click context menu for scene objects during a debug simulation has the same display options as in authoring/playback, plus the following:

Scene Object Display Options	
<b>Obstacle Map (surfaces)</b>	 <p>Displays the obstacle surface map for the selected scene object. Areas that are unavailable to agents (obstacles, floor edges) are coloured red. Note: The map may require several seconds to calculate and display after pressing the button. Be cautious when using this attribute on very large objects.</p>

### 4.4.3.2 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 currently selected.

#### Agents

Properties Tab	
<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>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>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>Queuing?</b>	True if the agent is queuing to reach the next waypoint.
<b>Time in Queue (s)</b>	If queuing, this is the time the agent has so far spent queuing 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).

#### 4.4.4 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 8.0).
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
-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	Displays the list of available parameters and exits.	-help
-nothreads	Disable the use of threads during the simulation.	-nothreads
-popscal e	A number greater than 0, used to scale the number of agents generated by all events.	-popscale 2.0
-project #	Specify the MassMotion project file (.mm file) to open and run.	-project C:\mm\testproject\testproject.mm
-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	-threads 1 (all operations are performed in the main thread) -threads 8 (8 threads are used)

	value is recommended.	
-verbosity #	Control the number and verbosity of messages written to the project log.txt file. Possible values include ERROR (only log error messages), WARNING (log error and warning messages), APPLICATION (log errors, warnings, and standard messages), VERBOSE, DEBUG. The default is APPLICATION.	-verbosity DEBUG -verbosity APPLICATION
-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

#### 4.4.5 Generated Simulation Files

The following files can be produced when execution 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 a visual map of the available space on the given surface and the distance from every point to the nearest obstacle or surface edge.

	<p>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> <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>
<b>Approach Map (*.jpg)</b>	<p>File (.jpg) containing a visual map 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>
<b>CostTree (*.csv)</b>	<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>

### 4.4.6 Randomness

#### Random Seed

MassMotion uses random numbers throughout the simulation. All random numbers are generated from an initial integer seed value (see [Project Settings](#)). A project simulated multiple times with the same random seed should produce exactly the same results. Changing the seed or any element in the project will result in different random numbers during the simulation and so different simulation results.

Distributions	
<a href="#">Agent Start</a>	Duration based distribution used by <a href="#">journey</a> and <a href="#">circulate</a> events to determine when agents enter the simulation.
<a href="#">Single Value</a>	Distributions used nearly everywhere, influencing how agents may interact with the scene.

#### 4.4.6.1 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 normal mean. <b>Sigma:</b> The normal standard deviation.
<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.

#### 4.4.6.2 Standard Distributions

Many object numeric 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 according to a distribution, with each agent given a single speed value according to the distribution.

For information on duration based distributions used by some events in 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.

	<p><b>Min:</b> The minimum possible value.  <b>Max:</b> The maximum possible value.</p> <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 normal mean.  <b>Sigma:</b> The normal standard deviation.</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. This section expands on a number of concepts that are important to many of the analysis functions in MassMotion. This section also details how to [export agent position data](#), [images](#), and [videos](#). For a comprehensive description of all the available analysis functions please refer to the [Analysis Objects](#) section.

### 4.5.1 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 profile 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 avatar 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.</p> <p><b>In Transit:</b> The agent is moving freely towards its target.</p> <p><b>Queuing:</b> The agent is queuing for its target.</p> <p><b>Waiting for Access:</b> The agent is waiting for access to its target (waiting for a gate to open).</p> <p><b>Pre-Contact Wait:</b> The agent is queuing for a server.</p> <p><b>In-Contact:</b> The agent is being processed by a server.</p> <p><b>Post-Contact Wait:</b> The agent is being held by a server until there is capacity downstream in the process chain.</p>
<b>Target</b>	The object to which the agent is moving.

<b>Tokens</b>	A list of tokens 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>Actions:</b>	<p>A list of all actions 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 an action event as it fires, it is recorded as 'Action Event'. However, if the action event 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 action event.</p>

### 4.5.2 Areas

Various types of analysis objects such as [agent filters](#), [transitions](#), [trips](#) and [population count graphs](#) use the concept of 'areas'. For the purposes of analysis, several types of objects can be used as areas:

Area Type	Agents Considered 'In'
<b>Walkable object</b>	Agents currently on the walkable object (floor, link, escalator, ramp or stair).
<a href="#">Analysis Region</a>	Agents with the point at the centre of their feet contained within the region.
<a href="#">Zone</a>	Agents on any of the walkable objects that are part of the zone.
<b>Collection</b>	<a href="#">Collections</a> 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.

