



403 Poyntz Ave., Suite B  
Manhattan, KS 66502, USA  
Phone: +1-785-770-8511  
Email: [support@thunderheadeng.com](mailto:support@thunderheadeng.com)  
Web: <https://www.thunderheadeng.com>



# Pathfinder User Manual

Version: 2024-1



# Table of Contents

Disclaimer .....	4
Acknowledgements .....	5
1. Getting Started .....	6
1.1. Introduction .....	6
1.2. System Requirements .....	6
1.3. Download and Install .....	6
1.4. Graphical User Interface .....	8
1.5. Model Representation .....	10
1.6. Simulation Modes .....	11
1.7. System Requirements .....	12
1.8. Configuration Files .....	13
1.9. Contact Us .....	14
2. Pathfinder Basics .....	15
2.1. Navigation View .....	15
2.2. 3D and 2D Views .....	18
2.3. View Options .....	24
2.4. Model Organization with Groups .....	27
2.5. Keyboard Shortcuts .....	29
2.6. Object Sets and Discrete Distributions .....	30
3. Creating Movement Space .....	35
3.1. Floors .....	35
3.2. Rooms .....	40
3.3. Obstructions/Holes .....	49
3.4. Obstacles .....	52
3.5. Doors .....	58
3.6. Stairs .....	64
3.7. Inferred Stairs .....	70
3.8. Ramps .....	71
3.9. Escalators .....	71
3.10. Moving Walkways .....	72
3.11. Elevators .....	72
3.12. Exits .....	82
3.13. Undo/Redo .....	83
4. Importing Files .....	84

4.1. Importing Images .....	84
4.2. Importing CAD Files .....	87
4.3. Working with Imported Data .....	100
4.4. Importing FDS Output Data .....	124
4.5. Importing Custom Avatars .....	126
4.6. Importing Custom Animations .....	131
5. Occupants .....	139
5.1. Profiles .....	139
5.2. Vehicle Shapes .....	161
5.3. Behaviors .....	164
5.4. Generating Occupants .....	188
5.5. Redistributing Profiles and Behaviors .....	201
5.6. Randomizing Occupants' Positions .....	201
5.7. Reducing Population .....	202
5.8. Occupant Tags .....	202
6. Queues .....	204
6.1. Services .....	204
6.2. Paths .....	204
6.3. Path Nodes .....	205
7. Assisted Evacuation .....	206
7.1. Assistance Process Overview .....	206
7.2. Preparing Clients .....	207
7.3. Preparing Assistants .....	209
7.4. Preparing Teams .....	209
8. Occupant Grouping .....	212
8.1. Grouped Movement .....	212
8.2. Movement Groups .....	213
8.3. Movement Group Templates .....	213
8.4. Adding Grouped Occupants from Occupant Sources .....	215
9. Occupant Targets .....	217
9.1. Creating Occupant Targets .....	217
9.2. Occupant Target Properties .....	218
9.3. Orienting Occupant Targets .....	219
9.4. Prioritizing Occupant Targets .....	220
9.5. Using Occupant Targets .....	222
9.6. Occupant Target Reservation System .....	223
9.7. Occupant Target Limitations .....	224

9.8. Occupant Target Output .....	225
10. Triggers .....	226
10.1. Creating Triggers .....	226
10.2. Trigger Behavior .....	228
10.3. Occupant Response to a Trigger .....	229
10.4. Trigger Memory .....	231
10.5. Trigger Rank .....	232
10.6. Trigger Properties .....	232
10.7. Trigger Limitations .....	236
10.8. Trigger Output .....	237
11. Views .....	238
11.1. Creating a View .....	238
11.2. Recalling a View .....	238
11.3. Editing a View .....	239
11.4. Views in the Results .....	239
11.5. Camera Tours .....	239
12. Editing and Copying Objects .....	240
12.1. Transforming and Copying Objects .....	240
12.2. Manipulating Objects with Handles .....	244
12.3. Enabling and Disabling Objects .....	247
13. Working with Large Models .....	249
13.1. Selection .....	249
13.2. Show Referencing Objects .....	249
13.3. Object Tags .....	250
13.4. Bulk Renaming .....	253
14. Model Analysis .....	256
14.1. Measurement Regions .....	256
14.2. Measuring Distances .....	257
15. Simulating .....	258
15.1. Parameters .....	258
15.2. Starting and Managing a Simulation .....	269
15.3. Stopping and Resuming a Simulation .....	270
16. Results .....	272
16.1. Summary Report .....	272
16.2. Cumulative Output .....	274
16.3. Door History .....	275
16.4. Room History .....	280

16.5. Measurement Regions Data .....	281
16.6. Occupant Parameters .....	284
16.7. Interpersonal Distance .....	294
16.8. Occupant Summary .....	301
16.9. Occupant History .....	305
16.10. Groups Output .....	310
16.11. Triggers History .....	311
16.12. Occupant Target History .....	312
16.13. 3D Results .....	313
17. Troubleshooting .....	315
17.1. Occupants cannot reach their goals, or they are getting stuck .....	315
17.2. Warnings and errors in the navigation tree .....	316
17.3. Find objects related to a specific object .....	316
17.4. System memory issues .....	316
17.5. Issues with custom avatars .....	318
17.6. Video display problems and crashes on startup .....	318
Appendix A: Pre-defined Speed Profiles .....	320
A.1. Profiles from the IMO's Revised Guidelines for Passenger Ships .....	320
A.2. Profiles from Fruin's Pedestrian Planning and Design .....	321
Appendix B: Avatar and Animation File Format .....	323
B.1. Avatar JSON File .....	323
B.2. Example Animation JSON File .....	325
B.3. Avatar/Animation JSON Structure .....	327
Bibliography .....	342

# Disclaimer

Thunderhead Engineering makes no warranty, expressed or implied, to users of Pathfinder, and accepts no responsibility for its use. Users of Pathfinder assume sole responsibility under Federal law for determining the appropriateness of its use in any particular application; for any conclusions drawn from the results of its use; and for any actions taken or not taken as a result of analyses performed using these tools.

Users are warned that Pathfinder is intended for use only by those competent in the field of egress modeling. Pathfinder is intended only to supplement the informed judgment of the qualified user.

The software package is a computer model that may or may not have predictive capability when applied to a specific set of factual circumstances. Lack of accurate predictions by the model could lead to erroneous conclusions. All results should be evaluated by an informed user.

All other product or company names that are mentioned in this publication are tradenames, trademarks, or registered trademarks of their respective owners.

Throughout this document, the mention of computer hardware or commercial software does not constitute endorsement by Thunderhead Engineering, nor does it indicate that the products are necessarily those best suited for the intended purpose.

# Acknowledgements

This work was originally made possible by a Small Business Innovative Research (SBIR) grant (2005-2007) by the United States National Science Foundation.

We would like to thank Rolf Jensen and Associates for their assistance with testing and other suggestions that helped guide the development of the simulator.

We would also like to thank the users whose feedback helps us improve the software and incorporate more useful features. The Pathfinder support forum can be found at [www.thunderheadeng.com/pathfinder/support/](http://www.thunderheadeng.com/pathfinder/support/).

# Chapter 1. Getting Started

## 1.1. Introduction

Pathfinder is an agent-based egress and human movement simulator. It provides a graphical user interface for simulation design and execution as well as 2D and 3D visualization tools for results analysis.

## 1.2. System Requirements

### 1.2.1. Minimum System requirements

Table 1. Minimum System Requirements

	Intel	AMD
<b>Operating System:</b>	Windows 10	Windows 10
<b>Processor:</b>	Core i5 3570 (4 core)	Athlon x4 970 (4 core)
<b>Graphics Card:</b>	Integrated HD Graphics	Integrated HD Graphics
<b>Memory:</b>	8GB RAM	8GB RAM

### 1.2.2. Recommended System requirements

Table 2. Recommended System Requirements

	Intel	AMD
<b>Operating System:</b>	Windows 10	Windows 10
<b>Processor:</b>	Core i7 8700 (6 Cores / 12 Threads)	Ryzen 7 1700 (8 Cores / 12 Threads)
<b>Graphics Card:</b>	NVIDIA GeForce GTX570 / AMD Radeon HD7870	NVIDIA GeForce GTX570 / AMD Radeon HD7870
<b>Memory:</b>	16GB RAM	16GB RAM

## 1.3. Download and Install

You can download the current version, sign up for a free trial, and purchase the software from the [Pathfinder Support Page](#). This page also provides instructions for installation and activation.

Troubleshooting info can be found on the [Pathfinder FAQs page](#). There is no functional difference



between the trial version of Pathfinder and the full version, the only limitation is the trial license duration.

Administrator privileges are required to install Pathfinder. This is necessary because the installer adds processes to the operating system for license management.

If an older version of Pathfinder is present, the installer will automatically remove the older version during the installation process. License files and properties files (e.g., recently opened PTH files) from the older installation will be preserved and utilized by the updated version of Pathfinder.

Pathfinder will regularly check for and notify the user of available updates to the software when configured to do so. By default, Pathfinder will check for updates on startup and display the relevant information in the Check For Updates dialog when one is available.

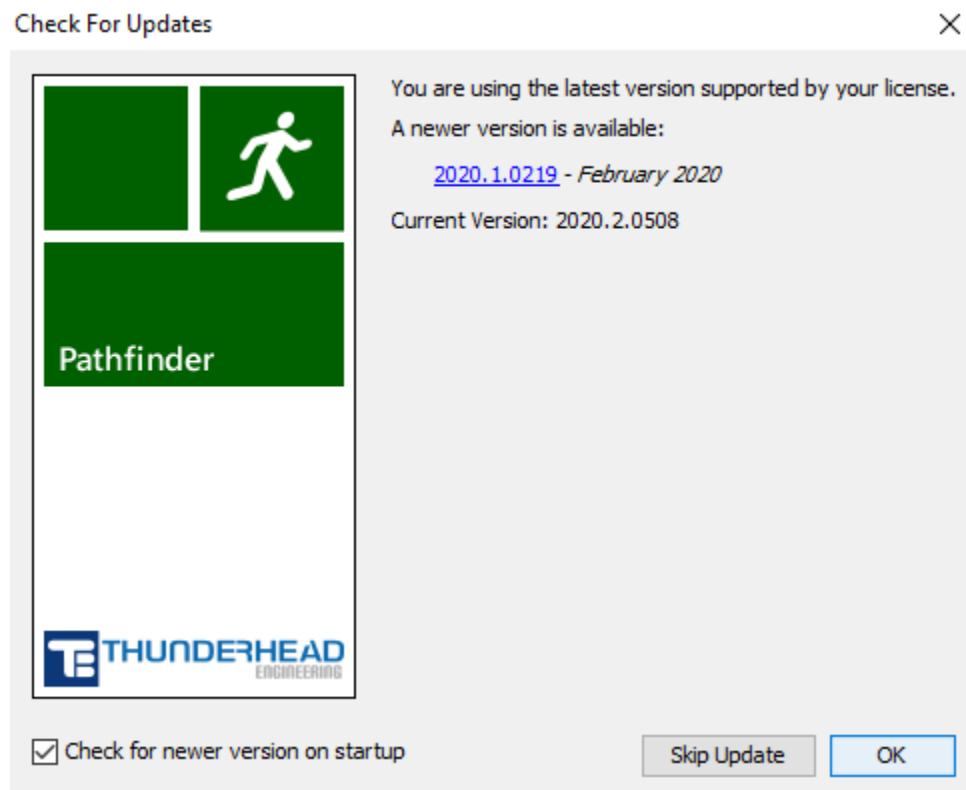


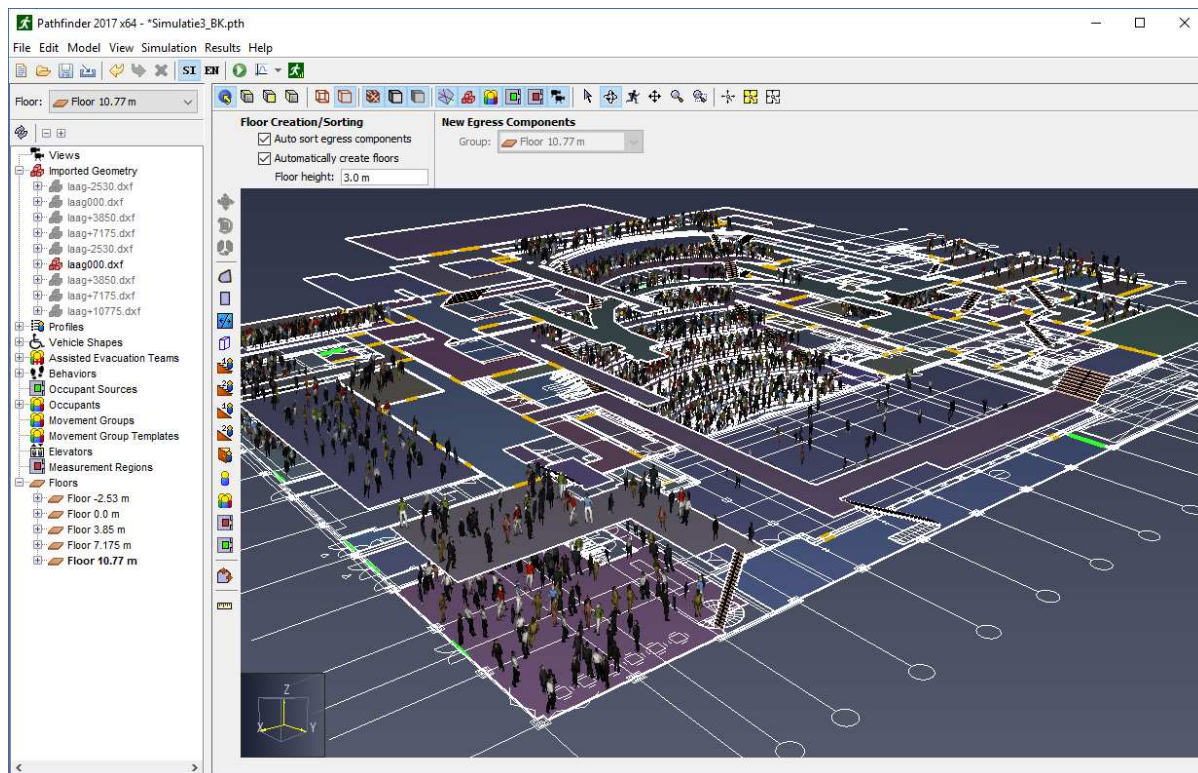
Figure 1. The Check For Updates dialog.

Users can also access this dialog by navigating to **Help – Check For Updates**

The dialog can be disabled on startup by unchecking **Check for newer version on startup**. Users can also elect to skip the current update by clicking the **Skip Update** button, which will prevent update notifications until a new version is released beyond the latest version.

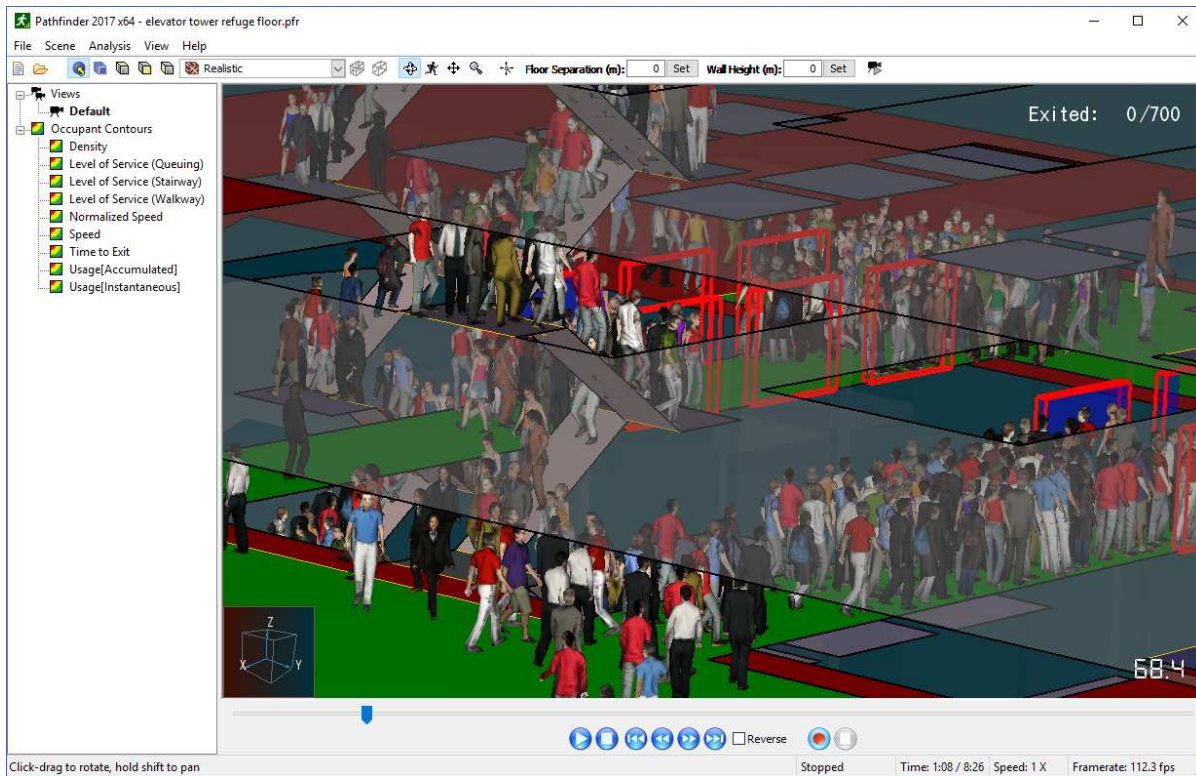
## 1.4. Graphical User Interface

Pathfinder includes a graphical user interface that is used primarily to create and run simulation models. A screenshot of this user interface is shown in [Figure 2](#). This screenshot displays a model of the Theater de Vest in Alkmaar, Netherlands. The model was created by Van Hooft Adviesburo. For clarity, the image shows only one of the DXF files used to create the geometry. The model includes 2177 occupants.



**Figure 2.** An example of the graphical user interface. Model of the Theater de Vest created by Van Hooft Adviesburo.

Pathfinder also includes a second program designed specifically for high-performance visualization of 3D time history. The 3D Results program is shown in [Figure 3](#). In this image, occupants are gathering at a refuge area before proceeding to elevators. Transparency has been used to help view occupants on the refuge floor.



**Figure 3. An example of the 3D Results view, showing occupants gathering at a refuge area before proceeding to elevators.**

In addition to 3D visualization, Pathfinder also provides output in the form of 2D time history plots of CSV (comma separated values) out files and a text summary of room clearing times and doorway flow rates. An example time history plot can be seen in [Figure 4](#). This plot shows the number of occupants in the refuge area and the total number of occupants in the building.

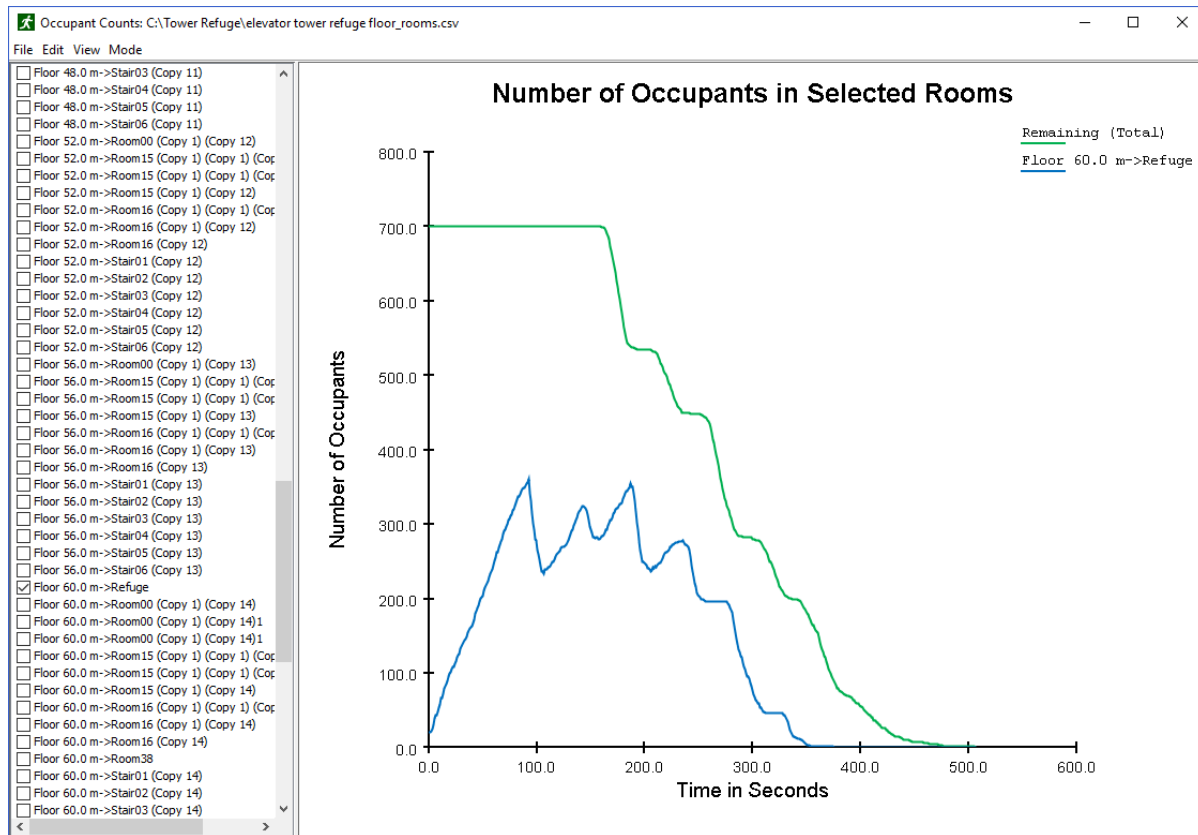


Figure 4. An example time history graph

## 1.5. Model Representation

The movement environment is a 3D triangulated mesh (Figure 5), designed to match the real dimensions of a building model. This movement mesh can be entered manually or automatically based on imported data (e.g. FDS geometry).

Walls and other impassable areas are represented as gaps in the navigation mesh. These objects are not actually passed along to the simulator, but are represented implicitly because occupants cannot move in places where no navigation mesh has been created.

Doors are represented as special navigation mesh edges. In all simulations, doors provide a mechanism for joining rooms and tracking occupant flow. Depending on the specific selection of simulation options, doors may also be used to explicitly control occupant flow.

Stairways are also represented as special navigation mesh edges and triangles. Occupant movement speed is reduced to a factor of their level travel speed based on the incline of the stairway. Each stairway implicitly defines two doors. These doors function just like any other door in the simulator but are controlled via the stairway editor in the user interface to ensure that no geometric errors result from a mismatch between stairways and the connecting doors.

Elevators are called to a floor when occupants arrive at the elevator door. The elevator model includes capacity, pick-up and discharge floors, and the ability to group elevators in banks.

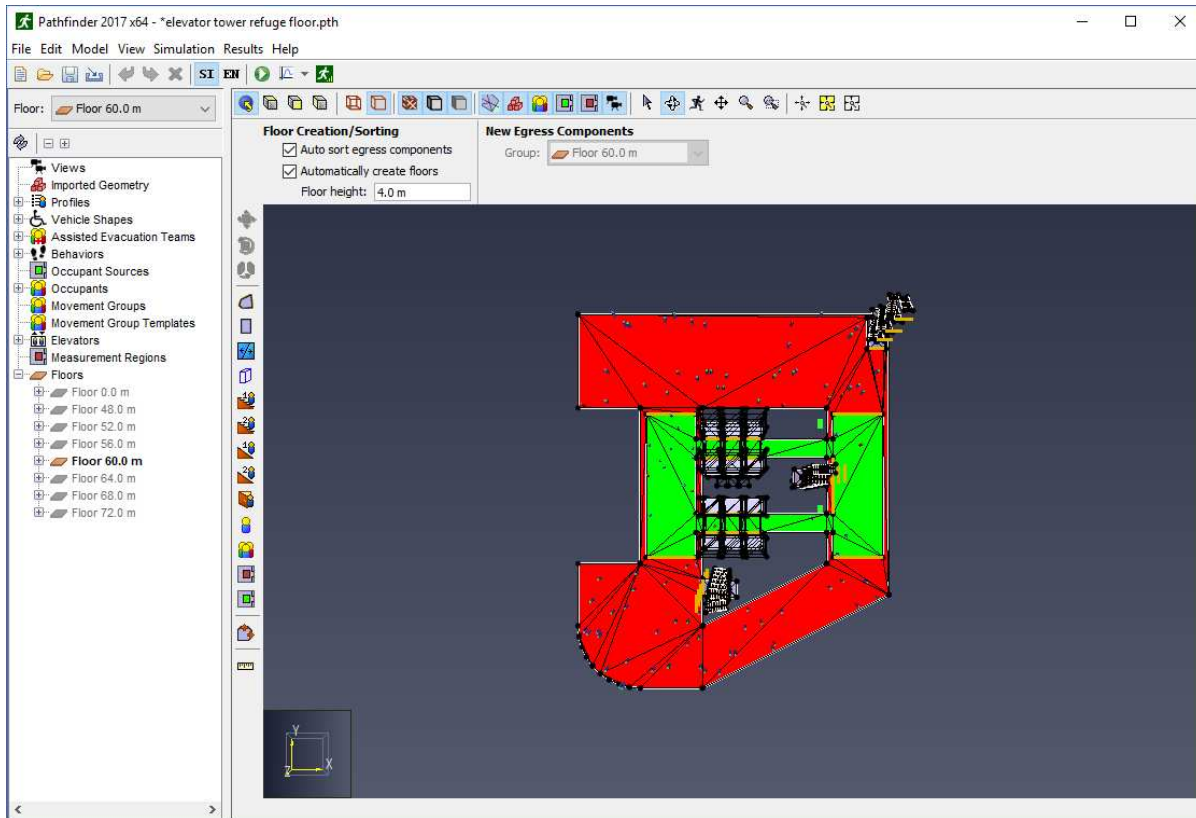


Figure 5. The triangulated navigation mesh used to represent one floor in the model.

Each occupant is defined by position, a profile that specifies size, speed, etc., and a behavior that defines goals for the occupant. The behavior allows scripting so that, for example, an occupant may wait at a location for a specified time and then proceed to an elevator. The occupant is represented as an upright cylinder on the movement mesh and movement uses an agent-based technique called inverse steering. Each occupant calculates movements independently.

## 1.6. Simulation Modes

Pathfinder supports two movement simulation modes. In "Steering" mode, occupants use a steering system to move and interact with others. This mode tries to emulate human behavior and movement as much as possible. SFPE mode uses a set of assumptions and hand-calculations as defined in the *Engineering Guide to Human Behavior in Fire* (SFPE 2019). In SFPE mode, occupants make no attempt to avoid one another and can interpenetrate, but doors impose a flow limit and velocity is controlled by density. You can freely switch between the two modes within the Pathfinder user interface and compare answers.

More information about both modes is provided in the Pathfinder Technical Reference ([Pathfinder](#)



*Technical Reference, n.d.).*

## 1.7. System Requirements

System requirements depend on the type of model being analyzed. To illustrate this, two different models (Table 3) were evaluated using a laptop running 64-bit Windows 8 Pro with an Intel Core i7 2.60 GHz processor, 8 GB of RAM, and NVIDIA NVS 5200M graphics card. The first model (Figure 6) had a single room with 50,000 occupants and did not include any imported geometry. The second model (Figure 7) imported a relatively complex Revit model to create the Pathfinder model and had 3,000 people.

The key parameters are the number of people in the model and the model complexity, measured by the number of navigation mesh triangles used in the Pathfinder solution and the number of imported Revit primitives (triangles). The simple model had only 4 triangles, with the consequence that the movement calculation of the path for each person is simple and that display performance is related to the drawing of people. The Revit model had 21,480 triangles for the navigation mesh but over 1,300,000 triangles for the Revit geometry.

The model with 50,000 people (Figure 6) solved in about 18 minutes, while the Revit model with 3,000 people (Figure 7) took about 5 minutes. The graphical display performance for the model with 50,000 people was responsive at 15 frames/sec, while the Revit model with the imported geometry displayed was slow at 5 frames/sec. When only the Pathfinder navigation mesh was displayed, the Revit model was responsive.

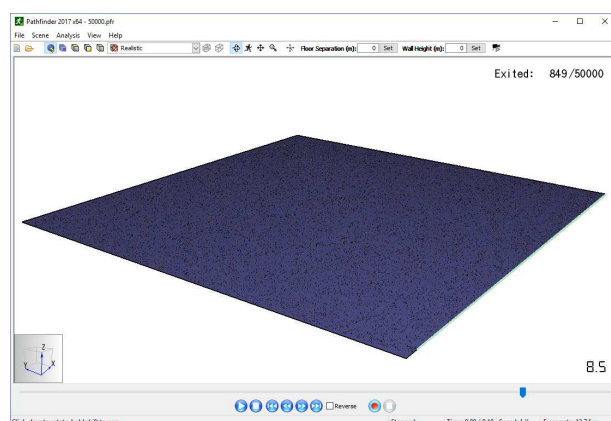


Figure 6. Model with 50,000 people



Figure 7. Revit import model with 3,000 people

Table 3. Comparison of performance for the two models

Parameter	Model	
	People	Revit Import

Parameter	Model	
Number of occupants	50,000	3,000
Number navigation triangles	4	21,480
Number Revit face primitives	0	1,300,000
CPU solution time (s)	1090	297
Navigation mesh display rate (fps)	~15	~70
Imported geom display rate (fps)	n/a	~5

The minimum requirements to run Pathfinder include:

- 32 or 64-bit Windows 7 or higher
- A processor the performance of an Intel i5
- 4 GB of RAM
- Graphics support for OpenGL 1.2

For a balanced performance we recommend:

- 64-bit Windows 7 or higher
- Intel i7-3770 (3.4 GHz, 4 Cores) processor
- 8 GB of RAM
- Graphics support for OpenGL 3.2 with an installed graphics card for large Revit models.

Revit models place a premium on graphics capability. We have found that mid-cost gaming graphics cards (GeForce GTX 570 / Radeon HD 7870 or equivalent) allow us to display relative large Revit models with good performance. Benchmarking sites that we find useful to compare CPU and graphics card performance can be found at: <http://www.cpubenchmark.net/> and <http://www.videocardbenchmark.net/>.

## 1.8. Configuration Files

Pathfinder stores data related to user preferences in a file called Pathfinder.props. By default, this file can be found in one of:

```
%APPDATA%\Pathfinder\Pathfinder.props
%PROGRAMDATA%\Pathfinder\Pathfinder.props
```

If at least one of these files exists, Pathfinder will load the user preferences. If both files exist, Pathfinder will load user preferences from both files, giving preference to the file located in the **APPDATA** folder. This way the preference file located in the **PROGRAMDATA** folder can be shared among multiple machines, and the file located in the **APPDATA** folder on each machine overrides the shared settings.

The **PROPS** file is stored in a plaintext format, and can be viewed or edited with any conventional text editor. While it is not recommended to edit the file directly, some troubleshooting techniques may involve deleting the **PROPS** file so that a new one can be created from scratch by Pathfinder.

## 1.9. Contact Us

Thunderhead Engineering  
403 Poyntz Avenue, Suite B  
Manhattan, KS 66502-6081  
USA

Sales Information: [sales@thunderheadeng.com](mailto:sales@thunderheadeng.com)  
Product Support: [support@thunderheadeng.com](mailto:support@thunderheadeng.com)  
Phone and Fax: +1.785.770.8511



## Chapter 2. Pathfinder Basics

Pathfinder provides three main views for working on evacuation models: the 2D View, 3D View, and Navigation View. These views represent your current model. If an object is added, removed, or selected in one view, the other views will simultaneously reflect the change. Each view is briefly described below.

### Navigation View

This view lists all objects in the model in a hierarchical format. It can be used to quickly locate and modify objects by name.

### 3D View

This view shows a 3D representation of the current model. The model can be explored and modified using various tools.

### 2D View

This view is very similar to the 3D View, but it provides an additional snapping grid and an orthographic view of the model.

## 2.1. Navigation View

The Navigation View helps you quickly find objects and data that are not always easily accessible from the 3D and 2D views.

The Navigation View is arranged in the following groups:

### Views

This group contains user-defined camera positions.

### Imported Geometry

This group stores items that were imported from an image or an IFC, FDS, PyroSim, or other CAD model. These objects do not affect the simulation but are carried through to help with results analysis. They can also be used to generate model elements as described in [Section 4.3](#).

### Profiles

This group contains the occupant profiles that have been created using the **Edit Profiles** dialog.

### Vehicle Shapes

This group contains occupant shapes defined as polygons.

### **Assisted Evacuation Teams**

This group contains user-defined teams of assistants and clients in an assisted evacuation scenario.

### **Behaviors**

This group contains user-defined scripts that tell occupants how to behave.

### **Occupant Sources**

This group stores occupant sources, which are defined either by area, room, or door, and can generate occupants during the simulation.

### **Occupants**

This group contains every occupant in the model. If occupants are added to the model using a tool that adds more than one occupant at a time, they will be collected in a sub-group.

### **Movement Groups**

This group stores occupants grouped into movement groups. Members of a movement group will stay together during the simulation.

### **Movement Group Templates**

This group contains templates that can be used to automatically create large number of movement groups. Templates can also be used for generating movement groups during the simulation.

### **Elevators**

This group contains elevators in the model.

### **Measurement Regions**

This group contains regions defined by area, within which velocity and density measurements will be taken.

### **Floors**

This group defines the floors in the model, and each floor contains all geometry necessary to create a movement mesh, including room, stairway, ramp, door, and exit definitions ([Figure 8](#)).

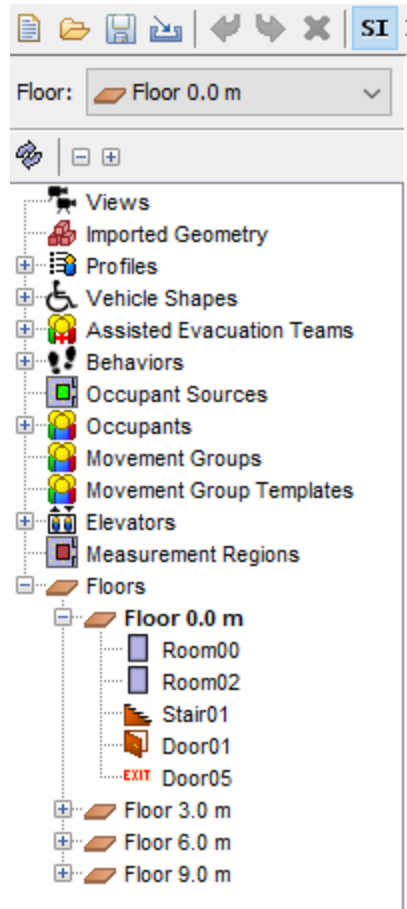


Figure 8. Navigation View with the Floors group expanded.

The buttons directly above the Navigation View perform the following actions:

#### Auto Expand Selection

When an object (or occupant) in the 3D or 2D view is selected, this action will expand the groups of the Navigation View as needed to show the selected object.

#### Collapse All

Collapses all expanded groups in the Navigation View.





#### Expand All

Expands all groups in the Navigation View (including sub-groups).

The **Floor** selection box above the Navigation View panel (see the top of [Figure 8](#)) can be used to manage floors. Any time a room, stair, ramp, or door is created it is added to a floor group matching the current selection in the **Floor** box. Changing the selection in the **Floor** box will cause the newly selected floor to be shown and all other floors to be hidden. Also, the **Z** property for all drawing tools will automatically default to the height of the floor currently selected in the **Floor** box. The visibility of any object or group of objects can always be manually set using the

right-click context menu. This technique is useful if you want to show two floors at the same time (e.g. when creating a stairway).

## 2.2. 3D and 2D Views

The 3D and 2D views as shown in [Figure 9](#) are the main views in which drawing is performed in Pathfinder. Both views contain tools to draw egress geometry and navigate in a model. The main difference between the two views is that the 3D view allows the model to be viewed from any direction, whereas the 2D view only allows viewing from one, orthographic direction. In addition, the 3D view contains no snap grid, whereas the 2D view does. The 3D view is entered by selecting the perspective camera, , and the 2D view is entered by selecting one of the orthographic cameras, , , or .

At the top of the view is several buttons that show different camera modes, display options, and navigation modes. The panel under this is known as the **Property Panel** and is a selection context-sensitive panel. If a drawing tool is selected, it will show properties that can be used to help draw. If no drawing tool is selected, and an object or several objects are selected, this panel will show the properties relevant to the selection. The panel of buttons on the left shows move/copy and drawing tools. The small panel at the bottom displays messages relevant to the current tool.

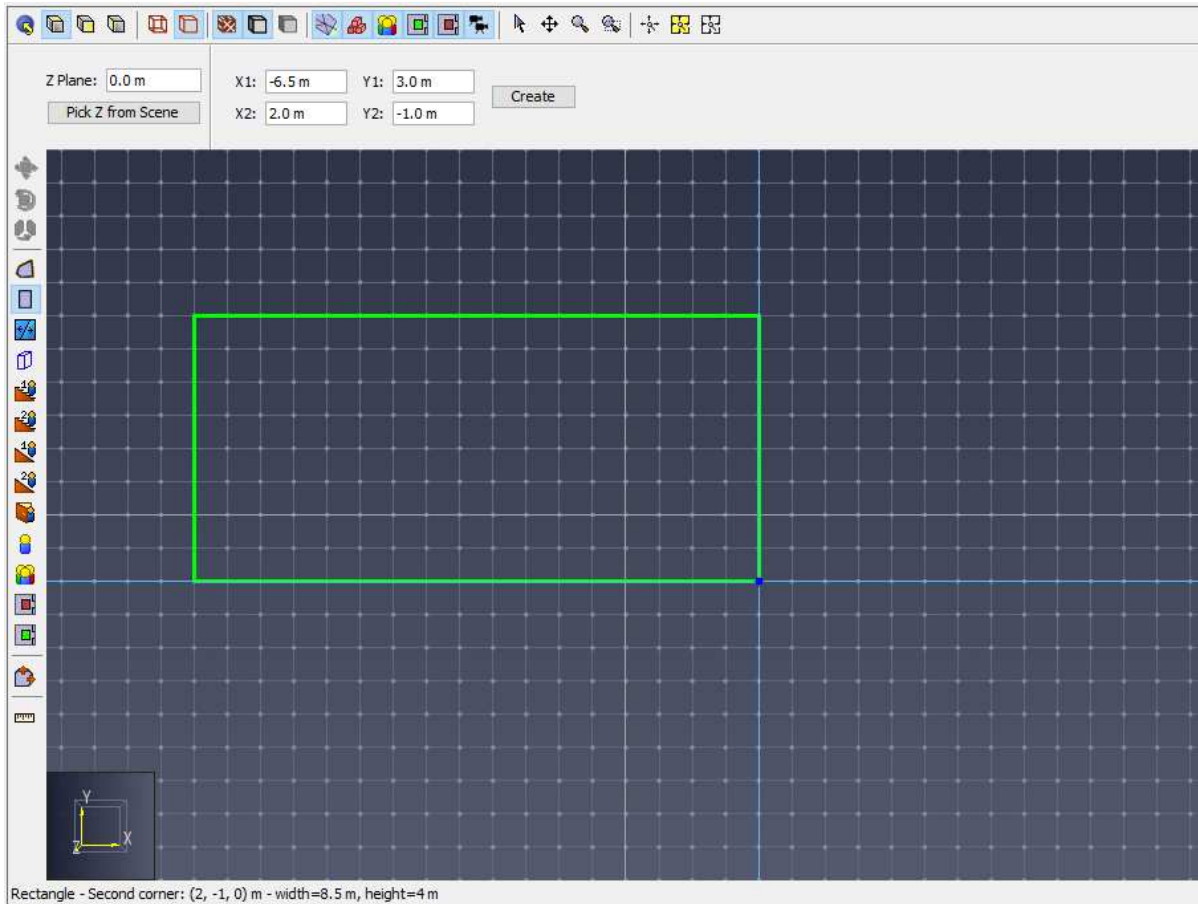



Figure 9. 3D and 2D views

### 2.2.1. Navigating the 3D View

Several tools are provided for navigating through the model in the 3D view, including orbit, roam, pan, and zoom tools.

The main navigation tool for the 3D view is the **Orbit** tool, .

- By left-clicking and dragging, the model is rotated about its center point.
- The scroll-wheel can be used to zoom in and out on a specific point.
- Holding *SHIFT* on the keyboard and then clicking and dragging will pan the camera.
- Holding *ALT* while dragging will zoom in and out.

Another navigation tool in the 3D view is the **Roam** tool, . This tool allows the camera to move in and out of the model at will. It has a higher learning curve but is the most flexible viewing tool because it allows the camera to be placed anywhere in the model. This tool can work in three different modes. In the first two modes, the movement speed of the camera can be changed by holding *CTRL* and spinning the mouse wheel up or down. Spin it up to increase the speed and

down to decrease the speed.

### Mouse+Keyboard Mode

Click and drag the left mouse button to look around. The camera will remain stationary. Dragging the mouse up will make the camera look up, dragging it down will look down, and dragging it left and right makes it look left and right. Holding *CTRL* while dragging will make the camera move forward and backward in the XY plane, and holding *ALT* while dragging will make the camera move up and down along the Z axis.

### WSAD Mode


This mode mimics the controls in video games. Click and drag to look around as in the previous mode. Press the *W* key to move forward along the viewing direction, *S* to move backward, *A* to move to the left, and *D* to move to the right. *SPACE* moves the camera up along the Z-axis, and *C* moves the camera down. Pressing these keys moves the camera at a fixed speed. Holding *SHIFT* while pressing the keys doubles the speed.

### Mouse-only Mode

This mode smoothly animates the camera to any location using only the mouse. To do so, press and release the middle mouse button. The cursor will disappear, and the tool will enter **Mouse-only Mode**. In this mode, moving the mouse will look around as in the other modes except that the mouse button doesn't have to be pressed. Pressing and dragging the left mouse button will move the camera in the XY plane. The further the mouse is moved from its button press, the faster the camera will move. This can simulate the effect of accelerating the camera. Doing the same with the middle mouse button will cause the camera to move forward/backward in the XY plane with changes along the Y mouse axis and turn left/right with changes along the X mouse axis. Pressing and dragging the right mouse button will move the camera along the Z axis in the same manner. To exit **Roam Mode**, press and release the middle mouse button again or press *ESC* on the keyboard.

<b>NOTE</b>	The <b>WSAD</b> keys may also be used in this mode to move the camera.
-------------	--


The other navigation tools include a **Pan/Drag** tool, which moves the camera left and right and up and down, a zoom tool, which zooms in and out of the model while click-dragging, and a zoom box tool, which allows a box to be drawn that specifies the zoom extents.

Pathfinder can also be navigated while using the **Selection/Manipulation** tool, . To Orbit the camera while in perspective view, use a right-click and drag combination. Similarly, use a middle-click and drag to Pan in perspective view.

### 2.2.2. Navigating the 2D view


Navigation in the 2D view is simpler than in the 3D view. The selection tool not only allows objects to be selected if single-clicked, but it allows the view to be panned by middle or right-clicking and dragging, and the view to be zoomed by using the scroll wheel. The drag and zoom tools are also separated into separate tools for convenience.

### 2.2.3. Resetting the view

At any time, the camera can be reset by pressing `CTRL+R` on the keyboard, or selecting **Reset All** tool, . This will cause the entire model to be visible in the current view. For all navigation tools but the Roam tool, reset will make the camera look down the negative Z axis at the model. For the roam tool, however, reset will make the camera look along the positive Y axis at the model.

The camera can also be reset to the current selection at any time by pressing `CTRL+E`. This will cause the camera to zoom in on the selected objects and the orbit tool to rotate about the center of their bounding sphere.

### 2.2.4. Filling the view

Very similar to resetting the camera, the view can be fit by pressing `F` on the keyboard or selecting the **Fill View** tool, . The difference between the **Fill View** and **Reset All** tools is that filling the screen does not change the view angle of the camera. Instead the camera will recenter/rezoom to fit the screen.

### 2.2.5. Drawing in the 3D and 2D views

Drawing can be performed in both the 3D view and the top 2D view. The 3D view allows the user to see the model from any angle, but most tools restrict drawing in the XY plane. The top view completely restricts drawing to the XY plane, but it also displays an optional snap grid. The snap grid size can be set under **Edit snap grid** in the **View** menu, and it can be turned off by deselecting **Show Snap Grid** in the **View** menu.

Drawing is performed in one of the two following modes:

#### Normal Mode

Single-click a drawing tool button on the left side of the view. Draw the object using the instructions in the appropriate section of the manual. When the object has been completed, the drawn object(s) will be selected and the view will revert to the previous navigation tool.

#### Sticky Mode

Double-click a drawing tool's button on the left panel before beginning to draw. When the object is completed, the same drawing tool will remain selected and more objects can be

drawn with the tool. To escape this mode, press *ESC* on the keyboard, and the previous navigation tool will be selected. A green dot on the tool's icon indicates that the tool is currently in sticky mode. Single-clicking the tool's icon again will turn off sticky mode but keep the tool selected.

At any time while drawing, the user can press *ESC*, which causes the current object to be cancelled and the previous navigation tool to be selected.

For each tool there are often two ways to create its object.

- Draw the object graphically using the mouse and keyboard.
- Interactively create the object by typing information such as coordinates, widths, etc., in the tool's property panel.

The property panel will update the graphical preview immediately to reflect changes in the input. This allows fine-grained control in creating the object. The individual drawing tools are discussed in [Chapter 3](#).

#### 2.2.5.1. Draw Tool Navigation

While using any of the draw tools, the mouse can still be used to zoom or pan the camera as follows:

##### Zoom

Use the scroll wheel on the mouse to zoom in or out.

##### Pan

Click+drag the middle mouse button to pan the camera.

### 2.2.6. Snapping

Snapping is one way to precisely draw and edit objects. It is the process of finding some element in the scene, such as a **vertex** or **edge** close to the cursor, and snapping the cursor to that element like a magnet.

In Pathfinder, snapping can be performed against objects in the model and orthographic constraints. The **2D View** additionally provides a sketch grid and polar (angle) constraints. If a snap point is found, an indicator dot, shown in [Figure 10](#), will appear at the snap point.



**Figure 10. Snap point indicator dot**

By default, snapping is enabled. It can be disabled by holding *ALT* on the keyboard while using a



drawing/editing tool.

#### 2.2.6.1. Sketch Grid Snapping (2D View Only)

Pathfinder provides a user-defined drawing grid, or sketch grid, in the 2D View. When a new model is created, the sketch grid is visible and can be snapped to in the 2D view. The default spacing for the divisions is **.5 m**, but can be changed by going to the **View** menu and clicking **Edit Snap Grid**. To disable grid snapping, on the **View** menu uncheck **Show Snap Grid**.

#### 2.2.6.2. Object Snapping

All objects displayed in the model can be snapped to when using the drawing/editing tools. There are three basic categories of geometry that can be snapped to on objects: **faces**, **edges**, and **vertices**. Objects can have any combination of types. If there are multiple types close to the cursor, Pathfinder will give vertices precedence over edges and edges precedence over faces.

#### 2.2.6.3. Constraint Snapping

Constraints are dynamic snapping lines that are only visible when the cursor is near them. They appear as infinite dotted lines that extend from the most recent relevant clicked point when using a multi-point tool.

Pathfinder contains two types of constraints:

##### Orthographic

These constraints allow the user to snap to a line parallel to the X, Y, or Z axis from the last relevant point. For instance, when using the **Copy/Move Objects** tool and after clicking the initial reference point, there will be three orthographic constraints extending from the reference point to aid in specifying an offset point or distance.

##### Polar (2D View Only)

These constraints are similar to orthographic constraints, but they are found at **15** degree increments from the current view's local X axis.

#### 2.2.6.4. Constraint Locking

If the cursor is currently snapping to a constraint, that constraint can be locked by holding *SHIFT* on the keyboard. While holding *SHIFT*, a second dotted line will extend from the cursor to the locked constraint (the first dotted line). This is useful for lining up objects along a constraint with other objects.

#### 2.2.6.5. Snapping in Different Views

Snapping behavior may change depending on which modeling tool is selected and whether a 2D

or the 3D view is active. If the 3D view is active, the cursor will snap to a 3D coordinate, which may then be further restricted by the tool. For instance, the occupant dropper tool (see [Section 5.4.1](#)) may first snap to a 3D coordinate and then project that coordinate down along Z onto the nearest room. If a 2D view is active, the cursor may first be snapped to a 3D coordinate, and then projected onto a drawing plane that is parallel to the camera's view plane. Most tools will indicate the snapped position in the status bar at the bottom of the 2D or 3D view.

#### 2.2.6.6. Asynchronous Snapping

Snapping may be a slow operation in complex models. In these cases, **asynchronous snapping** is used to keep the cursor and application responsive while the snapping operation takes place in the background. During **asynchronous snapping**, a wait cursor will appear at the cursor crosshairs while the snapping completes. While this takes place, either keep the cursor still to allow the current snapping operation to complete or move the cursor to abort the operation and snap to a different location. Asynchronous snapping can be disabled by unchecking **File → Preferences → Enable asynchronous snapping**. Note that if it is disabled, the cursor may briefly hang and the application will become unresponsive while these long snapping calculations take place.

## 2.3. View Options

Pathfinder provides a variety of view options for displaying both navigation geometry and imported geometry that can also aid with drawing. This includes options for rendering geometry, displaying agents, coloring rooms, and setting the transparency of rooms.

### 2.3.1. Render Options

In the toolbar above the properties window in the 2D and 3D views, there is a drop-down as shown in [Figure 11](#) and buttons as shown in [Figure 12](#) that control how geometry is rendered.

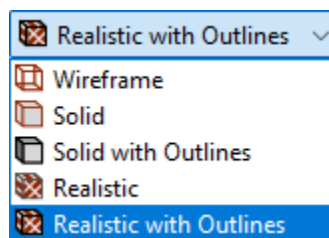


Figure 11. View drop-down

#### Wireframe

This renders geometry with only outlines.

## Solid

This renders geometry without **Materials**.

## Solid with Outlines

This renders geometry without **Materials**, but with outlines.

## Realistic

This renders geometry with **Materials**.

## Realistic with Outlines

This renders geometry with **Materials** and outlines.



### Figure 12. Render options

From left to right, the buttons are:

### Show Navigation Geometry

This toggles the visibility of all the navigation geometry. It does not affect anything else (including imported geometry and occupants).

### Show Imported Geometry

This toggles the visibility of all imported 3D geometry.

### Show Occupants

This toggles the visibility of occupants.

## Show Behaviors

This toggles the visibility of behavior objects, such as waypoints.

### Show Occupant Sources

This toggles the visibility of occupant sources.

### Show Measurement Regions

This toggles the visibility of measurement regions.

## Show View Objects

This toggles the visibility of view objects.

## Show Triggers

Toggles the visibility of trigger objects.

### Show Occ Targets

Toggles the visibility of Occupant Target objects (see [Chapter 9](#)).

### Show Obstacles

Toggles the visibility of obstacles (see [Section 3.4](#)).

## 2.3.2. Occupant Display

Occupants can be displayed using a number of options. They can be viewed as simple shapes, including disks and cylinders. They can also be displayed as the artist's mannequin or as their respective human avatars specified in their profiles. These options are available under **View** menu and **Occupant Display** submenu.

When occupants are not viewed as people, they can also be colored in several ways, available under **View** → **Occupant Color**:

#### Default

If an occupant has an individually-set color, the occupant is colored with this; otherwise, the occupant's profile color is used.

#### By Movement Group

If the occupant is part of a movement group, this colors all the occupants in their group the same color. The group color is set in the properties panel for the selected movement group. If the occupant is not in a movement group, their default color is used.

#### By Movement Group Template

If the occupant is part of a movement group that was created from a movement group template, this option colors the occupant using the color of the movement group template; otherwise, their default color is used.

#### By Behavior

Colors the occupant using the color specified in the property panel of the occupant's behavior.

#### By Profile

Colors the occupant using the occupant's profile color, even if the occupant has an individually set color.

## 2.3.3. Coloring Rooms

Rooms can be colored in a variety of ways. All coloring options are available under the **View** menu and **Color Rooms** submenu. Rooms can be colored by the following options:

### Default

Displays each room with a unique color.

### By Occupant Density



Colors the room with a color scale based on the occupant density of the room, defined as  $\text{number\_of\_occupants}/\text{room\_area}$ . Red indicates high density and blue indicates low density.

### Mixed

Rooms are colored by occupant density if they contain agents; otherwise, they are colored with their unique colors.

### By Room Type

Rooms are colored according to the type of room.

-  Indicates a normal room
-  Indicates a *Refuge Area*

## 2.3.4. Room Opacity

Sometimes it is useful to be able to see through rooms and stairways, such as when drawing on top of an imported background image. To change the opacity of a set of components, select them and in the property panel, change the opacity. Opacity settings will carry through to 3D results visualization.

## 2.4. Model Organization with Groups

The main method of organization in Pathfinder is to use groups. In every model there are already some implicit groups that cannot be modified, including **Views**, **Imported Geometry**, **Profiles**, **Vehicle Shapes**, **Assisted Evacuation Teams**, **Behaviors**, **Occupant Sources**, **Occupants**, **Movement Groups**, **Movement Group Templates**, **Elevators**, **Measurement Regions**, and **Floors** as shown in [Figure 13](#). Sub-groups can be created to further organize the model as discussed in the following sections.

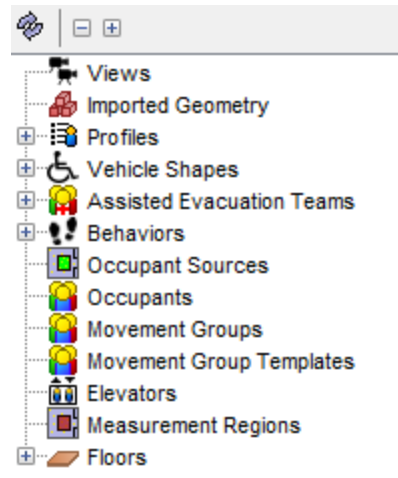


Figure 13. Predefined groups

### 2.4.1. Creating sub-groups

Sub-groups can be created under all groups (floors are discussed in [Section 3.1.](#)) Groups can also be created in other sub-groups.

To create a new group:

1. Right click the desired parent group in the navigation view
2. Select **New Group** or select **New Group Node** from the **Model** menu.
3. A dialog will display allowing the user to select the parent group (which will automatically be selected if performed from the right-click menu) and a name for the new group.
4. Click "OK" to create the new group.

### 2.4.2. Changing groups

An object can be moved from one group to another at any time.

To change an object's group:

1. Drag the object to the desired group in the Navigation View or right click the object and select **Change Group**.
2. This will show a dialog that will allow the user to choose the new group. The options shown for the new group will only be valid groups for which the group can be changed.
3. Select "OK" to change the group.

## 2.5. Keyboard Shortcuts

By default, Pathfinder uses a variety of keyboard shortcuts that are standard to Windows and Java applications. To accelerate model creation, shortcuts can be added or changed to in a variety of ways.

Some keyboard shortcuts are used by Java UI components. If a shortcut does not result in the expected behavior, it may be in direct conflict with a preexisting Java shortcut. It is best practice to avoid these conflicts (Default Swing key bindings ([IBM n.d.](#))).