### 4.5.3 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 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 <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="#">analysis cordon</a>.</p> <p>With a collection: Agents crossing any cordon in the collection are at the transition.</p>
<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>

Transition Types	
<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>

#### 4.5.4 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).

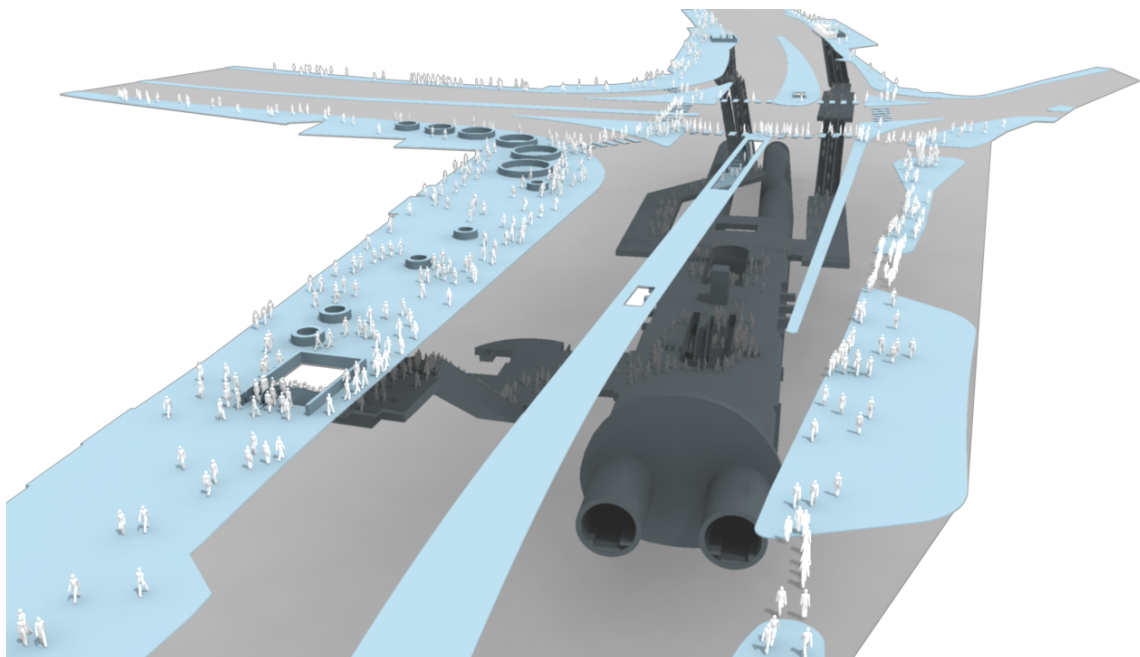
LOS Colour Mapping Values				
<b>Fruin Walkways</b>	Area of circle used to calculate density: 3.24m <sup>2</sup> .			
	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour
	A	$x \leq 0.309$	$x \geq 3.24$	
	B	$0.309 < x \leq 0.431$	$3.24 > x \geq 2.32$	
	C	$0.431 < x \leq 0.719$	$2.32 > x \geq 1.39$	
	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>	Area of circle used to calculate density: 1.81m <sup>2</sup> .			
	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>	Area of circle used to calculate density: 1.21m <sup>2</sup> .			
	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>	Area of circle used to calculate density: 2.70m <sup>2</sup> .			
	LOS	Density (person/m <sup>2</sup> )	Space (m <sup>2</sup> /person)	Colour
	A	$x \leq 0.370$	$x \geq 2.70$	
	B	$0.826 < x \leq 0.435$	$2.70 > x \geq 2.30$	
	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	$1.00 < x$	$1.00 > x$		

### 4.5.5 Alembic Export

Alembic is an open computer graphics interchange framework. Alembic distills 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 as 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>Output file</b>	What Alembic (.abc) file to export to.
<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.
<a href="#">Agent filter</a>	Which agents to include in the output.

### 4.5.6 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 analysis 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
<b>File</b>	The CSV file to which the table will be written.
<b>Simulation run</b>	The simulation run for which data will be written to the table
<b>Time range</b>	The time period over which agent positions will be included.
<b>Sampling period</b>	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.
<a href="#">Agent Filter</a>	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.</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>

**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

**4.5.7 Movie and Image Export**

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

Default settings for movie/image export can be set in the application preferences available through the main window [menu bar](#).

**Project and scene changes**

When the export window is first opened, the scene's appearance is copied from the scene in the main window. This includes agent appearance, background colour and camera position. These can be further changed within the export window without affecting the main window.

The export window operates on a copy of the current project and so any modifications within the main window will not affect the export window. However, simulation databases are locked and existing [simulation runs](#) cannot be re-run while the export window is open.

Both the appearance of the scene and the project in general can be reloaded from the main window by pressing the "Reload" button.

Toolbar	
<b>Reload</b>	Reload changes to the project and scene's appearance from the main window.
<b>Render</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>Single Image</b>	Take a single image. The user will be prompted for a file name. The following file formats are supported: *.png, *.jpg, *.tiff

File Options	
<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.

<b>Appearance</b>	
<b>Resolution</b>	The resolution of movies, frames and images produced. Can dramatically affect the size of file outputs.  Changing the resolution will darken regions at the edges of the scene. These regions are cropped in the outputs to fit the desired resolution.
<b>Background</b>	The background colour to use.

<b>Overlay</b>	
<b>Text</b>	The colour to use to display 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.

---

# Index

## - G -

Graph 165  
Graph Structure 164

Endnotes 2... (after index)



Back Cover