### 2.5.1. Keyboard Shortcut Editor

The Keyboard Shortcut Editor Window can be found in the File → Preferences dialog and clicking Edit Actions and tools can be bound to a combination of modifier keys (*ALT*, *CTRL*, *SHIFT*, etc.) and other key presses.

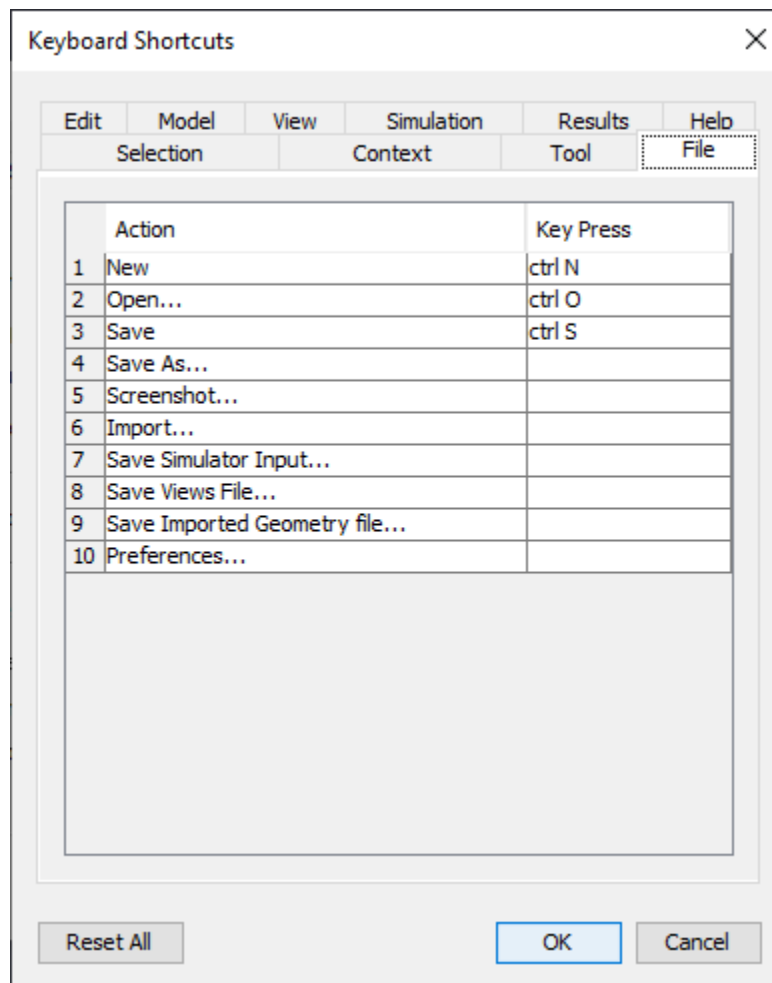


Figure 14. Keyboard Shortcuts Dialog

The dialog is split into sections similar to those used in the Pathfinder toolbar menus. There are

additional tabs for shortcuts related to tool activation, object selection, and context sensitive actions.

To change a shortcut, click on cell in the Key Press column and a window launches ([Figure 15](#)) with four different options.

**Type an input**

Input the desired keybinding to map it to the action.

**Clear**

Remove the current keybinding from the action.

**Reset**

Assign the default keybinding to the action.

**Cancel**

Exit the editor window without making any changes.

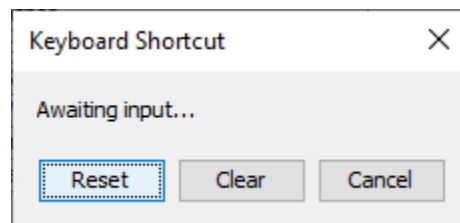


Figure 15. Keyboard Shortcuts Listener

## 2.6. Object Sets and Discrete Distributions

There are times when a set of objects might need to be selected or a distribution of discrete items specified. Pathfinder provides two dialogs that can be used to do so.

### 2.6.1. Object Sets

When specifying a set of items, such as the doors an occupant is allowed to use, the [Figure 16](#) is shown.



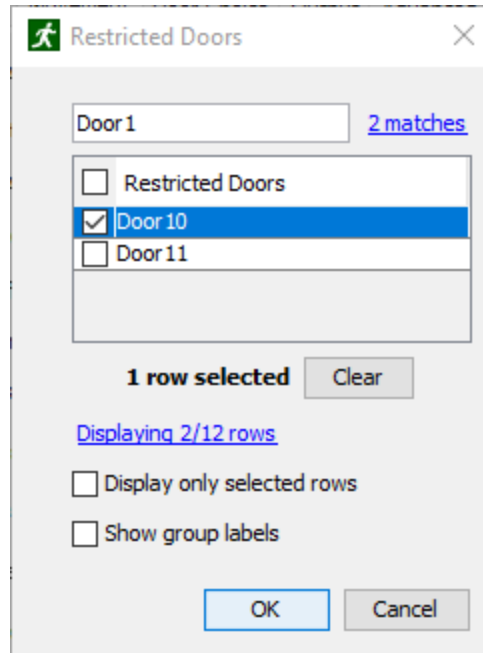


Figure 16. Set Chooser Dialog

The list in this dialog shows the items in the model that can be chosen. Click the checkbox next to the item in the list to select the item. Alternately, click the checkbox in the header of this list to select or deselect all items currently displayed in the list.

Click **OK** to commit the selected items.

Sometimes, the list of items might be quite long, making it difficult to quickly find the desired items. There are several ways to limit the displayed items, however. At the top of the dialog is a search field. Typing in this field will limit the list of items to only those that partially contain the entered text. To match the name of an object exactly, surround the text with quotes. For example "Door01" will show only the doors that have exactly the name, Door01. This search field also allows wildcards including the following:

?

Matches any single character

\*

Matches any number of characters

For example, typing Door?1 would find objects with the following names: Door01, Door51a, Exit Door91 (south), etc. It would not, however, find the following: Door001, Door9221, etc. Typing Door\*1, however, would find those.

Another way to limit the displayed item is to check the box next to **Show only selected rows**. This will only show items that have already been selected. By default, when the Set Chooser dialog

opens, if there are selected items in a long list, **Show only selected rows** will be automatically selected so it's easy to see the current selection.

**NOTE**

When the list of items has been filtered either by the search field or **Show only selected rows**, the number of displayed items is shown under the list. Clicking this label will reset the filters so that all items are displayed.

The following additional options are also available:

**Clear**

Clicking this button will clear the selection, including both displayed and hidden rows.

**Show group labels**

If checked, the group labels for each item is shown in the list. For example, "Door01" might become "Floor 0.0 m → Door01".

## 2.6.2. Discrete Distributions

Pathfinder also allows discrete distributions of objects in certain cases. For example, the [Change Behavior](#) action allows occupants to change their behavior by choosing a random behavior from a discrete distribution. Occupants might have a 30% chance of choosing [Behavior01](#) and 70% chance of choosing [Behavior02](#). The [Figure 17](#) allows these distributions to be specified.

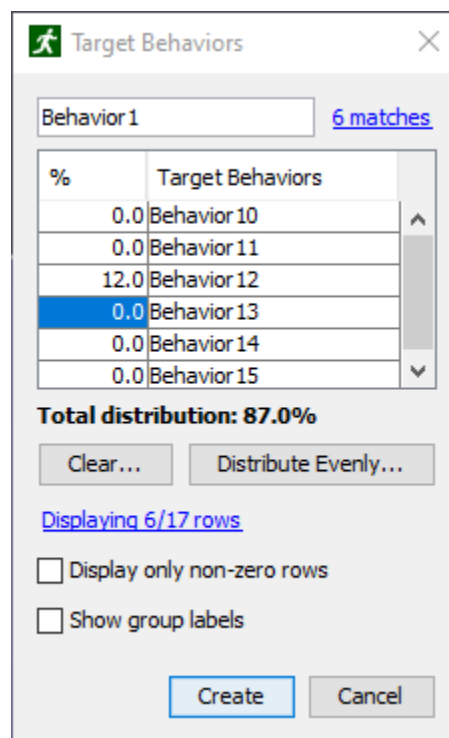


Figure 17. Discrete Distribution Dialog

In this dialog, the probability of choosing a value can be entered into the first column. The second column shows the name of each available item. The sum of the values in first column must add to **100%**.

As in the **Set Chooser** dialog described above, the displayed items can be filtered either by typing text into the search field at the top of the dialog or by clicking **Display only non-zero rows**, which only shows rows that have non-zero probabilities. In addition, **Show group labels** can be checked to show the names of the groups next to the item names in the list, such as "BehaviorGroup01 → Behavior01".

The distribution dialog has some additional actions that can be used to clear values or distribute probabilities evenly among several items.

To set the probability of multiple items to **0.0%**, click the **Clear** button and choose from one of the following options in the popup menu:

#### **Clear selected rows**

Clears the values for all highlighted rows.

#### **Clear displayed rows**

Clears the values for all rows currently in the list but not hidden rows.

#### **Clear all rows**

Clears the values for all rows, including hidden rows that do not match the current search field.

To distribute probability evenly across multiple rows, click the **Distribute Evenly** button and choose one of the following option in the popup menu:

#### **Distribute remaining X% to selected rows**

This is only available if the current total distribution is less than **100%**. It will distribute all remaining values to those currently highlighted in the list. For instance, if the current total distribution is **85%**, and there are **3** highlighted items in the list, this action will add  $(100 - 85)/3 = 5\%$  to each selected item's current probability, ensuring that the end total distribution is **100%**.

#### **Distribute remaining X% to displayed rows**

Distributes the remaining values evenly across all displayed rows.

#### **Distribute 100% to selected rows**

Clears all rows (including hidden rows) and then evenly distributes **100%** across only those that are highlighted.

**Distribute 100% to displayed rows**

Clears all rows (including hidden rows) and then evenly distributes 100% across those currently displayed in the list.

**Distribute 100% to all rows**

Clears all rows (including hidden rows) and then evenly distributes 100% across all rows, including hidden rows.

## Chapter 3. Creating Movement Space

Pathfinder is built on the idea of creating floor space on which occupants can walk. Every navigation component drawn in Pathfinder is some piece of flooring that can be travelled on, which can range from floors, to doorways, to stairs. Obstructions exist as holes in the floor.

The main egress components include rooms, which are empty floor spaces bounded by walls, doors, which connect rooms on the same level, stairs/ramps, which connect rooms on different levels, and elevators, which connect multiple levels.

### Rooms

Can have any polygonal shape, and can never overlap on the same level.

### Doors

Can be either thick if they are occupying a doorway (the area between two rooms) or thin if they are simply connecting two touching rooms.

### Stairs/Ramps

Are always rectangular and implicitly contain a thin door on each end to connect the adjacent rooms.

### Elevators

Can be any shape and can travel in any direction.

To organize egress components, Pathfinder provides the concept of floors, which group together components at different Z locations.

#### NOTE

When working with imported files, such as IFC, DWG, FBX, or glTF/GLB, Pathfinder provides tools that can simplify the creation of the movement space. In some cases, such as with IFC files, Pathfinder can even create most or all of the movement space automatically, see [Section 4.3](#).

## 3.1. Floors

Floors are the primary method of organization in Pathfinder. At their most basic level, they are simply groups in which rooms, doors, stairs, ramps, and exits can be placed, but they also control the drawing plane for most tools and filtering of imported geometry.

In every Pathfinder model, at least one floor must exist, and at any given time, there is one active floor. Whenever any navigation object is drawn, it will either be placed in the active floor or a subgroup of the active floor.

By default when a new model is started, there is one floor at  $Z=0$ , and additional floors are either created automatically depending on where the geometry is drawn or manually created. In addition, new navigation components are automatically sorted into the appropriate floor when drawn.

### 3.1.1. Automatically Creating Floors

When nothing is selected in the model, the **Floor Creation** panel is shown, as in [Figure 18](#). This panel controls the automatic creation of floors and automatic sorting of new objects into floors.

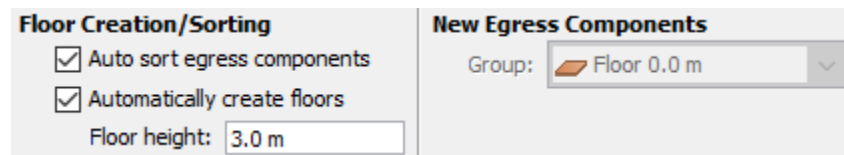


Figure 18. Floor Creation Panel

#### Auto sort egress components

If this is checked, navigation components are automatically sorted into the appropriate floor when created or modified; if this is unchecked, new navigation components are placed in the group specified under **New Egress Components** and remain in this group until manually moved.

#### Automatically create floors

If this is checked, floors are automatically created as navigation components are created and modified.

#### Floor height

This specifies the height at which new floors are automatically created. If a navigation component is created or moved to a location that is at least this distance from the previous floor, a new floor will be created at a multiple of this distance from the previous floor.

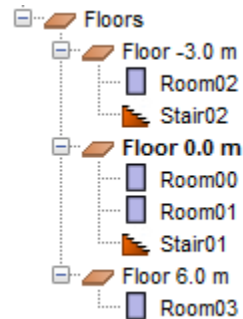
#### Group

If **Auto sort egress components** is unchecked, this drop-down specifies the group/floor for new navigation components.

The following scenario demonstrates how objects are organized when auto-sort and auto-floor-creation are enabled (organization of the model is shown in [Figure 19](#)):

1. A new model is created. The floor height is left at the default of 3 m.
2. "Room00" is drawn at  $Z=0$  m, and is auto-placed in "Floor 0.0 m".
3. "Room01" is drawn at  $Z=1.5$  m, and is auto-placed in "Floor 0.0 m".

4. "Stair01" is drawn connecting "Room00" to "Room01" and is auto-placed in "Floor 0.0 m".
5. "Room02" is drawn at Z=-1.5 m. A new floor, "Floor -3.0 m" is auto-created, and "Room02" is auto-placed in it.
6. "Stair02" is drawn connecting "Room02" and "Room00" and is auto-placed in "Floor -3.0 m".
7. "Room03" is drawn at Z=7.5 m.
8. A new floor, "Floor 6.0 m" is auto-created, and "Room03" is auto-placed in it.



**Figure 19. Auto floor creation and sorting**

In this example, only rooms and stairs were created. The floors were automatically created and the rooms and stairs were automatically sorted into the appropriate ones.

#### 3.1.1.1. Using Auto-sort with an Existing Model

Automatic floors can be created and components sorted into the floors by performing the following:

1. Open the model.
2. Clear the selection so that the **Floor Creation** panel is visible (Figure 18).
3. Ensure that the desired creation/sorting options are enabled and that the correct floor height for the model is set.
4. Select all the components that should be auto-sorted (if everything should be sorted, select the **Floors** top node).
5. Right-click the selection, and from the shortcut menu, click **Sort into Floors**.
6. The appropriate floors will be created and all selected items will be sorted into the appropriate floors.

#### NOTE

This will not delete any existing floors. If there are undesired existing floors, first move the navigation components out of them to another floor that will be kept, delete the undesired floors, and then perform **Sort into Floors**.

### 3.1.2. Manually creating floors

Floors can also be created manually at any time.

1. Click on the floor drop-down box above the Navigation View, and select **<Add New>** as shown in [Figure 20](#). A dialog will open asking for the floor location.
2. Enter the Z plane location ([Figure 21](#)) or click on a snap point in the 3D or 2D view and click okay. This Z plane will be used to update the drawing tools' working Z location when this floor is made active.

There are two other options in this dialog:

#### Set as active floor

The floor is set to be the active floor in the model after being created.

#### Resort existing egress components into new floor

All existing components that belong in the new floor will be moved to it.

By default, the name of the floor is **Floor [stem 332cc365a4987aacce0ead01b8bdcc0b]** where [stem 332cc365a4987aacce0ead01b8bdcc0b] is the working plane of the floor.

#### Manually Adding a New Floor

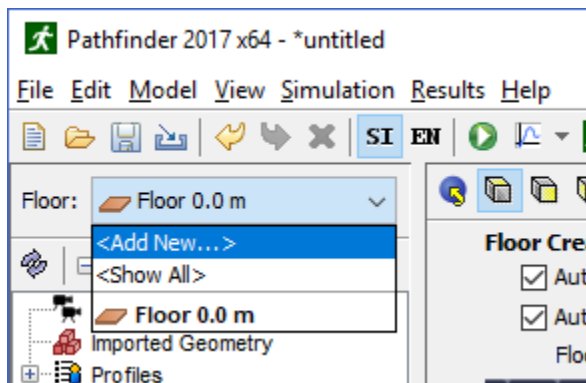


Figure 20. Adding a new floor

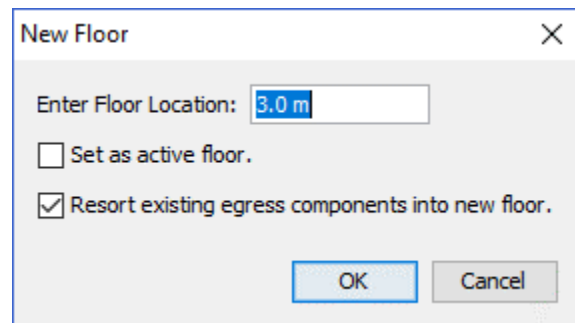


Figure 21. Floor Location Z plane

### 3.1.3. Changing the Active Floor

To change the active floor, click the floor drop-down box as shown in [Figure 20](#), and select the desired floor. This will make that floor active and all other floors non-active.

Whenever the active floor is changed, the following additional changes take place in the model:

- The floor, all objects in the floor group, and all occupants on the floor are set visible.



- All other floors, sub-objects of other floors, and occupants on those floors are set hidden.
- The working plane of the room and wall subtraction tools is set to the working plane of the floor.
- A clipping filter is applied to imported geometry so that only geometry within the Z clipping planes of the active floor is visible.

### 3.1.4. Showing All Floors

To show all floors click the floor drop-down box as shown in [Figure 20](#), and click <Show All>.

This will additionally show all occupants on the floors and all sub-objects of the floors groups, and it will set the import filter to the union of all the floors' filters.

### 3.1.5. Floor Properties

To edit a floor's properties, first select the desired floor. The property panel as shown in [Figure 22](#) will appear, showing the floor's name, its **Working Z** location, and the Z clipping planes (**Z Min Filter** and **Z Max Filter**) for imported 3D geometry.

It also shows some statistics of the floor including the total area of the floor (**Area**), number of people on the floor (**Pers**) and density of people. **Refuge Area** and **Speed Modifier** are discussed in [Section 3.2.5](#).

Floor 0.0 m	<input checked="" type="checkbox"/> Color: <span style="background-color: #800080; width: 20px; height: 10px; display: inline-block;"></span>	Working Z: 0.0 m	Area: 34.0 m²	<input type="checkbox"/> Refuge Area
<input checked="" type="checkbox"/> Visible	Opacity: 100.0 %	Z Min Filter: CURR_FLOOR	Pers: 0	Speed Modifier <span style="border: 1px solid #ccc; padding: 2px;">Always 1.0</span>
		Z Max Filter: NEXT_FLOOR	Density: 0.0 pers/m²	

Figure 22. Floor property panel

#### Working Z

Controls the plane on which new rooms and wall obstructions are drawn.

#### Z Min and Max Filters

Control the clipping planes of imported 3D geometry when the floor is visible. Anything below **Z Min** and above **Z Max** is clipped.

##### Z Min Filter

Can be a Z plane location or can have the special value, **CURR\_FLOOR**. If it is **CURR\_FLOOR**, then the clipping plane is set to the working Z location if there are any floors below this floor or [stem 1d5ba78bbbafd3226f371146bc348363] if there are no floors below.

##### Z Max Filter

Can be a Z plane location or can have the special value, **NEXT\_FLOOR**. If it is **NEXT\_FLOOR**, then

the Z plane is set to the working Z plane of the next higher floor if one exists, or [stem 701fa44621fd283e3f2c5468958859d8] if there are no higher floors.


## 3.2. Rooms

Rooms are open space on which occupants can freely travel. Each room is bounded on all sides by walls. Rooms can be drawn so that they touch each other, but an occupant can only travel between them if they are connected by a door. Only one room can occupy a given space at any time, so if one room is drawn overlapping another, the overlapping area will be subtracted from the old room and given to the new. Rooms can also be merged into one, separated into constituent parts, and have internal, thin boundaries drawn in them. These features are discussed in the following sections.

### 3.2.1. Adding New Rooms

Pathfinder provides two tools for adding new room geometry:

#### Polygonal Room Tool

The Polygonal Room tool  allows for the creation of complex shapes with any number of vertices ([Figure 23](#)).

Left-click anywhere in the model to set the first point, and continue left clicking to add more points to the polygon. When at least three points are defined, right-clicking will close the polygon and complete the shape.

Alternatively, **X,Y** coordinates can be entered from the keyboard with the **Add Point** and **Close Polygon** buttons from the property panel.

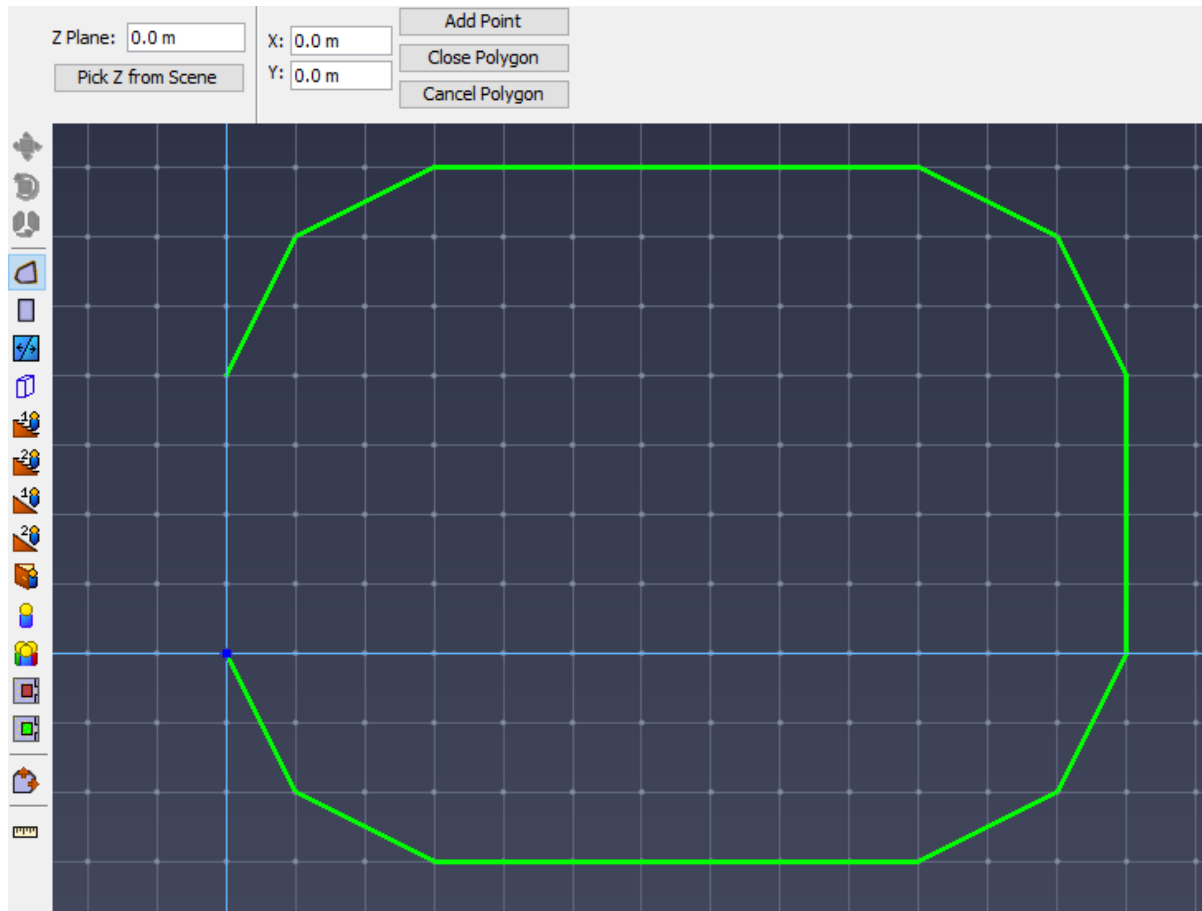


Figure 23. Drawing a room with the polygon tool

### Rectangular Room Tool

The Rectangular Room tool  creates simple rectangular geometry by left-clicking two points in the model ([Figure 24](#)).

The rectangular area can also be created by entering coordinates for two points in the property panel and clicking **Create**.



Figure 24. Drawing a room with the rectangle tool

In addition to creating new areas, both of these tools can be used on existing geometry to create negative areas. Creating new geometry over existing areas removes any interfering portion from those areas. The newly created geometry can then be deleted, leaving the negative space behind. This is discussed further in the section, Arbitrarily-shaped obstructions (desks, tables, etc.).

### 3.2.2. Drawing Plane

Each room tool draws on a particular Z plane that is specified in the property panel for the tool as shown in [Figure 25](#).

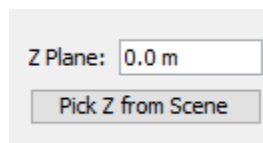


Figure 25. Drawing plane property

The Z plane can be specified either manually by typing the location into the **Z Plane** field or by picking the location from the **2D View** or **3D View** as follows:

1. Select one of the room drawing tools.

2. In the tool property panel, click **Pick Z from Scene**. This will clear the tool property panel while waiting for the user to click a location.
3. Click a point in either the **3D View** or a **2D View**. The tool property panel will return for the selected drawing tool, and the **Z Plane** field will be filled with the Z coordinate of the clicked location.

### 3.2.3. Splitting Rooms

Rooms can be split into two or more pieces using the [Thin Wall](#) tool. To do this, specify the two points such that they are on the outermost boundary of the room to be divided. The original geometry will be divided into two or more new rooms with that line as the boundary between them as shown in [Figure 27](#).

#### Dividing a Room

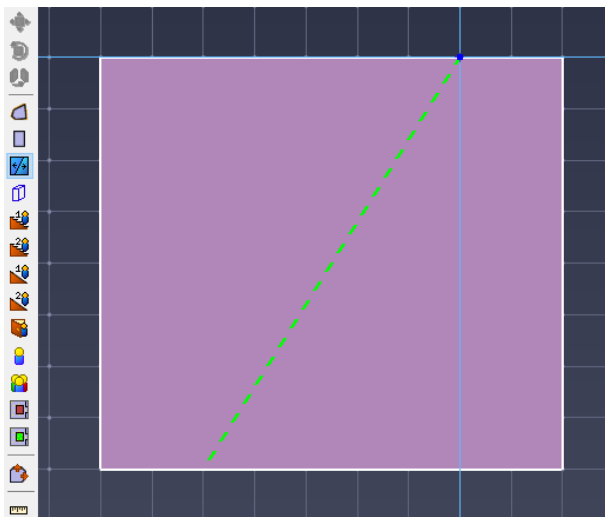


Figure 26. Original room

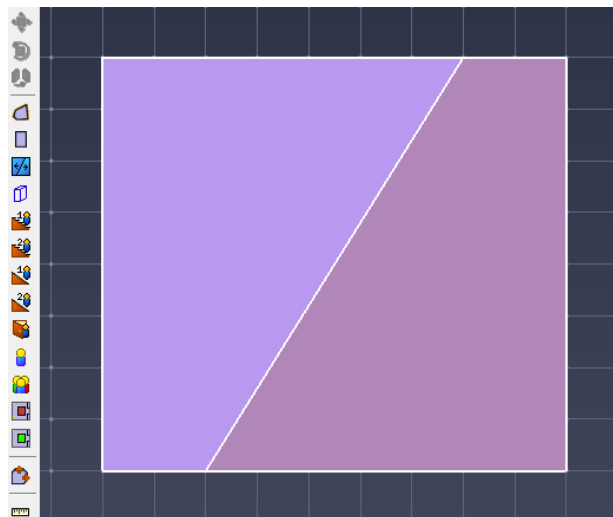


Figure 27. Two new rooms from the original

### 3.2.4. Separating and Merging Rooms

In addition to dividing rooms, Pathfinder has two additional means to aid in creating more complex room geometry.

#### Merge

The Merge command is used to join two or more rooms that share boundaries into one room. They can also be merged with stairs and ramps, but the stairs and ramps will be converted into rooms and will lose their stair/ramp properties.

To **Merge** rooms:

1. Select the neighboring rooms
2. Select **Merge** from the right-click menu, or select **Merge Rooms** from **Model** menu (Figure 28)

#### NOTE

Rooms can be merged even if they do not lie in the same plane as long as they share a common edge.

#### Merging rooms

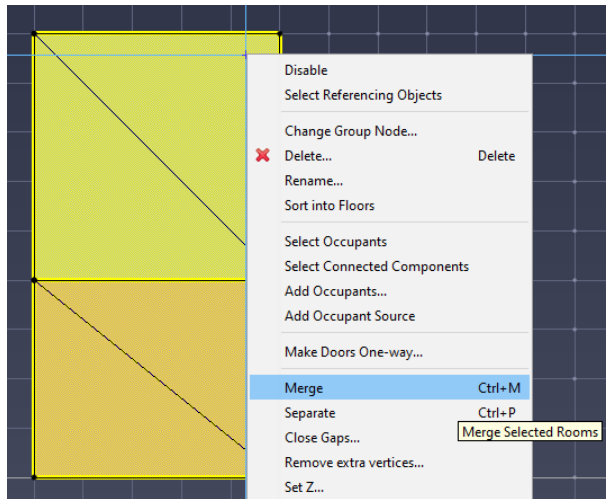


Figure 28. Two rooms to Merge

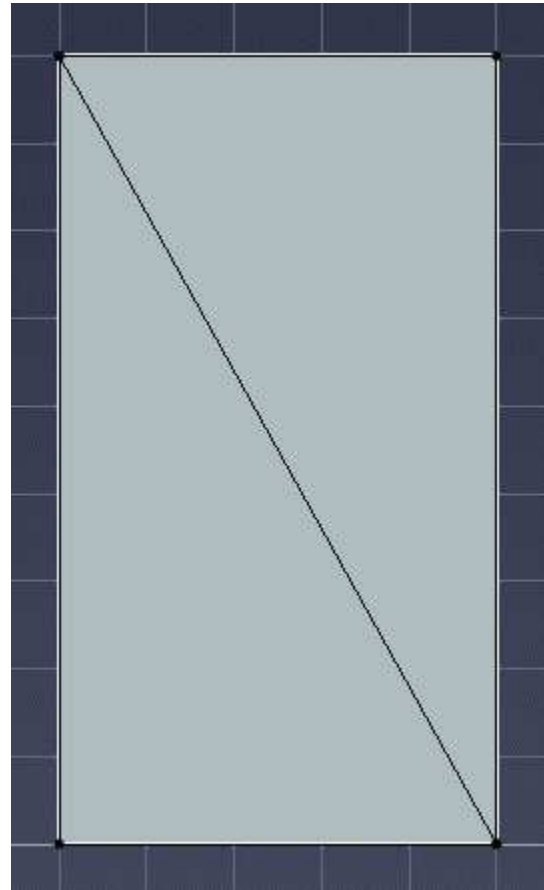


Figure 29. New merged room

#### Separate

The Separate command breaks a room into its constituent parts along any negative space that divides it (Figure 31).

To Separate a room:

1. Select the room to be separated
2. Select **Separate** from either the **Model** menu or the right-click menu (Figure 30).

## Separating a room

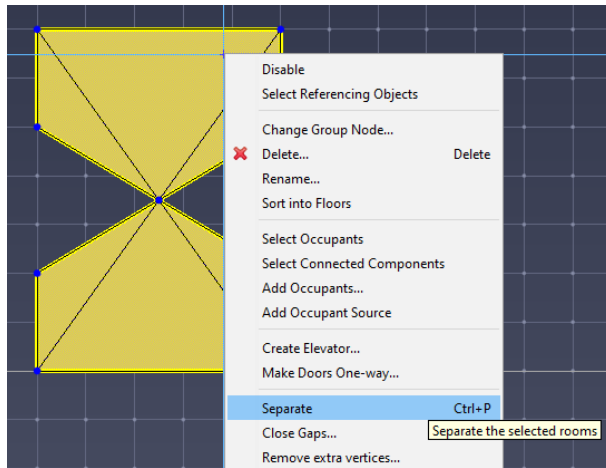


Figure 30. Single room shared point

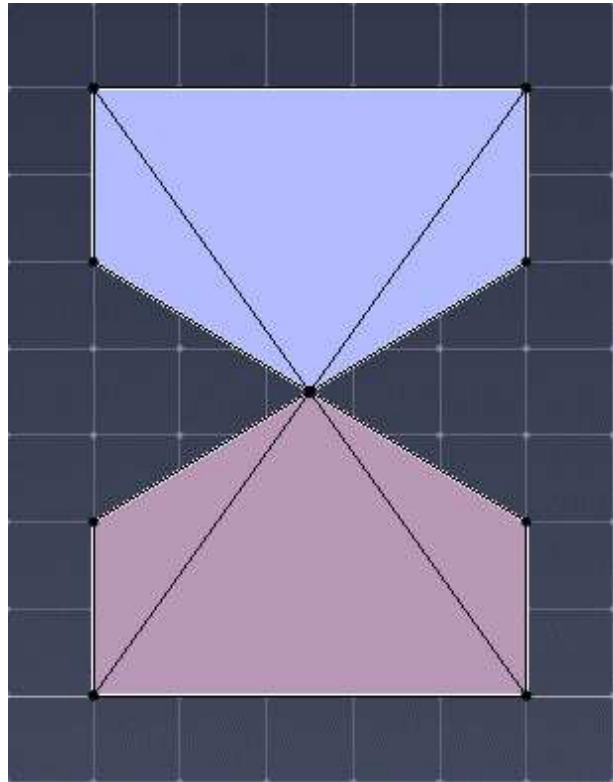


Figure 31. Two separated rooms

## 3.2.5. Room Properties

To view and edit room properties, select a room. Its properties will be displayed in the property panel as shown in Figure 32.

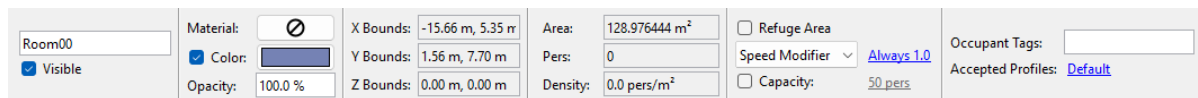


Figure 32. Room properties panel

### Name

An identifying name for room.

### Visible

Whether the room is currently visible. Uncheck to hide the room.

### Material

The material to display on the object when the **Realistic** option is selected. If a material is selected, the **Color** and **Opacity** settings below are only used if the **Diffuse/Albedo** coloring

option for the material is set to **From Object Color** (see [Section 4.3.7](#)). Clicking the material button will open the **Material Dialog** as shown in [Figure 97](#). In this dialog, the material can either be edited or a new material can be applied to the object by selecting one from the list on the left and clicking the **OK** button. To remove the reference to the material, select the **<No Material>** option from the material list.

**NOTE**

Because materials can be shared among multiple objects, editing a material applied to one object will also change the visual appearance of all objects referencing that material.

**Color**

The color of the object when there is no material or when using the **From Object Color** material option or when the **Solid** option is selected. If unchecked, a default room color is used.

**Opacity**

The opacity of the object when there is no material or when using the **From Object Color** material option or when the **Solid** option is selected. When used, making this value less than 100% will cause objects behind the room to become visible.

**X, Y, and Z Bounds**

The geometric bounds of the room.

**Area**

The geometric area of the room.

**Pers**

The number of occupants currently placed in the room.

**Density**

The number of occupants/area

**Refuge Area**

Marks the room as a **Refuge** area, which can then be specified in a list of rooms for the **Goto Refuge Rooms** action as discussed in [Section 5.3.3](#). When this option is selected, occupants will be tagged as "safe" whenever they are in the room. The "safe" times are reported in the **Occupant Summary** ([Section 16.8](#)) and **Occupant History** ([Section 16.9](#)) output files.

**Speed Modifier**

A time-variable factor that affects the speed of each occupant who travels in the room. When occupants travel in the room, their maximum speed is multiplied by this factor. This could be



used, for instance, to represent the effect of smoke on occupant speeds. By default, the modifier is 1.0 throughout the simulation. To change this, click the link. The **Edit Speed Modifier** dialog appears as shown in [Figure 33](#). In this figure, for instance, the room initially leaves occupant speeds unmodified, but over the first 40 seconds ramps occupant speeds down to 25% of their maximum values.

### **Speed Limit**

A time-variable speed limit that prevents occupants from traveling faster than the limit. This is set similarly to **Speed Modifier**.

### **Capacity**

A maximum capacity of the room, which can be specified using an occupant count or a density. Specifying room capacity is optional. If at any time during the simulation the room capacity reaches the limit, no occupants will enter the room until the room capacity decreases below the limit.

### **Occupant Tags**

A set of tags added to occupants when entering this room and removed from occupants when exiting it. Each tag is separated by a space (see [Section 5.8](#)).

### **Accepted Profiles**

A list of occupant profiles for which the occupants can enter this room (see [Section 2.6.1](#)). By default, all occupant profiles are accepted.

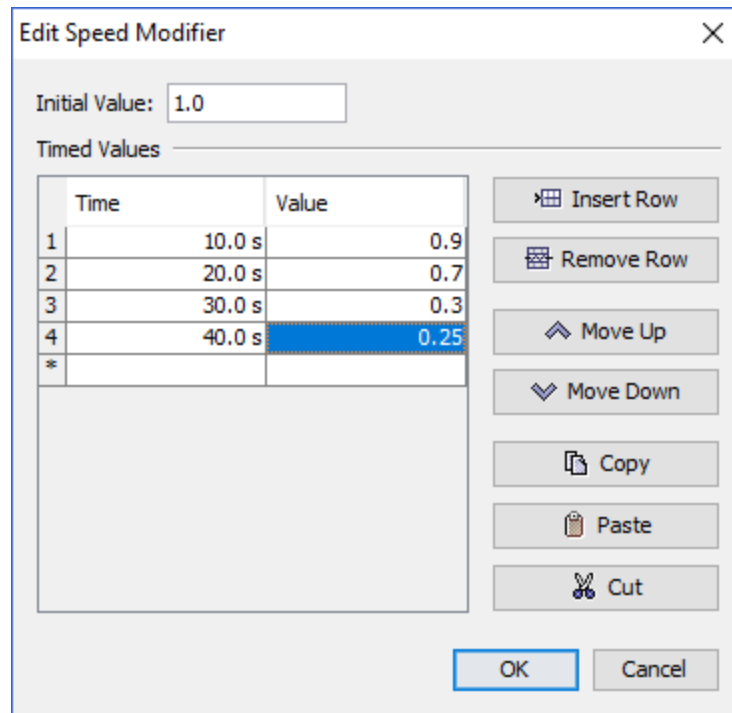


Figure 33. Edit Speed Modifier dialog

### 3.2.6. Preventing Room-Crossing

In some cases, such as modeling seating rows or shops in a mall, it may be desirable to only allow occupants to exit the room and not cross through it. This can be accomplished by making all the doors connected to the room one-way (see Doors) and ensuring that their directions point out of the room. Pathfinder provides a tool to make this easy. Instead of individually setting the one-way status of all the connecting doors, perform the following:

1. Select the room(s) that should not be crossable.
2. Right-click one of the rooms, and from the menu select **Make Doors One-way**. A dialog will appear as in [Figure 34](#).
3. From this dialog, choose whether occupants can only **Enter** or **Exit** the room. If any of the room's doors were already marked as one-way, another option will be provided to overwrite the one-way status of those doors.
4. Click **OK** in this dialog to make the doors one-way.

Pathfinder will automatically calculate the correct directions for the doors to make the rooms exit-only or enter-only.



Figure 34. Make Doors One-way dialog

**NOTE**

If any doors were shared between rooms in the selection, those doors will not be affected, as their direction would be ambiguous.

### 3.2.7. Mesh Optimization

Pathfinder performs automatic mesh optimization. An example of mesh optimization is removing multiple vertices in a straight line, or removing extra vertices left over after merging rooms. In some cases, it might be beneficial to disable mesh optimization.

To disable mesh optimization, select **Preferences** from the **File** menu, and uncheck **Optimize Navigation Geometry**.

**NOTE**

This setting will be maintained even after closing and reopening Pathfinder. If mesh optimization is being enabled, Pathfinder provides an option to optimize all navigation geometry at once.

## 3.3. Obstructions/Holes

In Pathfinder, obstructions that permanently block flow are modeled as holes in the navigation geometry. Holes can be created with an arbitrary polygonal shape or as thick walls.

**NOTE**

If you would like to model a transient obstruction that can slow down occupants or only block flow for a limited time, create an **Obstacle** instead (see [Section 3.4](#)).

### 3.3.1. Arbitrarily-Shaped Obstructions

To model an obstruction (e.g. an office desk or other standing obstacle) within a room, the subtractive property of rooms is used. This means that the room containing the obstruction must already exist.

To create the obstruction:

1. Select the **Add a Polygonal Room** tool or the **Add a Rectangular Room** tool and draw the shape and location of the obstructed area. This will subtract the area from the old room and create a new room.
2. Next, delete the new room. A hole is left in the old room in the shape of the obstruction. This process is shown in [Creating an obstruction](#).

### Creating an obstruction

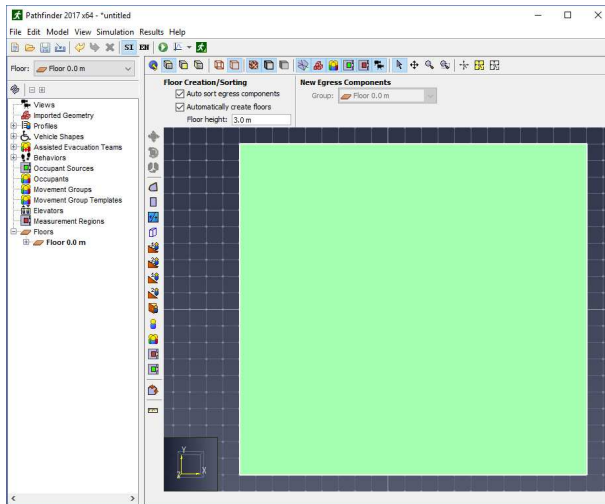


Figure 35. Start with an existing room

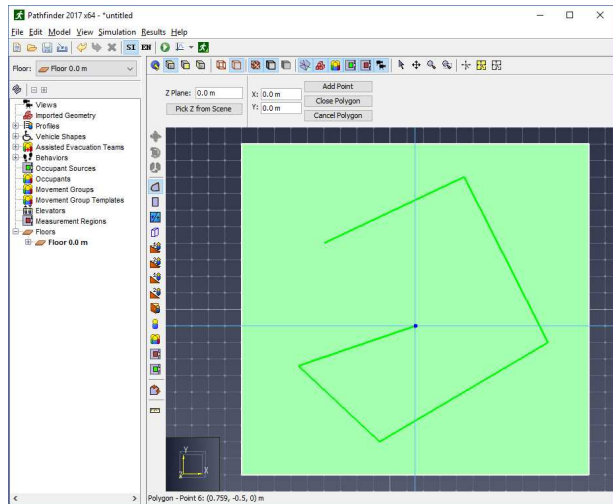


Figure 36. Draw the obstruction shape

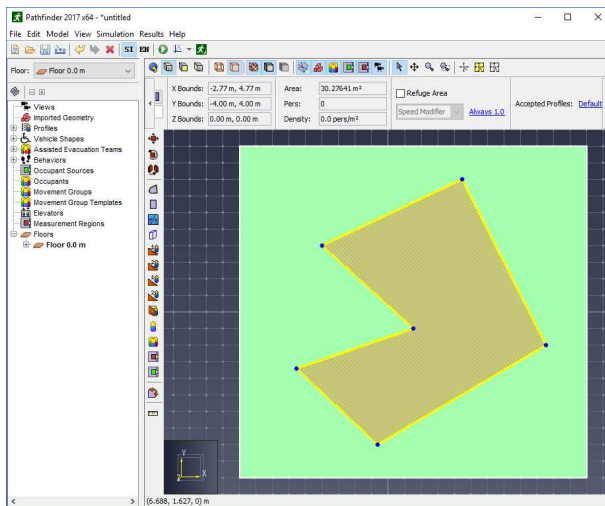


Figure 37. Delete the new room

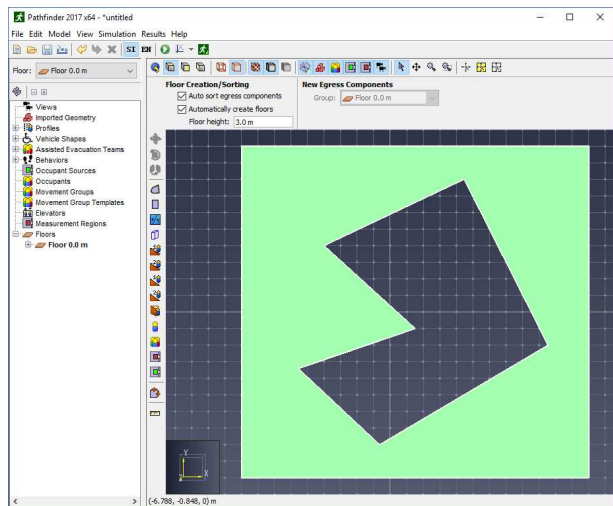


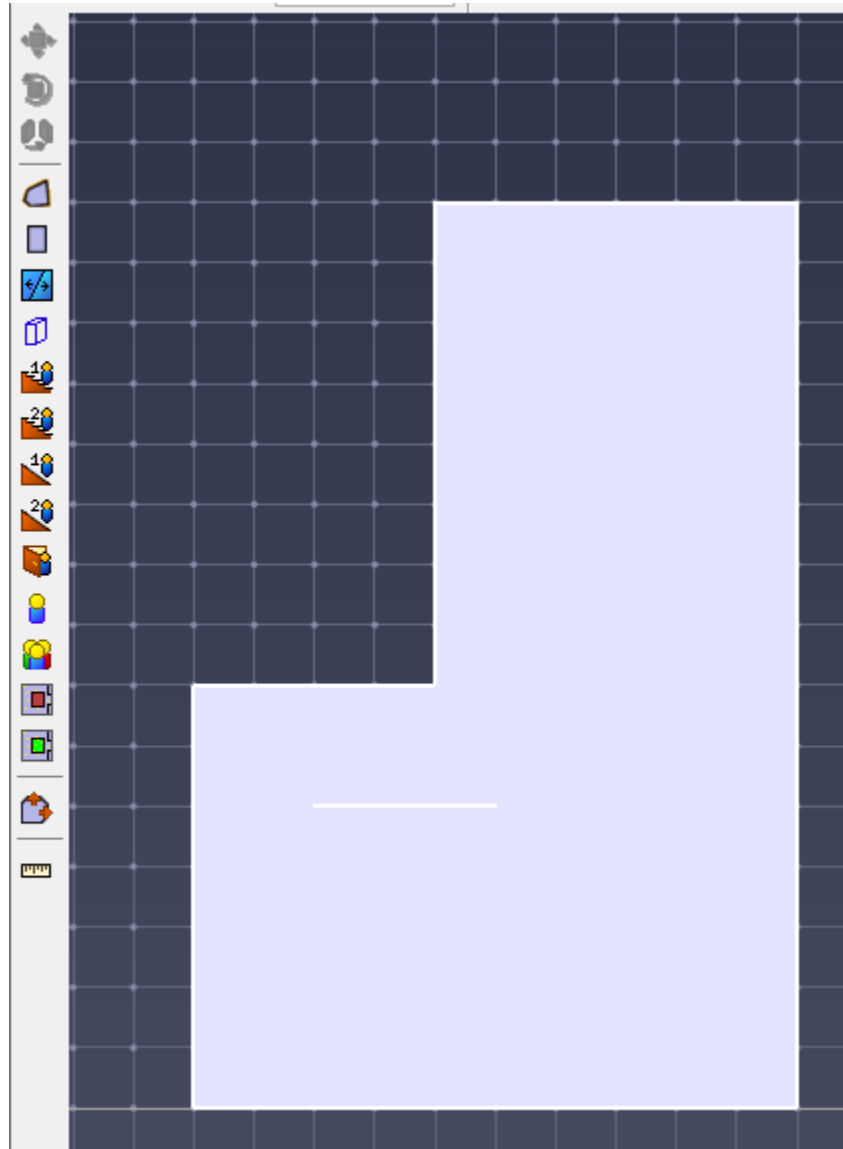
Figure 38. The cut out area obstructs movement

## 3.3.2. Thin Walls

Thin, internal walls or boundaries can be added to rooms with the Thin Wall tool .

To use this tool, click two points in the model as shown in [Figure 39](#). Pathfinder will attempt to

connect these two points with an internal boundary edge.



**Figure 39. Adding a thin wall to a room**

**NOTE**

In some circumstances, Pathfinder may have trouble connecting the two points. If this occurs, try limiting the two points to one room or to the number of room boundaries crossed.

### 3.3.3. Thick Walls

The wall tool  is used to make rectangular obstructions in existing geometry ([Figure 40](#)).

To make a thick wall:

1. Click on the wall tool.
2. Enter the desired wall width in the property panel.
3. Click or click-drag the two points the wall is to pass through. Holding the shift key will switch between alignment left and right of the defining line.

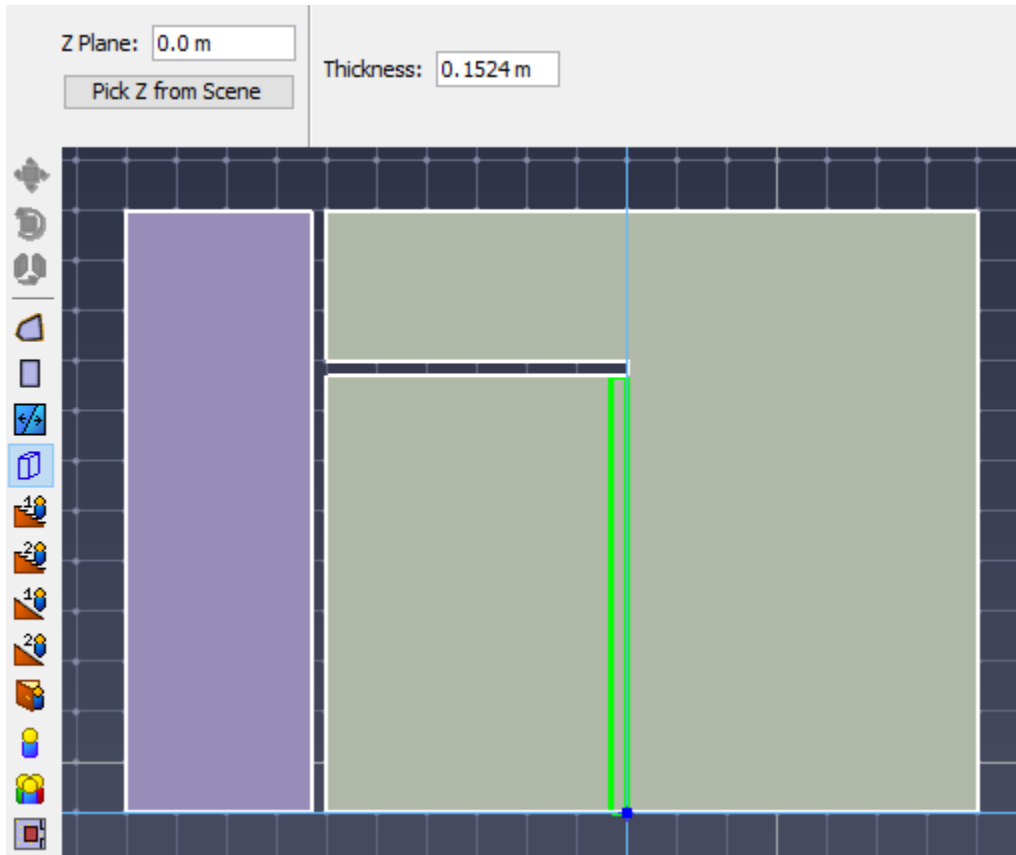


Figure 40. Drawing thick walls

**NOTE**

The thick wall tool is not as robust as the thin wall tool. It operates by subtracting area from any room that lies in its specified Z plane, whereas the thin wall tool will follow the slope of multiple rooms to connect two points.

## 3.4. Obstacles

Pathfinder **Obstacles** are objects that represent dynamic changes in the environment that might cause occupants to slow down in a portion of a room or even re-route, depending on the severity and size of the obstacle. They are applied as patches on the navigation mesh that override the current speed modifier in the affected rooms (see [Section 3.2.5](#)). Some uses of obstacles include, but are not limited to the following:

- A barrier that is introduced mid-simulation, such as a crowd-control device.

- A trip hazard that is introduced or removed during the simulation, such as luggage.
- An environmental hazard, such as smoke or fire.

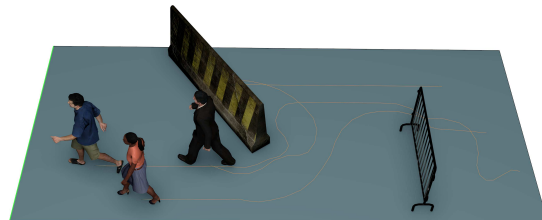
**NOTE**

When coupled with an FDS simulation as discussed in [Section 4.4](#), occupants can additionally be slowed down dynamically based on the [speed in smoke](#). An additional obstacle would currently be required to re-route occupants to another location, if desired.

The following images show occupants waiting for one obstacle to be removed and then re-routing after another is introduced later.



**Figure 41. Occupants waiting for a barricade obstacle to be removed**




**Figure 42. Occupants re-routing after another barricade is introduced**

### 3.4.1. Creating Obstacles

Obstacles can be created several ways, including by using the Obstacle drawing tool and by converting an imported CAD object into an obstacle.

#### 3.4.1.1. Creating axis-aligned obstacles


To define the obstacle as an axis-aligned block using the Obstacle drawing tool, perform the following:

1. From the toolbar on the left of the 3D or 2D view, select the **Add an Obstacle** tool .
2. Click and drag the left mouse button to define the shape of the obstacle.

If the selected points lie in the same Z plane, the resulting obstacle will appear as a 3D box that extends slightly above and below the selected Z plane. This is the **Search Volume** of the obstacle. It is intersected with the navigation mesh to determine which portion of the mesh should be categorized as part of the obstacle. The intersection portion is represented as a solid white polygon, while the **Search Volume** is represented as a translucent box as shown in [Figure 43](#).

### 3.4.1.2. Creating polygonal obstacles

An obstacle can also be drawn as a polygon. To do so, perform the following:

1. Select the **Add an Obstacle** tool .
2. With the left mouse button, click several points defining the shape of the polygon.
3. Right-click to connect the last clicked point to the first and finish the obstacle.

If the all the clicked points were on room faces that share the same plane, that plane is used as the basis for the **Search Volume**. The search volume appears as a prism, aligned with the plane and extending slightly above and below the plane as shown in [Figure 44](#).

If the clicked points are in faces that lie in different planes, the resulting **Search Volume** prism is aligned with the plane that maximizes the resulting area of the polygon when the points are projected onto the plane. If the drawing tool selects the wrong plane, right-click the obstacle and select **Align Obstacle with Plane**, which opens the **Set Normal** dialog. The correct plane can then be entered into the dialog or can be clicked in the 2D or 3D view.

#### NOTE

When using the drawing tool, if the selected points are on room faces with different planes, the resulting **Search Volume** is extended to include all the selected points in 3D space.

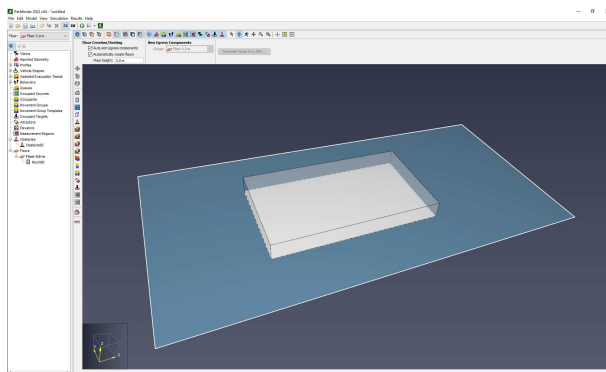


Figure 43. Obstacle drawn as an axis-aligned block

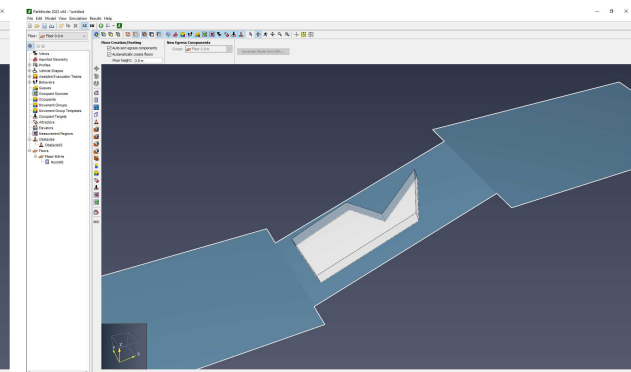


Figure 44. Obstacle drawn as a polygon on a sloped surface

### 3.4.1.3. Converting imported CAD objects into obstacles

Obstacles can also be created from imported CAD objects. This can be useful for visualizing the obstacle in a more realistic manner in Results and for automatically determining the **Search Volume** from the shape of the CAD geometry. See [Figure 42](#) for examples of CAD obstacles.

To create an obstacle directly from an imported CAD object, perform the following:



1. Import the CAD file as described in [Section 4.2](#).
2. Before extracting navigation components as described in [Section 4.3](#), select the CAD objects that will represent obstacles. If multiple objects should represent a single obstacle, select only the objects that will become the obstacle.
3. Right-click the selected objects and select **Convert to Transient Obstacle**.
4. If multiple objects were selected, a prompt will ask whether to merge the selected objects into a single obstacle. Choose the desired answer. If the CAD objects are not merged, an obstacle will be created for each selected object.
5. Choose whether to delete the source objects. If deleted, the original objects will be effectively converted into obstacles. If not deleted, the original CAD objects will remain, and their geometry and display properties will be copied to the resulting obstacles.

The obstacles are then added to the **Obstacles** group in the Navigation View, with their group hierarchies mirroring that of the original imported CAD objects.

By default, the obstacles' **Search Volumes** will be linked to their CAD geometry as described in [Section 3.4.2](#) and are hidden from view. To view them, under the **View** menu, select **Show Imported Obstacle Search Area**. The images below show the search volumes hidden and visible. The area where the search volumes intersect the navigation mesh are always visible as long as the source obstacle is visible.

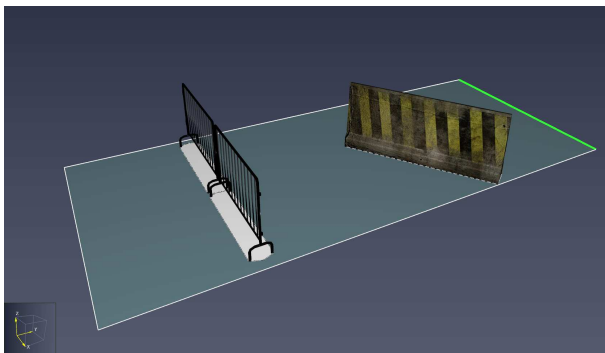


Figure 45. CAD Obstacle with search volumes hidden

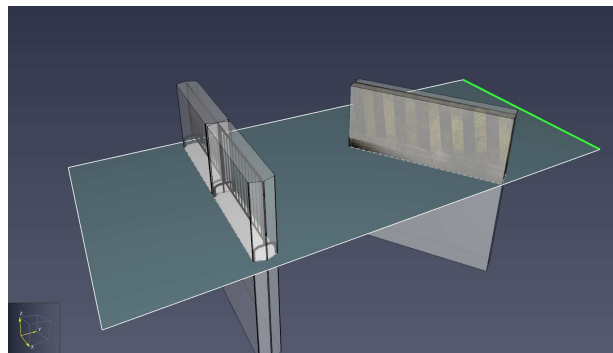



Figure 46. CAD obstacle with search volumes visible

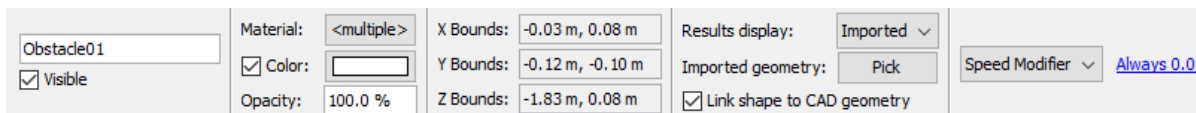
#### 3.4.1.4. Assigning imported CAD geometry to an existing obstacle

CAD obstacles can also be created from existing obstacles. This might be useful if a custom **Search Volume** shape is desired that does not match the automatic volume chosen by linking it to the CAD geometry. It might also be useful if obstacles are created before the CAD objects are imported. To create a CAD obstacle in this manner, perform the following:

1. Import the CAD file as described in [Section 4.2](#).
2. Right-click all imported CAD objects that you want to become obstacles, and select **Exclude from room extraction**. This ensures these objects will not become permanent obstructions during room extraction.
3. If multiple CAD objects will be represented as a single obstacle, right-click the CAD objects and select **Merge selected objects** to merge them into a single CAD object.
4. Generate the navigation components as described in [Section 4.3](#).
5. Draw the desired obstacle with the **Add an Obstacle** tool .
6. Select the obstacle, and in the property panel change its **Results display** to **Imported**. The property panel will update with new available properties.
7. Click the **Pick** button next to **Imported geometry**.
8. In the 3D or 2D view, click the imported CAD object whose geometry is to be used.
9. If prompted, choose whether to move the geometry of the object to the location of the drawn obstacle geometry. You should only choose **No** if the drawn obstacle geometry is already approximately aligned with the selected CAD object.
10. Choose whether to delete the source CAD object.

### 3.4.2. Obstacle Properties

To edit an obstacle's properties, select the obstacle. Its properties will appear in the property panel as shown in [Figure 47](#).



Obstacle01	Material: <multiple>	X Bounds: -0.03 m, 0.08 m	Results display: Imported	Speed Modifier: Always 0.0
<input checked="" type="checkbox"/> Visible	<input checked="" type="checkbox"/> Color: [Color Picker]	Y Bounds: -0.12 m, -0.10 m	Imported geometry: Pick	
Opacity: 100.0 %	Z Bounds: -1.83 m, 0.08 m	<input checked="" type="checkbox"/> Link shape to CAD geometry		

Figure 47. Obstacle property panel

#### Color

The color of the selected obstacles (see [Section 3.2.5](#)).

#### Opacity

The opacity of the selected obstacles (see [Section 3.2.5](#)).

#### Results display

Controls how the object will be displayed in Results. The following values are available:

##### None

The obstacle will not be displayed in Results. This might be useful, for instance, if the obstacle is being used to re-route occupants due to an environmental hazard such as smoke

or fire.

### Simple

The obstacle will be drawn as a prism that extends up from the navigation mesh. If using a speed modifier, the obstacle is only visible if the factor is less than 1, and its height varies from 0 m to 1.3 m proportional to the inverse of its current speed factor. If using a speed limit, the obstacle is only visible if the limit was not set to disabled in Pathfinder, and the height is directly proportional to the speed limit divided by 1.2 m/s (see Figure 48).

### Imported

The obstacle's display derives from an imported CAD object. It is only visible if the factor is less than 1. See Figure 49 for an example. See Section 3.4.1.3 or Section 3.4.1.4 for more details on how to specify the CAD geometry.

### Imported geometry

Allows the source of the imported geometry to be chosen from an imported CAD object in the model when the **Results display** is set to Imported.

### Link shape to CAD geometry

If selected, the **Search Volume** of the obstacle is automatically calculated from the imported geometry. It is defined as a prism formed from the convex hull of the imported geometry points, whose top is located at the Z plane of the highest imported point and whose bottom is located approximately 1.8 m below the Z plane of the lowest imported point. The lower bound is chosen such that the obstacle's imported geometry can be located above the navigation mesh and still affect the mesh below, such that the obstacle can be treated as an overhead obstruction. This is similar to how obstruction subtraction works when generating navigation elements as discussed in Section 4.3. By default, the search volume is hidden for linked CAD obstacles. To show it, under the **View** menu, select **Show Imported Obstacle Search Area**. See Figure 46 for an example showing the extended **Search Volume**.

### Speed Modifier

A time-variable factor that is multiplied by an occupant's maximum speed when they travel through the obstacle. This factor also affects how occupants plan their paths and whether they will try to avoid walking through the obstacle. A factor of 0 is the most costly, and will cause occupants to completely avoid the obstacle, treating it as a wall. Higher values will cause the occupant to only travel through the obstacle if it results in a faster path to their destination. If an occupant is caught in an obstacle as its factor changes to 0, the occupant's speed will be reduced by multiplying by .01 so the occupant can still eventually escape the obstacle.

### Speed Limit

This works similarly to **Speed Modifier** except that it sets a maximum speed for the occupant

rather than a factor.

#### NOTE

When specifying the speed limit with the **Edit Speed Limit** dialog, enter **disabled** into either the **Initial Value** field or the **Value** column of the table to disable the speed limit. This is preferable to entering a high value, as **disabled** informs Results that the obstacle should not be shown at these times.

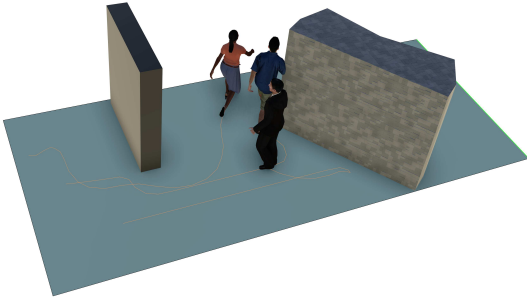


Figure 48. Obstacle with Simple Display in Results

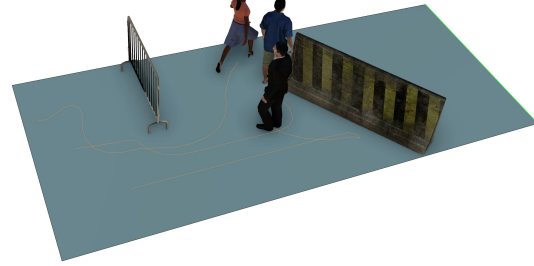


Figure 49. Obstacle with Imported Display in Results

## 3.5. Doors

In Pathfinder, occupants cannot pass between two rooms unless they are joined by a door. Doors provide useful flow measurements in simulation results. Also, in the SFPE mode doors act as the primary flow control mechanism. You can add doors using the **Add a New Door** tool.

#### NOTE

The simulator requires that each occupant must have a path to at least one exit door.

When adding doors, different parameters provide hints to Pathfinder for finding a valid door as shown in [Figure 50](#).

#### Min Width

Restricts potential edges to only those that are longer.

#### Max Width

Used as a target width for the door. If the full width isn't available, Pathfinder will display a shorter door as you hover with the mouse.

#### Max Depth

Represents the depth of the door cavity and is used to determine how far apart two rooms can be and still be joined by a door. Doors can be added between rooms that are separated by up to

this distance.

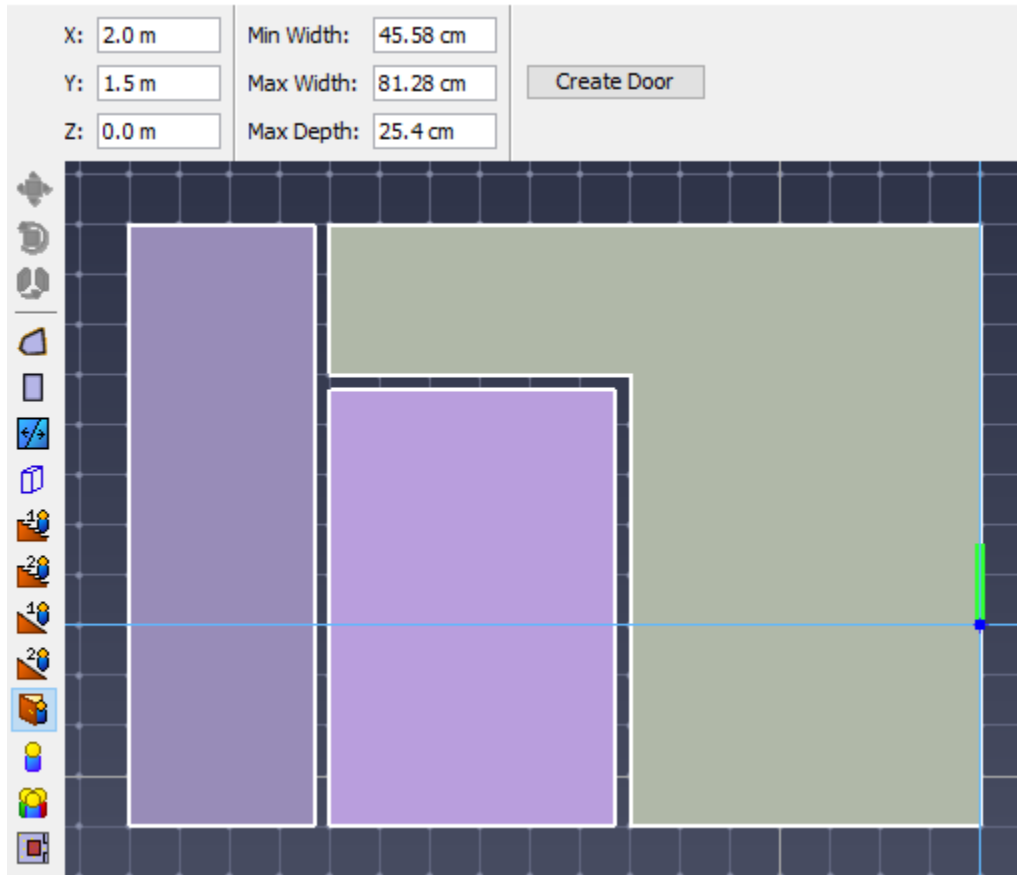



Figure 50. Door tool property panel

### 3.5.1. Thin Doors

Thin doors can be used to connect two rooms that touch one another as shown in [Figure 51](#). A door is needed in this example to allow occupants to travel from one room to the other.

To create a door in this manner:

1. Select the door tool .
2. Then use one of the following three methods:
  - a. **Manual Entry:** Enter the coordinates of the door in the property panel. If the coordinates specify a valid door location, the **Create Door** button will enable. Click this button, and a door no larger than **Max Width** will be created. For thin doors, **Max Depth** will be ignored.
  - b. **Single Click:** Move the cursor over the desired door location in the 3D or 2D view. A preview door will be displayed if the cursor is on a valid edge. The door displayed will lie either to the left or right of the hover point relative to the boundary edge, depending on

whether **Max Width** is positive or negative. Single click to place the door. The previewed door will then be added to the model.

- c. **Click-drag:** Move the cursor over the location of one end point of the door, and click drag along the same edge. While dragging, a preview door will be displayed from the first to the second point. When the mouse is released, a door is created along the edge between the two specified points. Creating a door in this manner ignores all the properties in the tool panel.
3. The created door will appear as a thin, orange line in the 3D and 2D views as shown in [Figure 52](#).

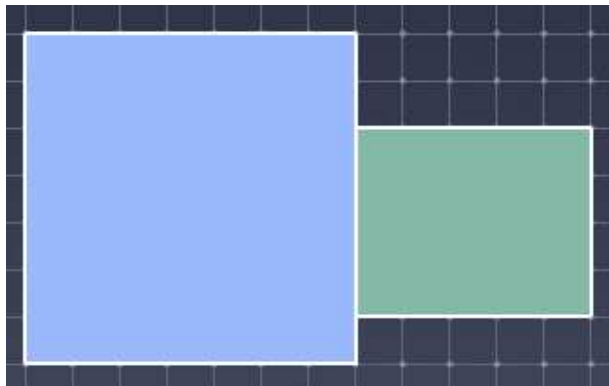


Figure 51. Two rooms with a shared wall

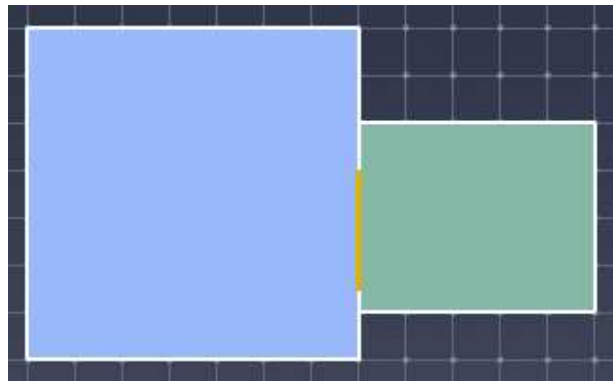



Figure 52. Adding a thin door to connect the rooms

### 3.5.2. Thick Doors

Thick doors are often useful in realistic models, especially when CAD geometry has been imported. In real scenarios, rooms will not touch each other by infinitely thin walls as shown in [Figure 53](#).

To create a thick door to connect these rooms:

1. Select the door tool 
2. Use one of the following three methods:
  - a. **Manual Entry:** Make sure **Max Depth** is greater than or equal to the distance between the edges the door will lie between. Enter a point on one of the edges in the property panel. If the coordinates specify a valid door location, the **Create Door** button will enable. Click this button, and a door no larger than **Max Width** will be created.
  - b. **Single Click:** Make sure **Max Depth** is greater than or equal to the distance between the edges the door will lie between. Then move the cursor over the desired door location in the 3D or 2D view. A preview door will be displayed extending between the rooms if the

cursor is on a valid edge. The door displayed will lie either to the left or right of the hover point relative to the boundary edge, depending on whether **Max Width** is positive or negative. Single click to place the door.

- c. **Click-drag:** Move the cursor over the edge of one of the rooms, and click drag along the corresponding edge on the second room. While dragging, a preview door will be displayed connecting the two edges. When the mouse is released, a door is created between the edges of the two rooms, where the diagonal of the rectangular door connects the two specified points. Creating a door in this manner ignores all the properties in the tool panel.

3. The created door will appear as an orange rectangle in the 3D and 2D views as shown in [Figure 54](#).

When simulating, thick doors have a special representation: the area of the door will be partitioned in two, and each half is attached to its touching room. A thin door is placed in the middle of the area to represent the thick door.

#### NOTE

The extra area attached to each room is neglected when the area of the room is reported in its property panel, but it is included during simulation.

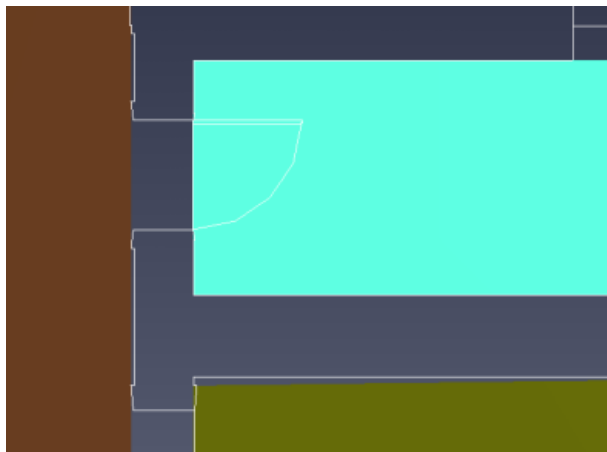


Figure 53. Two rooms separated by a gap

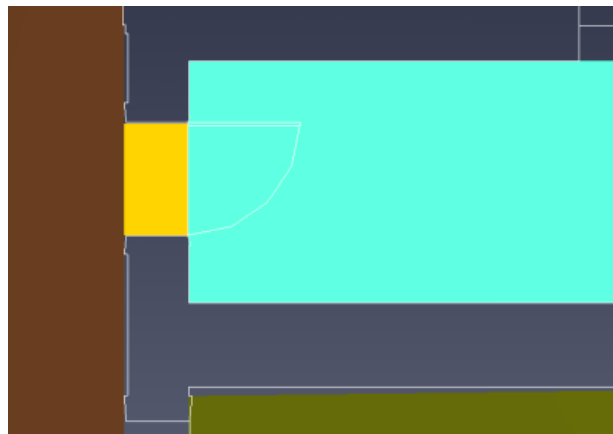


Figure 54. Thick door connecting the rooms

### 3.5.3. Door Properties

To edit a door's properties, select the door. Its properties will appear in the property panel as shown in [Figure 55](#).

Door00	<input type="checkbox"/> Color: <input type="text"/>	X Bounds: -15.66 m, -15.66	Width: 343.056178 cm	Wait Time: <input type="text" value="0.0 s"/>	Occupant Tags: <input type="text"/>
<input checked="" type="checkbox"/> Visible	Opacity: <input type="text" value="100.0 %"/>	Y Bounds: 4.01 m, 7.44 m	<input type="checkbox"/> Flow Rate: <input type="text" value="&lt;unlimited&gt;"/>		
		Z Bounds: 0.00 m, 0.00 m	State: <a href="#">Always Open</a>		
				Accepted Profiles: <a href="#">Default</a>	

Figure 55. Door property panel

**Color**

The color of the selected objects (see [Section 3.2.5](#)).

**Opacity**

The opacity of the selected objects (see [Section 3.2.5](#)).

**Width**

The width of the door. Changing this value will change the width of the door, but the value cannot exceed the length of its room edge.

**Flow Rate**

Checking this box overrides the default door flow rate setting in the **Simulation Parameters Dialog** (see [Section 15.1](#)). Setting this value controls the maximum occupant flow rate for the door in units of **pers/s**. This can be used, for instance, when a particular door has been measured to flow at a specific rate and that rate must be reproduced. A value of **0.9 pers/s**, for instance, would mean that one occupant can go through the door every 1.1 seconds (1/0.9).

**NOTE**

While setting a flow rate will set an upper limit, actual flow rates will often be slightly less in steering mode even if there is a large queue at the door. In addition, the flow rate is considered local knowledge. Occupants are only aware of the flow rate of doors in their current room and unaware of those downstream when calculating their global travel time as discussed in [Section 5.1.4](#).

**State**

Indicates the timed opening, closing, and changing the one-way direction of the door, see [Section 3.5.3.1](#). A One-way door is one through which occupants can only travel in one direction.

**Wait Time**

Amount of time for which each occupant must wait at the door before walking through it. This can be used to simulate turnstiles or doors with access keys. The specific wait time for each occupant will be drawn at random from a predefined continuous or discrete distribution.

**NOTE**

Wait time is considered global knowledge. An occupant knows about the wait times of downstream doors when choosing between doors in their current room. In addition, a local door's wait time is part of the **Global Travel Time** as discussed in [Section 5.1.4](#).



## Occupant Tags

A set of tags added to occupants when crossing this door. Each tag is separated by a space (see [Section 5.8](#)).

## Accepted Profiles

A list of occupant profiles for which the occupants can enter this door. By default, all occupant profiles are accepted.

### 3.5.3.1. Door State

By default, all doors are always open throughout the simulation. To change this, click the link to the right of **State**.

The **Edit Door State** dialog will appear as shown in [Figure 56](#). This dialog allows the initial state of the door to be specified as well as additional timed states.

As shown in the figure, for example, the door is initially open, closes at  $t=10$  s, opens in +X direction at  $t=20$  s, and then opens in both directions again at  $t=30$  s.

Even though a door can only be travelled through in two possible directions, the drop-down box allows +X, -X, +Y, and -Y. When one of these directions is chosen, the actual direction Pathfinder chooses is the closest along the door's normal.

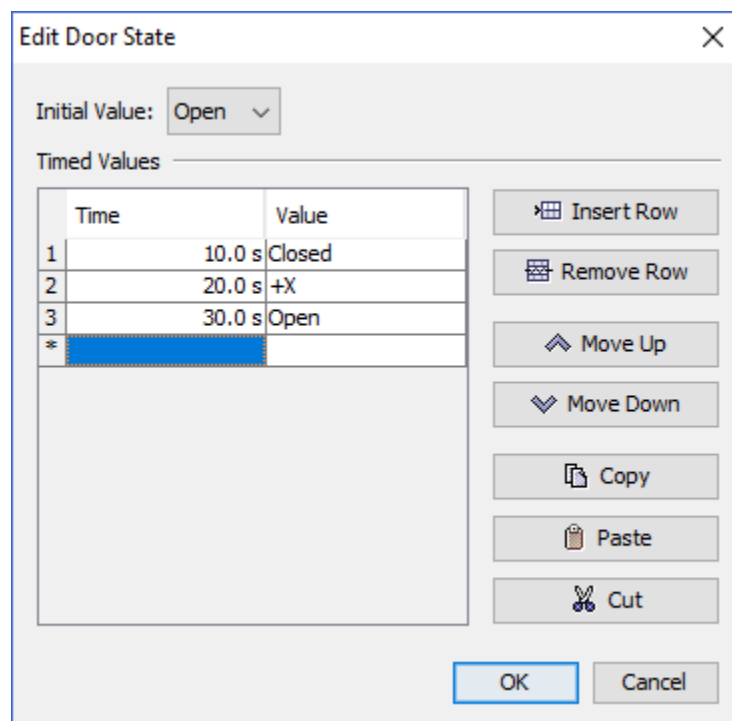


Figure 56. Door State dialog

**NOTE**

Occupants can ignore the one-way setting of doors if their profile has **Ignore One-way Door Restrictions** checked. This allows them to go through in either direction. In addition, door state is considered global knowledge - an occupant knows about the state of downstream doors when making a local door choice. For instance, if a door is permanently closed downstream, the occupant will plan a route that does not use that door. Similarly, they will not plan a route that goes the wrong way through a one-way door.

## 3.6. Stairs

Stairs in Pathfinder are represented by one straight-run of steps. They can be created with two tools. One tool allows creation of stairs between two semi-parallel boundaries of rooms, and the other allows creation of stairs that extend from one room boundary until a criterion is met, such as number of steps, height of stairs, etc., or until another room is reached.

One requirement of all stairs for successful simulation is that each end of the stairs must connect to boundary edges of the rooms, meaning that there must be empty space at the top of the stairway and empty space below the bottom. See [Stair geometry requirements](#) for an illustration of this issue. The size of the gap must be greater than or equal to the radius of the largest occupant to travel on the stairs.

### Stair geometry requirements

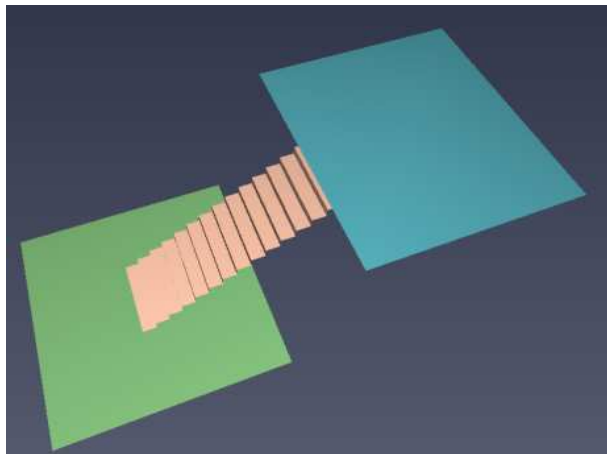


Figure 57. Shows a stair that will not simulate correctly

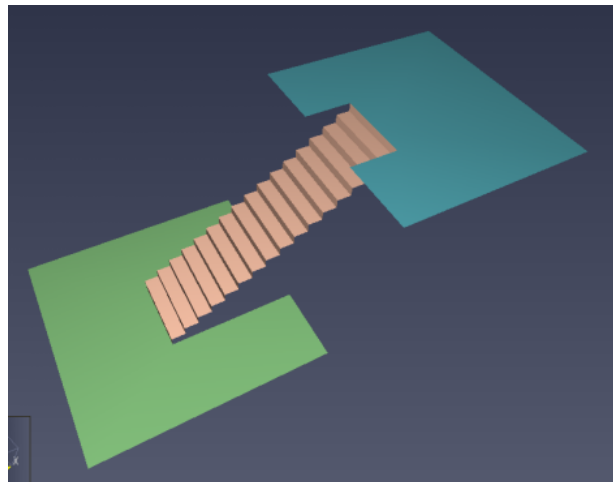



Figure 58. Shows a stair that will simulate correctly

### 3.6.1. Stairs Between Edges

One way to create stairs is to draw them between two pre-existing rooms. Stairs of this type will

match the ends of the stairs exactly to the edges that they were drawn between, which means that the tread rise and run may or may not match the actual slope of the stairs. In Pathfinder, the speed on stairs is determined by the specified tread and run. The geometric slope of stairs is for display only and not used to calculate the speeds.

To create stairs between edges:

1. Ensure that both the connecting rooms are visible. If the rooms are on different floors, at least one room will have to be manually set visible, which can be done through the right-click menu in the Navigation View.
2. Select the two-point stair tool, . The property panel will display the stair creation properties as shown in [Figure 59](#).
3. The stairs can now be created in one of the three following ways:
  - a. **Manual Entry:** Set the desired stair width, and for each edge on which the stairs should be created, enter a point. If the points are valid, a preview stair will be shown in the 2D or 3D view and the **Create** button will enable. Click **Create** to add the stairs.
  - b. **Two-click:** Set the desired stair width. Move the cursor over the first edge, and a preview line similar to a thin door will appear previewing the top or bottom of the stairs. Single-click to set the point. ([Figure 60](#)) Now move the cursor over the second edge. Now a preview of the stairs will appear connecting the first edge to the second. Single-click on the second edge to create the stairs ([Figure 61](#)).
  - c. **Two-click with drag:** Click-drag on the first edge to specify both the starting point and width of the stairs. Release the mouse and then single-click on the second edge to place the stairs.

#### NOTE

For stairs between edges, the **step count** for display of the stairs is determined by the tread rise and total vertical rise.

To change the display, you may:

1. Measure the **total vertical rise** of the stair.
2. Set the tread rise to **total rise/desired step count**

X1:	-0.5 m	X2:	0.0 m	Width:	121.92 cm	Tread Rise:	17.78 cm	Create
Y1:	2.0 m	Y2:	0.0 m	Door1 Width:	WIDTH	Tread Run:	27.94 cm	
Z1:	0.0 m	Z2:	0.0 m	Door2 Width:	WIDTH			

Figure 59. Property panel for the two-point stair tool

#### Drawing stairs with the two-point stair tool

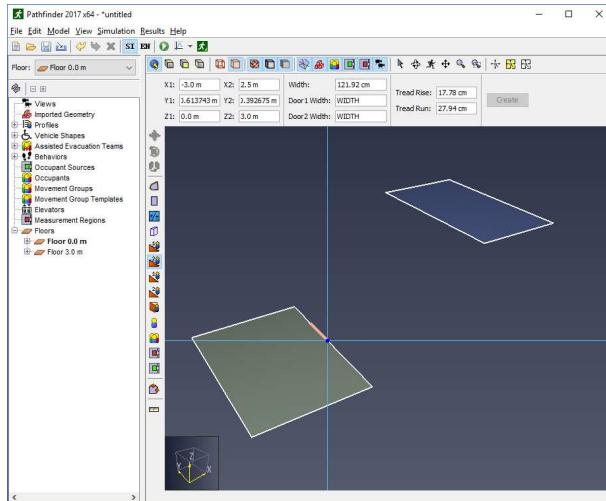


Figure 60. Click the first edge

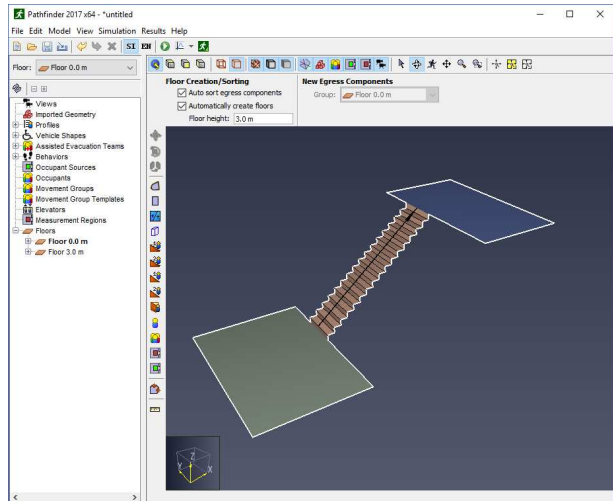


Figure 61. Click second edge to finish stairs

### 3.6.2. Stairs Extending from One Edge

Another way to create stairs is to have them extend from an edge and exactly match the specified tread rise and run. They will stop when they meet a specified criterion or reach another room.

The property panel for the one-click stair tool, as shown in Figure 62, provides four ways to terminate the stairs:

#### Step count

The stair will have this many steps.

#### Total rise

The stair will be this tall in the Z direction.

#### Total run

The stair will be this length in the XY plane.


#### Total length

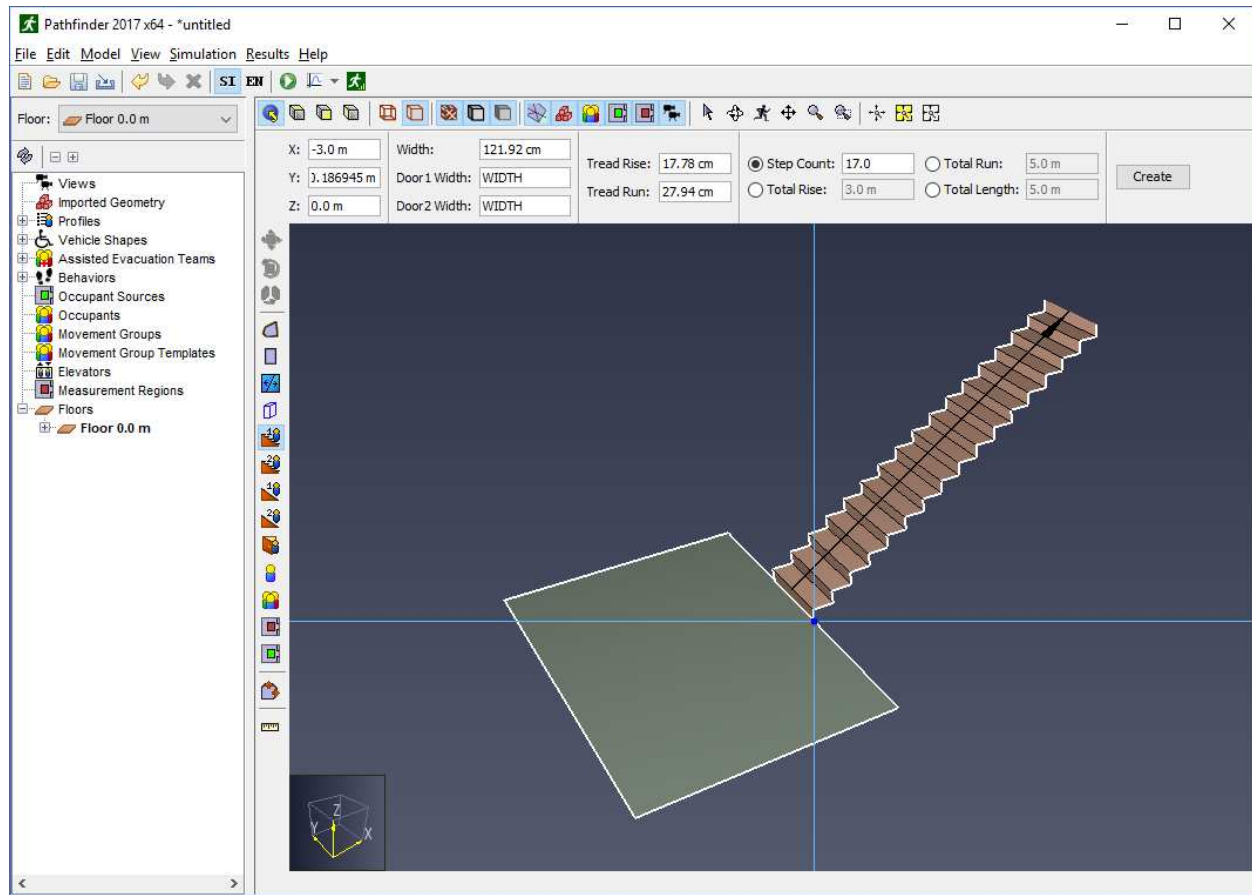
The hypotenuse of the stair will be this length.

X: 0.0 m	Width: 121.92 cm	Tread Rise: 17.78 cm	<input checked="" type="radio"/> Step Count: 17.0	<input type="radio"/> Total Run: 5.0 m	Create
Y: 3.0 m	Door 1 Width: WIDTH	Tread Run: 27.94 cm	<input type="radio"/> Total Rise: 3.0 m	<input type="radio"/> Total Length: 5.0 m	
Z: 0.0 m	Door 2 Width: WIDTH				

Figure 62. Property panel for the one-point stair tool

To create stairs in this manner:

1. Select the one-point stair tool, . The property panel will show.
2. If the tread rise is positive, the stairs will extend up from the starting edge, and if it is negative, the stairs will go down. Similarly, if the tread run is positive, the stairs will extend away from the room, and if it is negative, the stairs will extend in toward the room. Another way to change these values is to hold *CTRL* on the keyboard to make the tread rise negative and hold *SHIFT* to make the run negative.
3. Now the stairs can be created in one of the three following ways:
  - a. **Manual Entry:** Set the desired stair width, tread rise, tread run, and termination criterion. Specify the starting point on the desired edge. If the location is valid, a preview stair will be shown, and the **Create** button will enable. Click **Create** to add the stairs.
  - b. **Single-click:** Set the desired stair width, tread rise, tread run, and termination criterion. Move the cursor over the starting point on the room's boundary. A preview stair will be displayed. Single click to place the stairs.
  - c. **Click-drag:** Set the desired tread rise, tread run, and termination criterion. Now click-drag along the room's boundary to specify both the location and the width of the stairs.
4. Release the mouse to create the stairs.



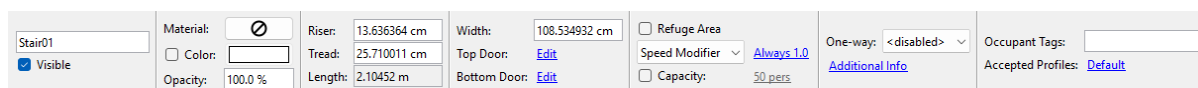
**Figure 63. Drawing stairs using the one-point stair tool**

After creating stairs in this manner, the Z location of the next floor or room will have to match the top of the stairs exactly for the next room to connect properly to the stairs. This can be done by clicking the top of the stairs in the 3D or 2D view when choosing the Z location for either the floor or next room.

After creating a one-click stair, you can only modify the stair by clicking the handles and dragging to existing geometry.

### 3.6.3. Stair Properties

Stairs have a number of properties that control their geometry and behavior of occupants that travel on them. When a stair is selected, these properties can be seen in the stair's property panel as shown in [Figure 64](#).



**Figure 64. Stair property panel**

**Material**

The material applied to the selected objects (see [Section 3.2.5](#)).

**Color**

The color of the selected objects (see [Section 3.2.5](#)).

**Opacity**

The opacity of the selected objects (see [Section 3.2.5](#)).

**Riser and Tread**

Together, these parameters control the speed at which occupants can travel on the stairs during simulation. The default speed on stairs uses the SFPE speed modifiers, see ([Pathfinder Technical Reference, n.d.](#)). While the one-point stair tool uses the tread rise and run to create the initial shape of the stair, these properties can later be changed without affecting the stair shape.

**Length**

The total length of the stair from the bottom to top edge. This is the same as the hypotenuse formed by the total stair rise and total stair run.

**Width**

The width of the stair.

**Top Door and Bottom Door**

Clicking these links show the dialog in [Figure 65](#). Here, properties of each implicit door can be edited independently of the other door, including Width, Flow Rate, and State (see [Section 3.5.3](#)).

**One-way**

Indicates whether the stair should only allow occupants to travel in one direction and if so, which direction.

**Speed Modifier**

A time-variable factor that affects the speed of occupants who travels on the stair. This acts the same as the Speed Modifier property for rooms (see [Section 3.2.5](#)).

**Capacity**

A maximum capacity of the stair, which can be specified using an occupant count or a density. This acts the same as the Capacity property for rooms (see [Section 3.2.5](#)).

### Additional Info

Clicking this link shows additional information about the stair, such as its geometric bounding box, area, and number of occupants.

### Occupant Tags

A set of tags added to occupants when entering the stair and removed from occupants when exiting it. Each tag is separated by a space (see [Section 5.8](#)).

### Accepted Profiles

A list of occupant profiles for which the occupants can enter this stair. By default, all occupant profiles are accepted.

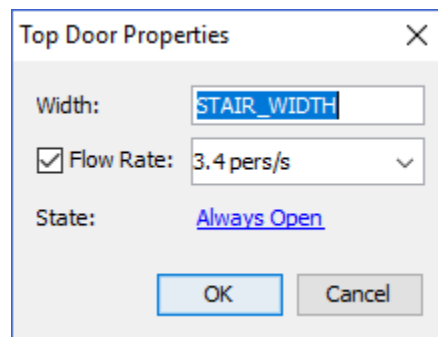


Figure 65. Stair door property dialog

## 3.7. Inferred Stairs

In areas on the navigation mesh that have not been defined as stairs or ramps, during the simulation Pathfinder attempts to identify when occupants are traveling over stair-like geometry and model their movement as though they are moving over stairs. In these situations where stairs have been inferred from the geometry, occupants will move slower based on the same speed fraction used for user-defined stairs and calculate distance traveled as though they are moving on a diagonal.

Inferred stairs commonly come from user-defined stairs that have been merged with surrounding geometry or from uneven geometry that has been joined using vertical floor or stair elements. Because Pathfinder agents' positions are confined to the navigation mesh, using such geometry as-is can introduce inaccuracies in speed and distance related to the diagonal nature of movement on stairs.

This feature is enabled by default and will operate without creating any specific modeling element. Inferred stairs can be disabled in the following two ways:

- Create an environment variable `PTH_STAIR_INFERENCE_ENABLE` and set the value to `false` then restart Pathfinder, or



- Run Pathfinder using the command-line parameter `-J-Dstair_inference_enable=false`

In most cases where the two are interchangeable, user-defined stair objects and inferred stairs produce very similar simulation results. Using one or the other may be more useful when creating a specific model element based on the situation. The following table offers a comparison between the two approaches to help clarify when one approach or the other might be more helpful.



**Table 4. Comparison with Explicit Stairs**

Characteristic	User-defined Stairs	Inferred Stairs
Control	Specify the intended tread height and depth independent of geometry as well as one-way options for escalators and moving walkways.	Slope is automatically determined by the raw simulation geometry and direction constraints are not supported.
Limitations	Top and bottom entry only.	No specific geometry limitations and stadium-style side entry steps are possible.
Simulator Performance	Slightly faster for occupants on the stair component because maximum velocity does not need to be calculated.	Slightly slower for occupants in the area of the inferred stair because maximum velocity must be calculated by analyzing the navigation mesh.

Additional information is available in the Pathfinder Technical Reference ([Pathfinder Technical Reference](#), n.d.).

## 3.8. Ramps

Ramps are nearly identical to stairs in how they are created and represented. Like stairs, they have two implicit doors at either end and always take the shape of a rectangular piece of geometry.

They also have very similar creation tools: the two-point ramp tool , and the one-point ramp tool . The key difference between ramps and stairs is that ramps do not affect the speed at which occupants travel by default.

## 3.9. Escalators

Pathfinder provides some limited support for escalators. They are essentially stairs with slightly modified properties.

To create an escalator, perform the following:

1. Create a stair as discussed previously.
2. Select the stair (or several) so its properties are visible in the property panel as shown in [Figure 64](#).
3. Set a one-way direction for the stair.
4. Click the **Speed Modifier** drop-down, and choose **Speed Constant** as shown in [Figure 66](#).
5. Edit the speed of the escalator in the speed constant field. As with the **Speed Modifier** for stairs, the **Speed Constant** can be time-variable. This would normally be used to turn the escalators on or off throughout the simulation by using the values 1.0 and 0.0 respectively, but any value can be entered.

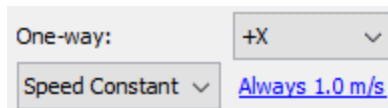


Figure 66. Changing a stair into an escalator

In the results view, escalators do not appear differently than stairs.

By default, occupants do not walk on moving walkways or escalators, and they will not stand on any specific side. This can be changed by modifying the occupant's profile and selecting their **Escalator Preference** (see [Section 5.1](#)). If an occupant is walking, the escalator's speed constant will be added to the occupant's current speed on the escalator.

#### NOTE

When escalators are turned off, occupants will use them as stairs, regardless of whether their profile indicates that they want to walk on escalators, or if their profile indicates that they should stand on a given side.

## 3.10. Moving Walkways

Pathfinder also provides limited support for moving walkways. This is similar to creating escalators, but instead of setting a speed constant on an existing stair, the speed constant is set on a Ramp instead, which can be made flat like a walkway.

## 3.11. Elevators

Pathfinder offers two different types of elevators, EVAC and SCAN. The two types differ in the order in which they decide to service levels and discharge occupants.

### 3.11.1. EVAC Elevators

EVAC elevators are intended to model egress-mode operation, which is based on current thinking described in *Using Elevators In Fires* (Bukowski and Li 2010).

The basic operation of elevators in evacuations can be summarized as follows:

- Each elevator has one *discharge* floor. This is where the elevator starts at the beginning of the simulation and is the default location to discharge occupants using a **Goto Elevators Action** (Section 5.3.3.3).

#### NOTE

In previous versions of Pathfinder, egress-mode elevators were restricted to discharging occupants at the specified discharge floor. This is no longer the case, as occupants can now target any floor that the elevator services, as long as it pertains to their current goal.

- Each elevator has at least one *pickup* floor. These are floors where the elevator will pick up occupants that it will take to the *discharge* floor.
- An elevator is *called* on a pickup floor by an occupant when they come within .5 m of the elevator door.
- The elevator uses a priority system to serve *called* floors. By default, floors are served from top to bottom; however, other floors can be given higher priority to simulate fire floors.
- When travelling to a pickup floor, the elevator can change to another pickup floor mid-flight if a higher-priority floor is called that is above the elevator's current location.
- Once an elevator has picked up occupants, it will only travel to a floor to discharge before letting the occupants off. It will not travel to any other floor to pick up more occupants.

### 3.11.2. SCAN Elevators

SCAN elevators are intended to model general-purpose elevator use. The basic operation of SCAN elevators can be summarized as follows:

The elevator will wait until it is called to service a floor. As soon as it is called, it will begin moving in the direction that it has been called from.

While the Elevator is moving, it will continue to travel in that direction as long as it has serviceable floors. Serviceable floors while the elevator is moving are:

1. In front of the elevator in the current direction of travel.
2. Have been called to continue in the current direction of travel.

**NOTE**

When calling a real elevator, occupants indicate which direction they intend to take the elevator by typically pressing an up or down-arrow button. This is what is meant by the direction that the elevator was called to travel in.

Once the elevator no longer has any serviceable floors in the current direction, it will look for the furthest elevator call regardless of the direction it was called to travel in. It will first prioritize the furthest call in the same direction last traveled. If no call exists, it then will move to the furthest call in the opposite direction last traveled.

This behavior is intended to scan or sweep the elevator floors from end to end, giving relatively equal priority to each floor that the elevator is called to. Once there are no active service calls, the elevator will wait until called again.

### 3.11.3. Creating Elevators

Elevators can be made after creating the rest of the model.

Perform the following steps to create the elevator (refer to [Figure 20](#)):

1. Draw a room that defines the shape of the elevator, preferably on the discharge floor, see [Figure 67](#).
2. Draw all doors on the boundary of the base room, see [Figure 68](#). Occupants will use these doors on every floor to enter and exit the elevator.
3. Right-click the base room, and from the right-click menu, select **Create Elevator**, see [Figure 69](#). This will show the **New Elevator** dialog as shown in [Figure 70](#).
4. In the **New Elevator** dialog, enter all parameters for the elevator:

**Name**

The name of the elevator.

**Type**

The type of the elevator (EVAC or SCAN).

**Nominal Load**

The number of people in a full load (estimated). Please read the Nominal Load section below for details.

**Elevator Geometry**

The base room that defines the elevator shape. This defaults to the room that was originally selected.

**Travel Direction**

A vector defining the direction the elevator can travel. This vector will automatically be normalized. The elevator can travel negatively against this vector.

**Elevator Bounds**

This defines the bottom-most and top-most floors the elevator can connect to.

**Elevator Timing**

This defines a basic timing model used to calculate the travel times for the elevator to travel between each level.

**Acceleration (Elevator Timing) [optional]**

The acceleration of the elevator.

**Max Velocity (Elevator Timing)**

The maximum velocity that the elevator can accelerate to.

**Open+Close Time (Elevator Timing)**

The sum of the door opening and closing times. Each value will be taken as half of this.

**Call Distance**

The distance away from the elevator door at which an occupant can call the elevator.

**Double-Deck**

Whether the elevator should use two connected decks to transport occupants. Please see the Double-Deck Elevators Section for details.

5. Click **OK** to create the elevator.

If necessary, Pathfinder will automatically subtract holes in existing geometry to make space for the elevator shaft. It will also delete existing rooms, doors, stairs, and ramps in the elevator shaft. Pathfinder will ask before making any of these changes.

In order for occupants to use elevators, they must either be allowed to use elevators in their Profile under Restricted Components (see [Section 5.1.2](#)) or be explicitly told to do so through their behaviors, as discussed in Behaviors. Occupants can be encouraged to prefer elevators to stairs using the Elevator Wait Time Profile parameter (see [Section 5.1.4](#)).

**NOTE**

The timing values can be re-calculated using a new timing model by selecting the elevator, selecting **Edit** next to **Level Data** in the property panel, and clicking the **Edit Elevator Timing...** button in the level data dialog.

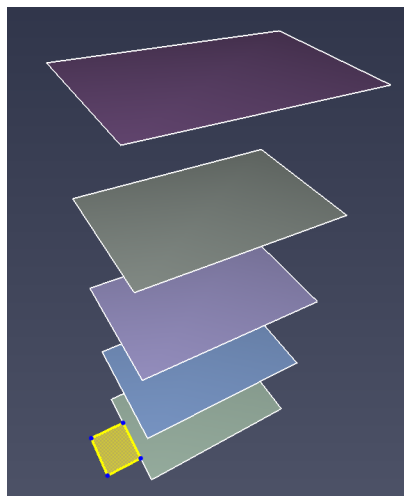


Figure 67. Draw elevator room geometry

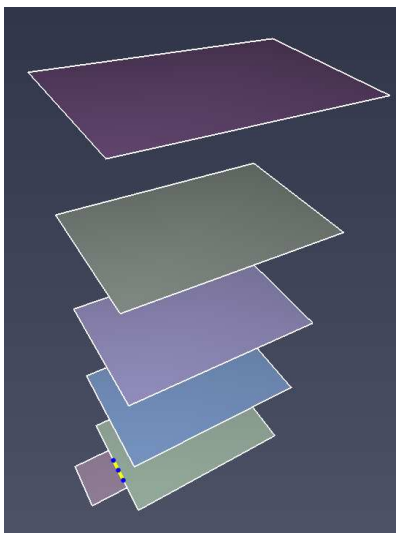


Figure 68. Draw elevator door

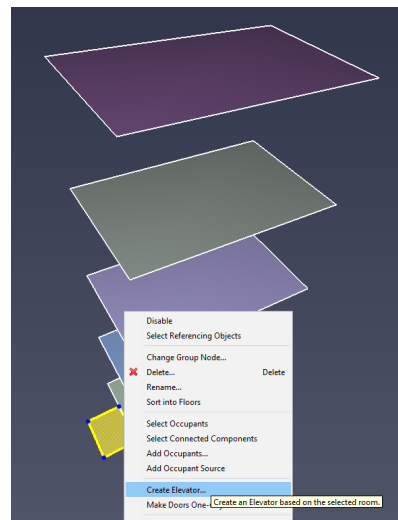


Figure 69. Select Create Elevator in right-click menu

New Elevator

Name:

Elevator00

Type:

EVAC

Nominal Load:

12.0 pers

Elevator Geometry:

Level 0.0 m

Travel Direction:

X: 0.0 Y: 0.0 Z: 1.0

Elevator Bounds

Bottom Floor:

[lowest]

Top Floor:

[highest]

Elevator Timing

☒ Acceleration:

1.2 m/s<sup>2</sup>

Max Velocity:

2.5 m/s

Open+Close Time:

7.0 s

Call Distance:

0.5 m

Elevator Type

☐ Double-Deck

OK

Cancel

Figure 70. New Elevator dialog

### 3.11.4. Elevator Representation

Once an elevator is created, it will appear in the model as a series of "rooms" and doors connected by a transparent elevator shaft as shown in [Figure 71](#). There is one room and set of doors for each floor to which the elevator can connect. In the 3D and 2D Views, each room is shaped the same as the base room that created the elevator. In the Navigation View, each room is shown under the elevator rather than the Floors top node, see [Figure 72](#). In addition, each set of doors for the room is shown under the room. By default, each of the rooms is named after the floor on which it connects. If the elevator is disconnected completely from a floor as discussed in [Connecting/Disconnecting floors](#), the room is named "<Disconnected Level>".

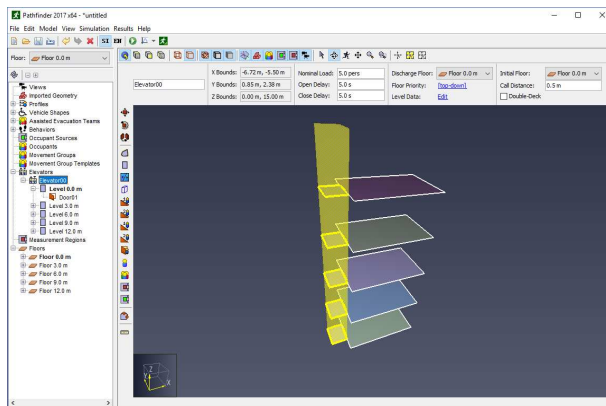


Figure 71. New elevator shaft highlighted.

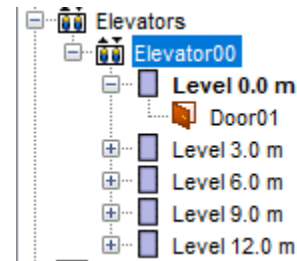


Figure 72. Elevator in Navigation panel.

### 3.11.5. Elevator Properties

Once an elevator is created, the elevator's properties can be edited by first selecting it from the navigation view or by *ALT*-clicking one of its rooms from the 2D or 3D view. Its properties can be edited in the property panel as shown in [Figure 73](#).

Elevators	Type: EVAC	Nominal Load: 9.0 pers	Discharge Floor: Floor 0.0 m	Initial Floor: Floor 6.0 m
		Open Delay: 5.0 s	Floor Priority: <a href="#">top-down</a>	Call Distance: 0.5 m
		Close Delay: 5.0 s	Level Data: <a href="#">Edit</a>	<input type="checkbox"/> Double-Deck

Figure 73. Elevator property panel

#### Type

The type of the Elevator which will dictate how it services floors.

#### Nominal Load

The number of people in a full elevator load (estimated). Please read [Section 3.11.6](#) below for details.

### Open Delay

The minimum amount of time an elevator's doors will stay open on a pickup floor.

### Close Delay

The minimum amount of time the elevator's doors will stay open after an occupant passes through one of the doors. This can also be thought of as the time someone in the elevator will hold the door open waiting for another occupant to enter. This delay time will reset every time an occupant passes through the doors. If the delay time passes without another occupant passing through the doors or without another occupant appearing in range of the doors or the elevator reaches capacity, the doors will close. The total amount of time for elevator doors to stay open, assuming no other occupants are within sight of the elevator who are heading toward it, can be summarized as: [stem 5e303889c624abe30a2c2c556d2ccfd6]

### Discharge Floor

The default floor at which occupants using a **Goto Elevators Action** will discharge at during an evacuation.

### Floor Priority

The priority of the floors for pickup. This is applicable only to EVAC type elevators, and by default is top-down. This can be changed however by clicking the text, which will show the **Floor Priority** dialog as shown in [Figure 74](#). This allows the simulation of a fire floor.

### Level Data

Click the **Level Data: Edit** link to edit timing for each floor. This will open the **Elevator Levels** dialog as shown in [Figure 75](#).

### Delay

The time delay from the start of the simulation for when the elevator can start picking up occupants from a floor. This value does not affect the discharge floor.

### Open+Close Time

The total amount of time it takes to open and close the doors for this floor. The door is assumed to open and close at the same speed, so after arrival at a floor the door will open after a delay of half the specified Open+Closed Time. Similarly, after the door closes the elevator will leave the floor after a delay of half the specified Open+Closed Time.

### Elevator Timing Model

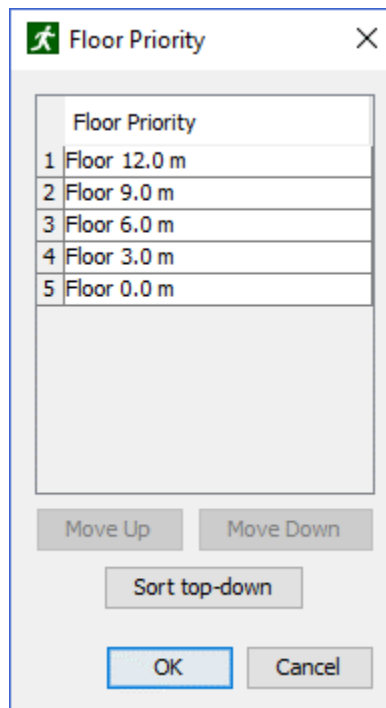
By selecting the **Edit Elevator Timing...** option at the bottom of the dialog, an Elevator Timing dialog will open as shown in [Figure 76](#), allowing you to modify the speed of the elevator moving between floors.



**NOTE**

Some models made in previous versions of Pathfinder may indicate that they are using a "Legacy" timing model. This model uses speeds derived from what was defined as the travel time between each level and the discharge floor for that elevator. When these elevators discharge on levels that are different from the discharge floor, they will fall back to using an average of the known elevator timings. Editing these timing models will require a conversion to either an acceleration or a velocity model that will broadly dictate travel time between each floor.

The timing options shown are the same as those shown when creating the elevator. Initial Floor:: The floor at which the elevator deck will be located at the beginning of the simulation. Call Distance:: The distance away from the elevator door at which an occupant can call the elevator. Double-Deck:: Whether the elevator should use two connected decks to transport occupants. Please see [Section 3.11.9](#) for details.



**Figure 74. Elevator Priority dialog**

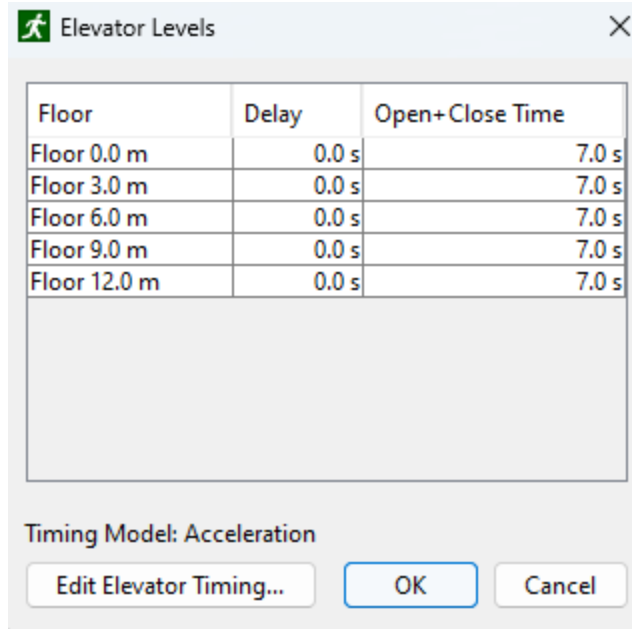


Figure 75. Elevator Levels dialog

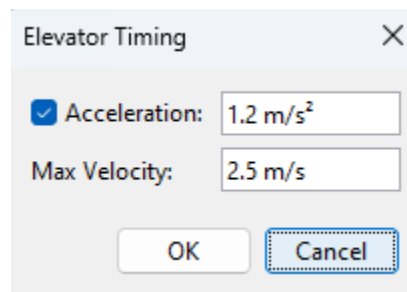


Figure 76. Elevator Timing Model dialog

### 3.11.6. Nominal Load

The nominal load is an estimate of the number of people that represent a full elevator load. The default value is based on an estimate of how many occupants of default size (diameter = 45.58 cm) would normally fill the elevator in steering mode. Increasing or decreasing the nominal load will cause occupants' sizes to be scaled up or down while they are on the elevator.

The scale factor (default: 1.0) is determined by a correlation to the density produced by the nominal load. This makes it possible to adjust loading while still accounting for differences in individual occupants' size.

In steering mode, the geometry of the elevator can lead to reduced loads (e.g. if the elevator is 2.8 persons wide). Please verify that the resulting (post-simulation) elevator loads match elevator manufacturer recommendations.

### 3.11.7. Connecting/Disconnecting Floors

When an elevator is created, by default it is connected to every floor its doors touch along the elevator shaft. Individual elevator doors can be disabled, however, to prevent entering/exiting the elevator through those doors on specific floors.

To do so:

1. Right-click the elevator door on the desired floor from the Navigation View or 3D/2D View.
2. Select **Disable** from the right-click menu.

To re-enable it, right-click the door from the Navigation View and select **Enable**.

Alternatively, right-click an elevator level and select **Disable** to disconnect all the elevator's doors on that level, effectively preventing the elevator from picking up occupants on that level.

### 3.11.8. Call Sets

By default, each elevator is called individually. Elevators can be grouped into call sets, however, so that when one is called, all elevators in the call set respond and travel to the pickup level.

To create a call set:

1. Create a group in the top Elevators group.
2. Add the desired elevators to the new group.

All elevators in the group will be in the same call set as shown in [Figure 77](#).

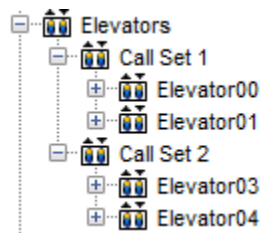


Figure 77. Elevator call sets

### 3.11.9. Double-Deck Elevators

Double-deck elevators use two connected decks to transport occupants, which increases the efficiency of moving occupants between floors. Occupants use a double-deck elevator similarly to a regular elevator.

After calling an elevator, the double-deck elevator arrives at the given floor, and the doors of its two decks open at the two adjacent floors (even and odd). When the decks are loaded, the double-

deck elevator moves to either pickup or discharge at another set of floors.

Occupants traveling in a double-deck elevator can only discharge on a floor with the same parity that they entered from. Occupants that entered on the lower (even) deck will only be able to discharge on even-numbered elevator levels. Occupants that entered on the upper (odd) deck will only be able to discharge on odd-numbered elevator levels. If no such level exists, the occupants could become stuck if using the explicit **Goto Elevators Action**, and otherwise they will choose not to use the elevator while pathfinding.

After arriving at a floor to discharge, the elevator doors of both decks open, allowing occupants to leave both decks of the elevator.

There are several conditions that must be followed when using a double-deck elevator in Pathfinder:

- The total numbers of elevator levels above and below the pair of discharge levels must be even.
- Any floor explicitly targeted by an occupant using a **Goto Elevators Action** ([Section 5.3.3.3](#)) must be enabled, and it must be the same parity (even or odd) as the level they will enter from.
- The vertical distance between each pair of even and odd level must be equal to the vertical distance between the lower and upper decks of the elevator, since the distance between the elevator decks is fixed. However, the distances between the different level pairs can be arbitrary.

## 3.12. Exits

In Pathfinder, exits are merely thin doors that exist on the boundary of the model. An exit can only have a room on one of its sides.

Exits are created in almost the same way as thin doors as discussed in the section, Thin doors. The only difference is that the door must lie on an edge of a room, and the edge must not be shared between two rooms.

Exit doors are displayed the same as thin doors except that they are green as shown in [Figure 78](#).

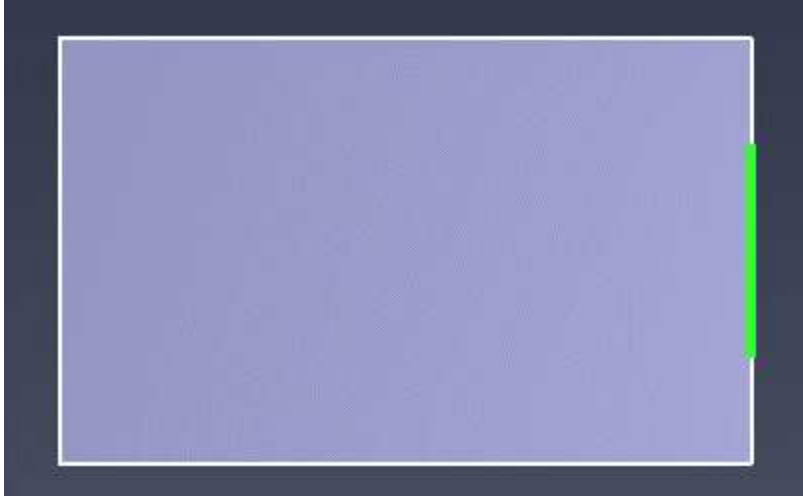


Figure 78. An exit door

### 3.13. Undo/Redo

All selections and model edits can be undone and redone using the Undo (↶) and Redo (↷) buttons, as well as the shortcuts **Ctrl+Z** and **Ctrl+Y**, respectively. The shortcuts **Ctrl+Shift+Z** and **Ctrl+Shift+Y** will skip selection history and revert or continue to the next model edit.

Alternately, the drop-down tab adjacent to the Undo (↶) and Redo (↷) buttons or the **Edit → Undo** and **Edit → Redo** menu can be used to view your Undo or Redo history and jump to a specific point.

The size of your history can be changed in **File → Preferences → Undo/Redo History**. History for model edits is limited by that preference, while history for selection-only changes has double that limit.

# Chapter 4. Importing Files

Pathfinder can import a large number of image and CAD formats. Imported files can be used as an aid to more quickly generate the navigation mesh and give more context and visual appeal to a simulation.

## 4.1. Importing Images

Background images can be imported by clicking **New Background Image** on the **Model** menu.

1. When clicked, a dialog will appear prompting for an image file. The following image formats are currently supported: BMP, GIF, JPG, PNG, and TGA. After a file is selected, a new dialog will appear as shown in [Figure 79](#). This dialog allows properties of the image to be specified so that the proper scaling, rotation, and offset can be applied.
2. An anchor point is specified on the image that indicates where that point is located in 3D space of the model. By default, the point you click will be placed at the origin point  $(0,0,0)$  in the model.
3. To specify the scaling for the image, two points, A and B, and a distance between them can be specified ([Figure 80](#)).
4. To specify the rotation of the image, an angle can be input that specifies the angle of the vector from A to B with the vector  $(1,0,0)$ . As shown in the figure, the  $A \rightarrow B$  vector should be 90 degrees from the X axis.
5. Click **OK** and the image will be placed in the model as specified ([Figure 81](#)).

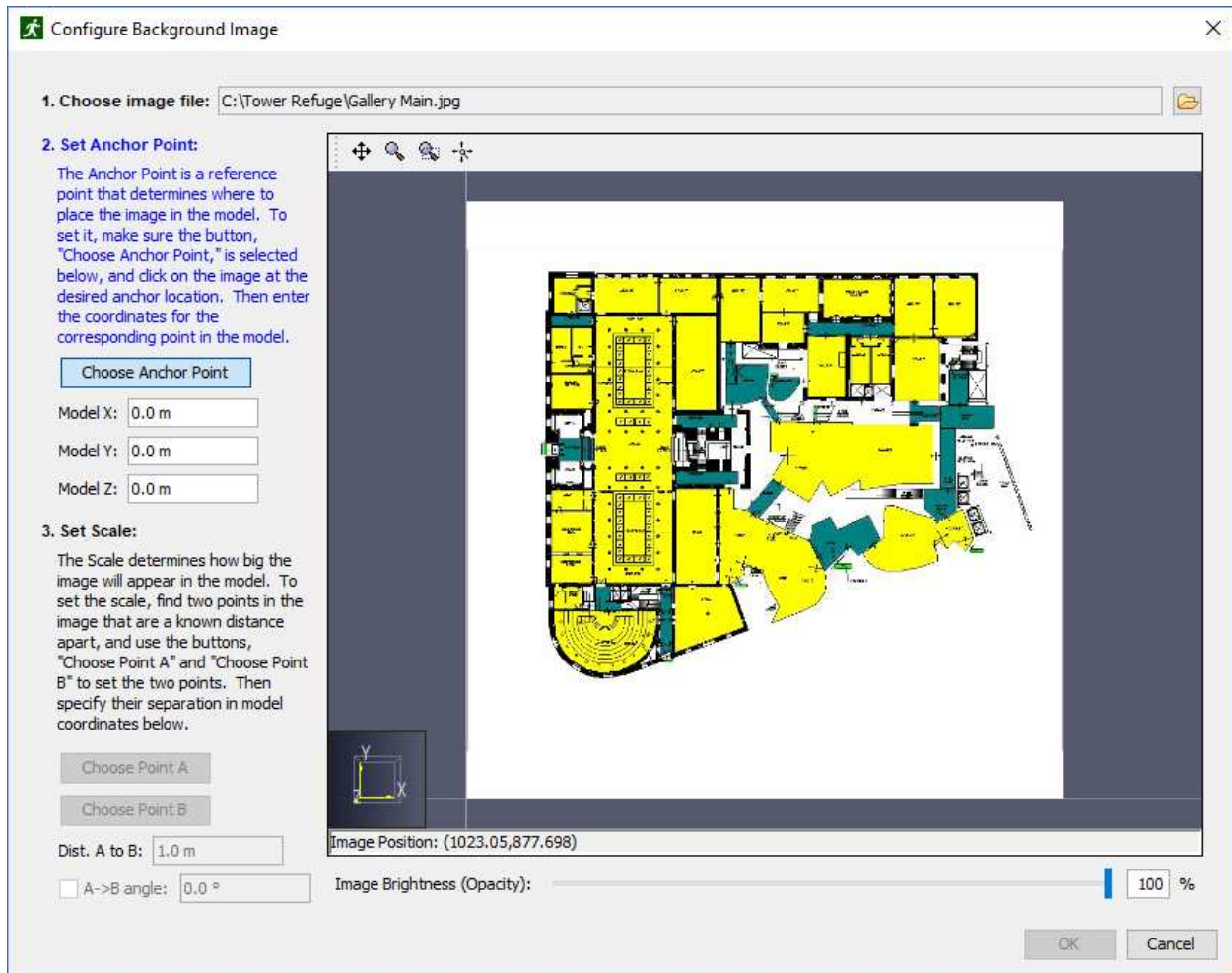


Figure 79. Set the anchor point

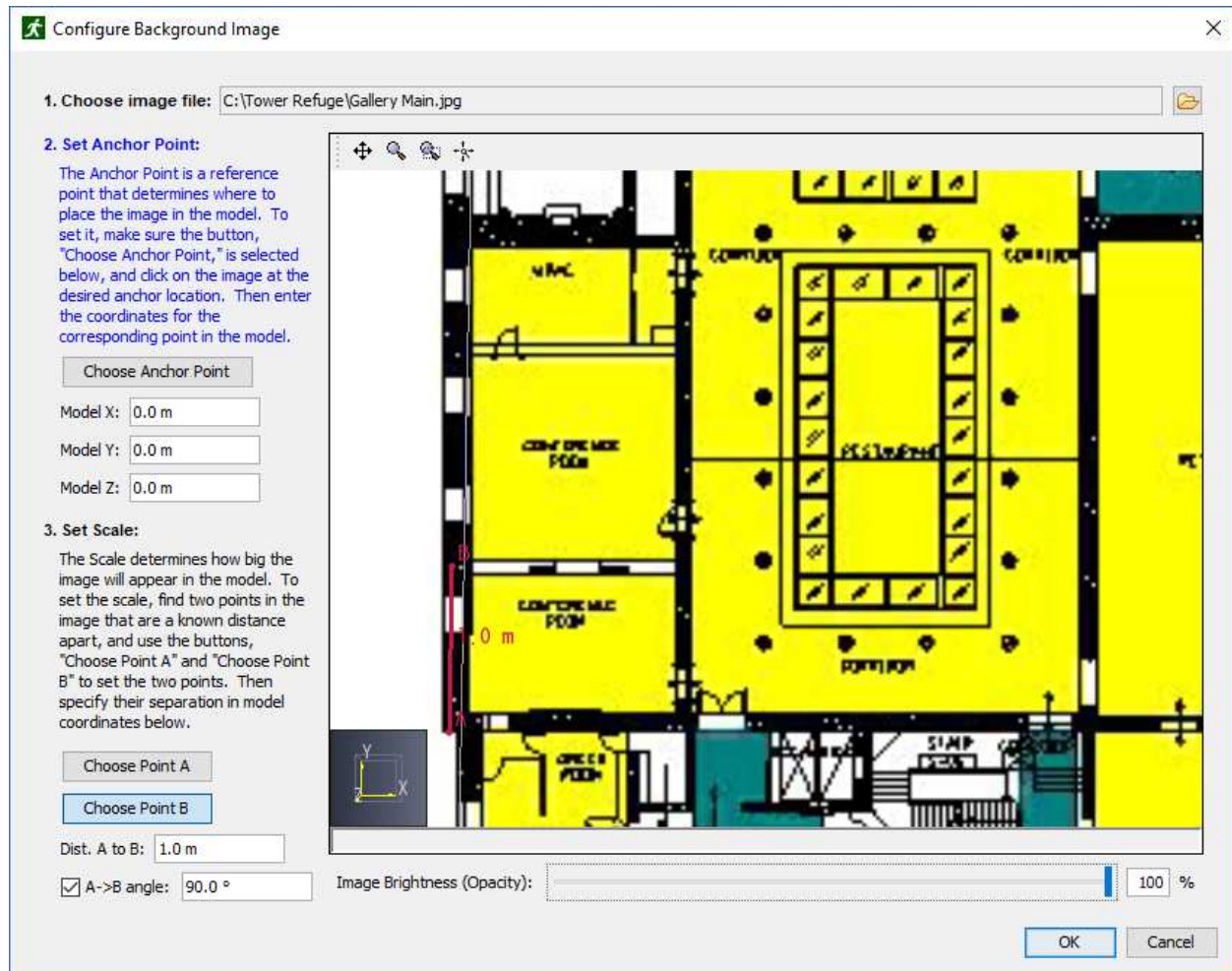


Figure 80. Set scaling points A and B



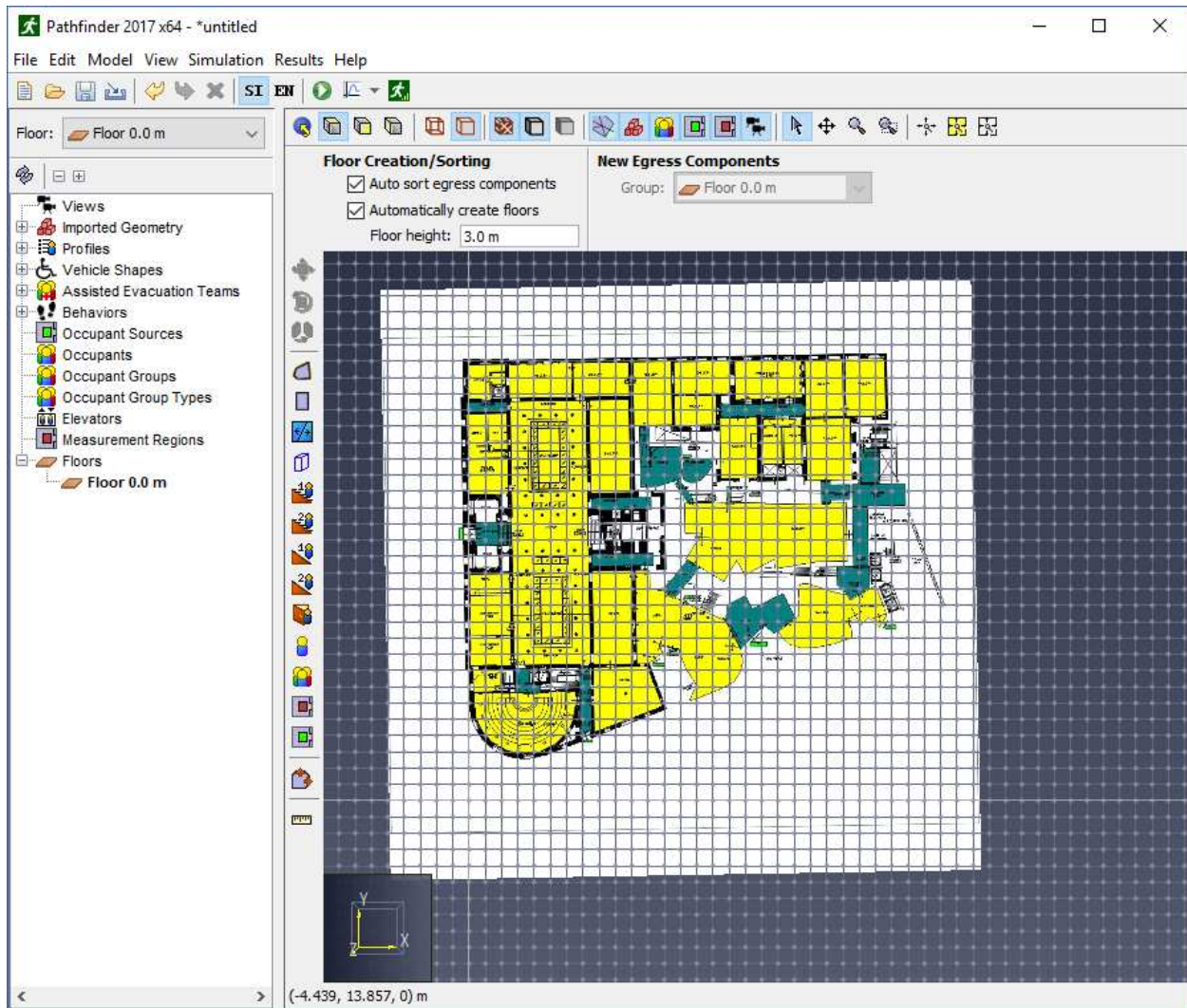


Figure 81. Background image added to the model.

The imported image is added to the **Imported Geometry** → **Background Image** group in the Navigation View. The image can be edited and deleted from there.

**NOTE** Any number of images can be added to any floor.

## 4.2. Importing CAD Files

Pathfinder can import geometry from several CAD formats, including buildingSMART's IFC format for building information models (BIM), AutoCAD's DXF (Drawing Exchange Format), DWG, FBX, DAE, OBJ, and glTF/GLB (Graphics Language Transmission Format) files.

Depending on which type of file is imported, Pathfinder also provides various tools to generate the navigation mesh elements such as rooms, doors, and stairs, see [Section 4.3](#).

To import one of these files:

1. Under the **File** menu, select **Import** and select the desired file.
2. After selecting a file, a step-by-step dialog will open as shown in [Figure 82](#).

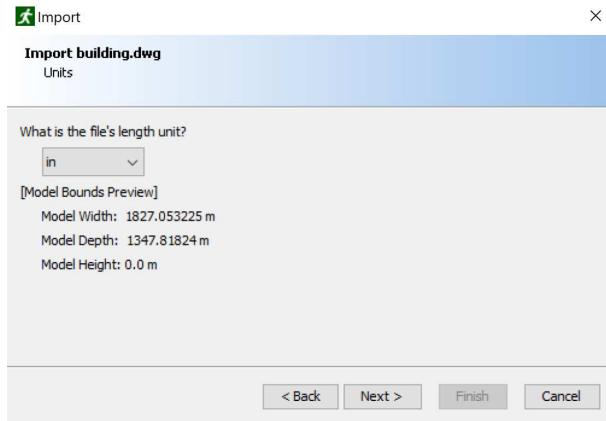


Figure 82. CAD import dialog (units)

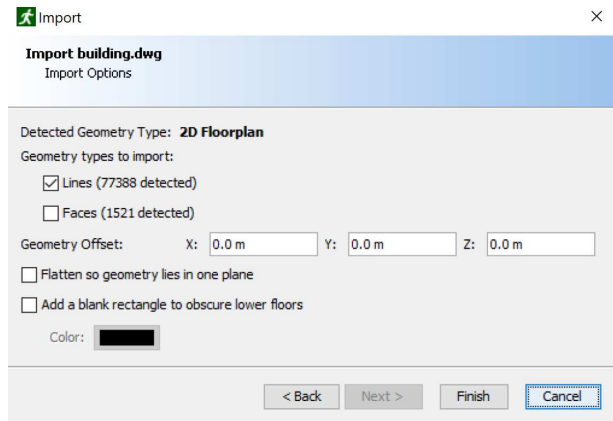


Figure 83. 2D CAD import options

3. The first prompt asks whether the CAD data should be imported into the current model or into a new one. Importing into the current model allows many CAD files to import into one Pathfinder model.

### FBX Exporter

If an FBX file is being imported, the second prompt will ask what software was used to export the file. If a SimLab FBX plugin or Unreal Engine was used to create the FBX file, choose the appropriate option from the drop-down box; otherwise, choose **Unknown**. This selection controls the default settings in the subsequent prompts. In some cases, Pathfinder is able to detect whether the file was exported using a SimLab plugin and will select this option automatically.

4. **Units:** Select the base unit in which the CAD file was created. If the drawing was saved in a more recent file format, the prompt will default to the unit type stored in the file. The dialog box shows the model's width, depth, and height based on the selected unit as a guide for selecting the unit.
5. **Import Settings:** The next prompt allows the user to control how some of the data is imported and to correct some data that may have been written incorrectly by the file's CAD exporter.

### Normal Tolerance (DWG/DXF only)

Controls the quality of curved objects. Decreasing this value produces higher quality objects at the expense of slower rendering speed. The default value of [stem 6fed8823511909ba34704c1948f66a03] provides a nice balance.

**NURB Gridlines (DWG/DXF only)**

Controls the quality of NURB surfaces. Increasing this value gives higher quality curves at the expense of slower rendering speed. The default value of [stem c4a08d3f36d17948846ec69cfe53c5a2] provides a good balance.

**Lighting**

Controls how objects are lit.

**Auto-correct inverted polygons**

Some CAD files contain information about the normal of a polygon that affects how the polygon is lit. In some cases, the normal may not match the orientation of the polygon, which can cause the polygon to appear too dark. Selecting this option will allow Pathfinder to try to detect these cases and correct the orientation of the polygon so it may be lit correctly. This option works well in most models and is generally safe to leave on.

**Crease Angle**

Some CAD files do not provide lighting data for objects to determine whether they should look smooth or faceted. In this case the crease angle is used to determine this information. If the angle between two adjacent faces is greater than this value, the faces will be look like they are two separate, faceted faces. Otherwise, the two faces will look like one smooth curved face. This option may also affect whether the edges of faces are displayed when the **Solid/Realistic with Outlines** option is selected as explained in [Section 2.3.1](#).

**Materials**

Specifies how CAD materials are imported. CAD materials control the imported objects' color and lighting (see [Section 4.3.7](#)).

**Merge identical materials**

Some CAD exporters (namely SimLab's Revit FBX plugin) will create a unique material per object in the file, which may lead to hundreds or thousands of materials that have duplicate properties. Selecting this option allows the materials with duplicate properties to be merged into one material, significantly reducing the number of materials in the model with no loss of quality. The only disadvantage to using this option is that some selection granularity is lost when using the **Select All by Material** action as more objects will be assigned the same material.

**Ignore transparency color (FBX only)**

In FBX files, material transparency is determined from a transparency color and factor. Some CAD exporters (SimLab's FBX plugins) export the color incorrectly. Selecting this

option will allow Pathfinder to ignore the transparency color in the FBX file and only use the transparency factor, which allows transparency in these files to import correctly. This option should only be selected if it is known that the file came from a SimLab plugin or there are transparency problems without it selected (For example, objects that should be transparent are not or vice versa).

#### **DirectX normal maps (FBX and IFC only)**

Indicates whether the normal maps in the imported file use the DirectX convention, where +Y is down.

#### **Workflow (FBX and IFC only)**

Specifies the lighting workflow to be used for imported materials. More information can be found in [Section 4.3.7](#). The following values are available:

##### **As defined in file**

The materials are imported exactly as they were specified in the file. Currently, this means that the materials will be imported using the **Specular (Basic)** workflow, as this is the only workflow currently supported by IFC and FBX files. In some cases, however, such as when an FBX file is exported from Unreal Engine, the PBR parameters are packed into the basic specular texture property. In this case, the following two options can be used to reinterpret the basic specular workflow as a PBR workflow for correct lighting.

##### **Metallic (PBR)**

The materials should be imported as if they use the **Metallic (PBR)** workflow, even if they are specified in the import file using the **Specular (Basic)** workflow. When using this option, the PBR parameters can either be set to constant values or can be reinterpreted from other non-PBR color properties. For instance, when an FBX file is exported from other software, for each material it might create a single image containing the **Metallic**, **Roughness**, and **Ambient Occlusion** parameters, stored in the red, green, and blue color components, respectively. It then might set the specular texture in the FBX file to this combined image. When an exporter does this, there should be accompanying documentation with the FBX file that indicates how these PBR parameters are stored in the FBX file. When importing the FBX file in the above example, in the import dialog, the **Metallic**, **Roughness**, and **Ambient Occlusion** properties should all be set to **From Specular**, and the color components should be set to **R**, **G**, and **B**, respectively (see [Section 4.3.7](#)).

##### **Specular (PBR)**

Similar to the **Metallic (PBR)** option, this forces imported materials to use the **Specular (PBR)** workflow, where the PBR parameters can be reinterpreted from

non-PBR properties.

### Object Grouping (IFC only)

Specifies how objects will be grouped once imported into Pathfinder. The value can be one of the following:

#### Spatial (default)

The objects are grouped based on the spatial layout of the model. For instance, the top group is the building site, the next group is the building, the next are the floors, etc.

#### By Type

Objects are grouped by type. For instance, there will be a group for walls, furniture, slabs, etc.

6. **Import Options:** This prompt allows the user to specify more options for import. Before this prompt is shown, Pathfinder will attempt to discern if the CAD file contains a 2D Floorplan or a 3D Model and will select default values for the options based on the detected type.

#### Lines

Check to import lines in the file (default=checked only for floorplans.)

#### Faces

Check to import faces in the file (default=checked only for 3D models.)

#### Move geometry to origin (0,0,0)

If checked, the imported geometry will be moved such that the minimum point of the bounding box surrounding the imported geometry will correspond with the model origin.

#### Geometry Offset

Will offset imported geometry on the X, Y, and Z planes by the specified values. If **Move geometry to origin** is checked, the geometry will be moved to the origin before being moved additionally by the **Geometry Offset**.

#### Flatten so geometry lies in one plane

If checked, all geometry will be scaled in the Z dimension by a very small scale (1e-9). This is useful for floorplans that have entities drawn in several planes. This option will flatten them all into one plane (default=checked only for 2D floorplans.)

#### Add a blank rectangle to obscure lower floors

If checked, a solid rectangle of the specified color will be added to the model. This is useful to obscure geometry located on lower floors. The imported rectangle will have its **Import**

**Type** set to **<ignored>** by default so it does not contribute to generating the model as described in [Section 4.3](#), (default=checked only for 2D floorplans.)

7. Choose **Finish** to import the file.

All imported elements will be added to the "Imported Geometry" node in the Navigation View. If the CAD file was a DWG or DXF, the grouping structure will include the model level, the layer level, and all entities distributed within the layer. For IFC files, the grouping is determined by the **Object Grouping** setting as specified above. For other CAD files, the grouping structure will match the node structure in the file. If both lines and faces were included in the import and an entity contained both lines and faces in the CAD file, the entity is split into two in Pathfinder - one with the lines, and one with the faces.

#### NOTE

In versions prior to Pathfinder 2012.1, DXF import allowed a background image to be created that could be passed through to the 3D results viewer. While this option is no longer available, both imported line and face data is now sent directly to the 3D results viewer instead, eliminating the need for a background image.

### 4.2.1. Imported Objects

When a CAD file is imported, there may be many resulting objects. Depending on the type of imported file, there are various levels of information that may be imported for each resulting object. Each object has a name and always includes some geometric information, such as the 2D curves or 3D faces composing the object. For some files, such as IFC files, there may be even higher-level information such as whether the object is a door and the door's width. Pathfinder uses this information to generate model elements as described in [Working with Imported Data](#).

Common to all import objects is the ability to set some visual properties, such as color and opacity for each component of the geometry. The imported geometry is sent as-is to 3D Results, resulting in a clean and fast graphical representation of the data.

When an imported object is selected, its property panel appears as shown in [Figure 84](#).

**Figure 84. Imported IFC object's property panel**

Imported objects have the following properties:



**Visible**

Whether the object is currently being shown in the 3D or 2D view.

**Material**

The material applied to the selected objects (see [Section 3.2.5](#)).

**Color**

The color of the selected objects (see [Section 3.2.5](#)).

**Opacity**

The opacity of the selected objects (see [Section 3.2.5](#)).

**X, Y, Z Bounds**

The bounds of the object's geometry.

**Object Type**

The object's originating type as specified in the imported file. For IFC files, this is the object's IFC Entity type, such as **IfcRoof**. This value cannot be changed.

**Import Type**

Specifies how the object is to be treated when automatically generating the model as described in [Section 4.3.1](#). Pathfinder chooses the value for this property during import. For more information on how the **Import Type** is chosen, see the section for each import file type (see [Section 4.2.2](#)).

**Results Visibility**

Controls the visibility of the object over time in the 3D results. This can be used to animate changes in the scene geometry.

**Generate Model from BIM Selection**

This action attempts to generate a Pathfinder model from the selected imported objects (see [Section 4.3.1](#)).

## 4.2.2. Importing IFC Files

IFC files provide building information model (BIM) data in a fully 3D format. This format contains advanced data about the types of objects in the building, including slabs, stairs, doors, etc., and it provides the smoothest workflow for converting imported objects into Pathfinder elements (see [Section 4.3.1](#)). It is also supported as an export format for many architectural CAD packages, including Revit.

**NOTE**

Unlike with the other import formats, however, textures are not currently supported.

When exporting an IFC file from another CAD package, such as Revit, it is preferable to use the IFC 2x3 Coordination View, though other IFC views should work as well.

Each object imported from an IFC file corresponds with an IFC Entity instance. Currently, only instances with 3D geometry are imported. In addition, openings (holes) are pre-subtracted from objects, and the openings are not imported as objects. For example, a wall in the IFC file might be associated with an opening object for a window. When importing, Pathfinder will subtract the opening's geometry from the wall geometry and only import the resulting wall. The window would additionally be imported as a separate object.

The **Object Type** for each object is set to the object's IFC Entity type, such as IfcWall. The **Import Type** is chosen automatically based on the object's **Object Type** and possibly other properties specified in the IFC file. For instance, an IFC object with entity type, IfcCovering has an associated PredefinedType property that specifies what kind of covering it is, such as a wall covering or floor covering. For floor coverings, Pathfinder will automatically set the object's **Import Type** to Floor during import. Other types of coverings become Obstruction.

Table 5 specifies how the **Import Type** is chosen for imported IFC objects:

**Table 5. IFC Import Type for IFC Objects**

IFC Entity Type	IFC Predefined Type	Object Name	Import Type
IfcBuildingElementProxy		"Path of Travel" "RPC Male" "RPC Female"	<ignored>
IfcCovering	FLOORING		Floor
IfcDoor			Door
IfcElement			Obstruction
IfcRamp			Floor
IfcSlab			Floor
IfcStair			Stair
IfcTransportElement	ESCALATOR		Stair
IfcTransportElement	MOVINGWALKWAY		Floor
IfcSpace			Room
IfcBuilding			Building
<all others>			<ignored>



**NOTE** The object's name must contain one of the specified texts.

The IFC Entity Type includes derived entities unless specifically listed in the table. For instance, **IfcElement** includes **IfcBeam**, **IfcColumn**, etc., since those entities derive from **IfcElement** and are not listed in the table. **IfcElement** does not, however, include **IfcDoor** since **IfcDoor** is in the table. In addition, **IfcDoor** includes both **IfcDoor** and **IfcDoorStandardCase** since **IfcDoorStandardCase** is derived but not listed.

### 4.2.3. Importing DXF Files

DXF is a basic CAD format provided by Autodesk. This format supports basic geometry types, including 3D faces, lines, and text, but it does not support material information, such as textures, lighting parameters, etc.

**Import Type** for all DXF objects is **Obstruction**. In order for the **Import Type** to be more useful, it must be set manually. Some suggestions for doing so are given in [Section 4.3.1](#).

### 4.2.4. Importing DWG Files

The DWG format is similar to DXF, but it also has basic support for materials, including textures. It has only basic support for mapping textures onto objects, however, and few CAD applications can export DWG files. Some, such as Revit, exclude material and texture information (see more information in Importing Revit Files below).

**Import Type** for all DWG objects is **Obstruction**. In order for the **Import Type** to be more useful, it must be set manually. Some suggestions for doing so are given in [Section 4.3.1](#).

### 4.2.5. Importing FBX Files

FBX provides support for 3D faces only, but it has good support for material information and materials mapping. In addition, many 3D modeling applications have native support for exporting FBX files.

**NOTE**

FBX files do not natively support physically-based materials (PBR). PBR materials can greatly increase the visual fidelity in Results when combined with image-based lighting. Some FBX files may be structured to partially support PBR materials, but these are rare and are more difficult to import correctly (see [Importing CAD Files](#)). In most cases, it may be preferable to use glTF files instead, as described below.

**Import Type** for all FBX objects is **Obstruction**. In order for the **Import Type** to be more useful, it must be set manually. Some suggestions for doing so are given in [Section 4.3.1](#).

### 4.2.6. Importing glTF Files

Pathfinder also supports importing **glTF 2.0** files (and the **GLB** variant). Like FBX files, glTF is a 3D interchange format with good support for materials and material mapping and is most commonly used to specify 3D faces, though it has some limited support for lines. glTF files can also include skinning information, which makes it useful for importing custom avatars as well (see [Section 4.5](#)).

Unlike FBX files, glTF files have native support for physically-based materials (PBR), which can significantly improve the appearance of imported geometry, especially when combined with image-based lighting in Results. In addition, glTF is a newer format than FBX and is not natively supported for export by as many 3D modeling applications; however, external export plugins are available for many of these applications.

glTF files come in two flavors:

#### glTF

A lightweight JSON file that references other texture and mesh data files. When copying/moving a glTF file, all the referenced files also need to be copied/moved.

#### GLB

A binary file that contains all required data in a single file.

The following glTF extensions are also supported:

- [KHR\\_draco\\_mesh\\_compression](#)
- [KHR\\_texture\\_transform](#)
- [KHR\\_materials\\_pbrSpecularGlossiness](#)

**Import Type** for all glTF objects is **Obstruction**. In order for the **Import Type** to be more useful, it must be set manually. Some suggestions for doing so are given in [Section 4.3.1](#).

### 4.2.7. Importing PyroSim and FDS Files

PyroSim and FDS files provide objects with 3D faces. If the imported file contains holes, the holes will be automatically subtracted from the solid obstructions and discarded. If the file contains grids, the grids will be intersected with each other as FDS would, and the remaining faces of the grids will be imported. If the file contains OPEN vents, the vents will be subtracted from the appropriate grid faces and discarded.

**Import Type** for all PyroSim and FDS objects is **Obstruction**. In order for the **Import Type** to be more useful, it must be set manually. Some suggestions for doing so are given in [Section 4.3.1](#).

## 4.2.8. Importing Revit Files

While Pathfinder cannot directly import Autodesk Revit files (RVT), there are several ways to export the data from Revit into a file format that Pathfinder can read. Each method has advantages and disadvantages as discussed below.

### 4.2.8.1. Revit to FBX using third-party plugin

This method requires the use of a third party plugin, but it generally produces good results with materials, textures, and texture coordinates well-supported. In many cases, this is the most reliable method of reproducing the graphical representation of the original Revit file within Pathfinder. **SimLab Soft** is one company that provides commercial FBX export plugins for several CAD packages, including Revit and Sketchup, among others, and provides robust texture support.

To export using a third-party plugin, perform the following:

1. Download and install the appropriate plugin.
2. Follow the plugin's instructions to export an FBX file from Revit. If the plugin supports embedded media, select this option before exporting. This option allows textures to be embedded into the FBX file, making it much easier to transfer the FBX to another computer, as only file has to be transferred.
3. If the FBX file is to be imported into Pathfinder on the same computer as the one that exported the file or the embedded media option was selected, continue to the next step; otherwise, some additional steps may be necessary to ensure the textures can be found when importing into Pathfinder:
  - a. Determine the directory into which the FBX exporter saved the textures. Some exporters may place the textures in a sub-directory of the FBX file and give it the same name as the FBX file. Others may save the textures to a common program-specific location. The SimLab Revit exporter, for example, saves textures for a particular file to `C:\ProgramData\Autodesk\Revit\Addins\SimLab\FBXExporter\data\Imported_Textures\#` where `#` is a number specific to the exported file, such as `40`.
  - b. If the folder is not already a sub-directory of the FBX file, cut this folder and paste it in the same location as the FBX file. The pasted folder may be left as is or renamed to be the same as the FBX file, without the `.fbx` extension.
  - c. Transfer the FBX file and the texture folder to the computer that will be importing the FBX file into Pathfinder.
4. Import the FBX file into Pathfinder.

### 4.2.8.2. Revit to IFC (direct)

The first method exports a building information model (BIM) in the industry foundation classes

(IFC) format (see [Section 4.2.2](#)).


To perform the export in Revit 2019, perform the following:

1. Open the desired RVT file within Revit.
2. Under the **File** menu, click **Export** → **IFC**.
3. Choose the desired IFC filename.
4. Click **Modify setup** to choose different export settings, if desired.
5. Click **Export** to save the file.
6. Import the IFC file into Pathfinder.

#### 4.2.8.3. Revit to DWG (direct)

This method exports a DWG file directly from Revit, which can then be imported into Pathfinder. While simple to perform and only requires Revit, this method loses all information about materials, including textures, due to Revit's limited DWG support.


To perform the export in Revit Architecture 2014, perform the following:

1. Open the desired RVT file within Revit Architecture.
2. Click the Revit icon at the top left .
3. Select **Export** → **CAD Formats** → **DWG**.
4. In the **DWG Export** dialog, for **Export**, select <In session view/sheet set>.
5. For **Show in list**, select Views in the Model.
6. Click the **Check None** button, and then in the view table, select the check box for 3D View (Other views may be chosen, but the DWG will only contain entities visible in the selected views).
7. Click **Next** and choose a file name for the DWG file.
8. Click **OK** to create the DWG.
9. Import the DWG into Pathfinder.

#### 4.2.8.4. Revit to FBX (direct)

This method exports an FBX file directly from Revit, which can then be imported into Pathfinder. As with exporting a DWG, this method is simple to perform and only requires Revit. Unfortunately, this method also loses all information about materials and textures because Revit encrypts the material data, making it unreadable by Pathfinder.


To export using Revit Architecture 2014, perform the following:

1. Open the desired RVT file within Revit Architecture.
2. Click the Revit icon at the top left .
3. Select **Export** → **FBX**.
4. Choose a file name for the FBX file.
5. Click **OK** to create the FBX.
6. Import the FBX into Pathfinder.

#### 4.2.8.5. Revit to FBX to AutoCAD to DWG

This method requires both Revit and AutoCAD and does not perform a perfect conversion, but it retains some information about materials and texture coordinates.

The steps described here use Revit Architecture 2014 and AutoCAD 2014:

1. Open the desired RVT file within Revit Architecture.
2. Click the Revit icon at the top left .
3. Select **Export** → **FBX**.
4. Specify the desired filename and click **Save**.
5. Open AutoCAD.
6. On the **Insert** tab in the ribbon, select **Import**.
7. Select the FBX file created by Revit.
8. The **FBX Import Options** dialog will appear, see the recommended settings below.
9. Click **OK** to finish the import. You may receive a warning about the clip plane of the camera.
10. Save the file as a **DWG**.
11. Import the DWG into Pathfinder.

The following are recommended settings for the FBX import:

#### Import section

Make sure **Objects** and **Materials** are checked. **Lights** and **Cameras** are unused in Pathfinder.

#### Assign Objects to Layers

Any option may be selected, but **By Material** is a useful option for Pathfinder.

#### Unit Conversion

This section is somewhat misleading. While the **Current Drawing Unit** is correct, the **FBX file unit** tends to be incorrect. No matter what unit is displayed in the greyed-out text for **FBX file**

**units**, the actual unit in the FBX file is always FOOT. The appropriate values need to be specified to make the proper unit conversion. For instance, if the current drawing unit is Millimeters, you can enter the value, 1 on the left and 304.8 on the right because there is 1 foot per 304.8 millimeters.

### Block

Uncheck **Insert file as block**.

## 4.2.9. Importing Occupant CSV Files

See [Section 5.4.5](#) for more information.

## 4.3. Working with Imported Data

Each type of file that can be imported provides an aid for creating navigation geometry. The different types can be worked with in various ways to create the desired rooms, stairs, and doors.

### 4.3.1. Generate Model from BIM

The easiest way to create a complete Pathfinder navigation mesh, including rooms, doors, and stairs, is to use the **Generate Model from BIM** action. This action works best with imported IFC files, but it can work with other CAD file types as well, as long as those files contain 3D face data, such as from DXF, DWG, FBX, and GLB/gltf files. These non-IFC file types require some extra steps as outlined below.

To use this action, perform the following steps:

1. Import all CAD files that contribute to the model into the same Pathfinder model.
2. If any of the imported files were not IFC, set the **Import Type** for the objects imported from those files. These files default all objects to the **Import Type**, **Obstruction**, so this only needs to be done for non-Obstruction objects.  
Some fast ways to do this include:
  - a. If the imported objects are grouped together in the navigation view by type, select the group containing the objects such as the one containing all the doors, and set the **Import Type** in the property panel to Door.
  - b. If the imported objects contain the object type in their name, such as **Door43**, use the search tool (**Edit** → **Find**) to select all with the text **Door** in the name. Then set the **Import Type** to **Door** on the whole selection.
3. For any imported Obstructions that might come and go throughout the simulation or that should only slow down occupants rather than prevent movement, convert them to Obstacles as defined in [Section 3.4.1.3](#).

4. Delete any objects that should not contribute to the Pathfinder model or set their **Import Type** to **<ignored>**. This may be necessary, for instance, if the file contained stand-in objects, such as the building envelope.
5. In the **Model** menu, click **Generate Model from BIM**. This will show the **Generate Settings** dialog as explained in [Section 4.3.1.2](#).
6. Set the desired properties for generating the model, and click **Generate**. This will generate rooms, stairs, and doors for the entire imported model.
7. If any exits still need to be added, do so manually.
8. There may be extra detached rooms that are not needed in the model. An easy way to remove these is to first right-click a room that should be in the result, and click **Select Connected Components** from the right-click menu. Select the Entire graph option in the dialog, and click **OK**. Hide the selected objects. Repeat this until all required components have been hidden. The remaining extra rooms can be selected and deleted.

If only a subset of the imported objects needs to be modeled in Pathfinder, either set the **Import Type** for unnecessary objects to **<ignored>** before generating the model with the above steps or perform the following instead:

1. Select the imported objects that should be turned into navigation elements. The selection does *not* have to include objects with **Import Type** set to **Obstruction**, as obstructions are automatically subtracted during model generation.
2. Right-click the selected objects, and from the menu, click **Generate Model from BIM Selection**.
3. In the **Generate Settings** dialog, enter the desired settings and click **Generate**.

#### NOTE

When using the **Generate Model from BIM** actions, any existing rooms will be overwritten with newly generated rooms if they exist in the same space; however, existing doors and stairs will *not* be overwritten, which may result in duplicate doors and stairs.

#### 4.3.1.1. Generate Process

The following table describes how Pathfinder converts imported objects into Pathfinder navigation elements based on their **Import Type** and any additional properties that may have been imported along with the object:

**Table 6. Import types to Pathfinder Navigation Elements**

Import Type	Pathfinder Type	Details
<ignored>	—	These objects are completely ignored when generating the model.
Door	Door	To generate a Pathfinder door, Pathfinder first obtains the geometry of the imported door that will define the shape of the Pathfinder door. If the imported door has an associated wall opening, the wall opening's geometry is used. If not, the door's geometry is used. The minimum bounding rectangle of the geometry is then used to define the Pathfinder door shape. If the door's geometry was used and the door has an associated opening width property, the resulting door will be no wider than this opening width. The minimum bounding rectangle is then extruded into a box such that the bottom of the box is slightly below the bottom of the source geometry and the top is slightly above the top of the source geometry. This box is then subtracted from the generated rooms, and a Pathfinder door is used to fill the gap.
Escalator	Stair and Room	The conversion is performed the same as for <b>Import Type</b> Stair. If the resulting stair must be treated as an actual escalator, its properties must be set after generating the model (see <a href="#">Section 3.9</a> ).
Floor	Room	Pathfinder will identify the potential walking surfaces of the imported objects. It will then identify all potential obstructions and subtract them from the walking surfaces. NOTE: With the exception of objects with <b>Import Type</b> , Door and <ignored>, all imported objects are treated as obstructions to flow, even if they don't have <b>Import Type</b> , Obstruction.
Moving Walkway	Room	The conversion is performed the same as for <b>Import Type</b> Floor. If an actual moving walkway is needed instead of a room, delete the generated room and create a moving walkway as described in <a href="#">Section 3.10</a> .
Obstruction	—	These objects do not directly become Pathfinder objects. Instead they become either holes in the rooms or thin boundary walls.



Import Type	Pathfinder Type	Details
Ramp	Room	The conversion is performed the same as for <b>Import Type</b> Floor. If an actual ramp is needed instead of a room, delete the generated room and create a ramp instead as described in <a href="#">Section 3.8</a> .
Stair	Stair and Room	To generate a Pathfinder stair, the imported object's steps are first generated as if they were rooms. Like with rooms, overhead obstructions, such as railings, are subtracted from them. After this, the steps of the stair are connected together using Pathfinder stairs. If multiple steps can be strung together in a row and have similar rise and run characteristics, they all become one stair. Otherwise, they are separated into multiple stairs so that each stair has similar rise/run to the underlying imported geometry. This process may leave part of the stair geometry, such as the landings, as rooms.
Room	Room	The conversion is performed similarly to <b>Import Type</b> Floor. However, Pathfinder will seek to use only the bottom of the geometry as a walking surface. Rooms may also have custom parameters to set occupancy and generate occupants; these are exported by the Evac4Bim Plugin for Revit ( <a href="https://github.com/YakNazim/Evac4Bim/releases">https://github.com/YakNazim/Evac4Bim/releases</a> ).
Building	—	These objects do not directly become Pathfinder objects. Instead they provide additional information to create new occupant profiles and provide an initial delay for generated behaviors. These parameters are only exported by the Evac4Bim Plugin for Revit ( <a href="https://github.com/YakNazim/Evac4Bim/releases">https://github.com/YakNazim/Evac4Bim/releases</a> ).

#### 4.3.1.2. Generate Settings

When using the **Generate Model from BIM** action, various settings can be specified in the **Generate Settings** dialog as shown in [Figure 85](#).

Figure 85. Generate Settings dialog

The following settings are available in the **Generate Settings** dialog:

### Max Slope

Determines the maximum slope of a floor to be considered as a walkable surface. A slope of 0° indicates a flat surface, while a slope of 90° is vertical.

### Max head height

The maximum height for objects to be considered overhead obstructions. Overhead obstructions are used to cut out portions of the resulting rooms to prevent occupants from walking in that space. Imported objects higher than this will not be counted as obstructions.

### Generate from tops of solids

If checked, walkable surfaces will only be generated from closed, solid objects if the surface is on the top of the solid (i.e. there are no other surfaces in the object above it). This helps reduce the number of surfaces that will be considered for generating rooms, which also reduces the

computation time. The result should be the same, however, if this is checked or unchecked. If it seems like any rooms are missing from the result, try unchecking this option.

### Exclude rooms in solids

If checked, rooms that were generated inside solid objects such as walls are excluded from the result. This helps reduce the number of resulting rooms. If any rooms appear to be missing from the result, try unchecking this option.

### Close gaps smaller than x

If checked, thin boundary edges are inserted into a room anywhere where there are room boundaries close together. This is useful in combination with **Exclude rooms with area less than x** to reduce the complexity of the model.

### Exclude rooms with area less than x

If checked, rooms with small area are excluded from the result. This is useful in combination with **Close gaps smaller than x** to reduce the complexity of the model and remove portions where an occupant would not walk, such as between a desk and wall.

### Use imported names

If checked, the resulting navigation elements (rooms, doors, stairs) will be named after the imported object from which they derived. Otherwise, a generic name will be used.

### Generate doors

If checked, adds Pathfinder doors to the navigation mesh from the imported doors.

<b>NOTE</b>	Any imported object can be considered a door if its <b>Import Type</b> is set to Door.
-------------	--

### Generate stairs

If checked, adds Pathfinder stairs to the navigation mesh from the imported stairs.

<b>NOTE</b>	Any imported object can be considered a stair if its <b>Import Type</b> is set to Stair. In addition, a stair might be turned into multiple Pathfinder stairs, including single-step stairs if the imported stair does not conform to a single Pathfinder stair, as with spiral staircases.
-------------	---

### Max vertical step distance

The maximum allowed distance in the Z direction between steps of a stair. If an imported stair contains any consecutive steps further away than this, they will not be connected.

### Max horizontal step distance

The maximum allowed distance in the XY direction between steps of a stair. If an imported

stair contains any consecutive steps further away than this, they will not be connected.

### Min stair width

The minimum width of resulting stairs. If an area that looks like a stair is narrower than this distance, it will not result in a stair.

The following settings will only impact the handling of objects exported by the Evac4Bim Plugin for Revit (<https://github.com/YakNazim/Evac4Bim/releases>):

### Generate profiles

If checked, generates profiles based on properties of objects with **Import Type** Building.

### Generate occupants

If checked, adds occupants to the model based on properties of objects with **Import Type** Room. If any profiles were generated, a uniform distribution of those profiles is used for the resulting occupants; otherwise, the existing **Default** profile is used. If the **Default** profile has been deleted, the first existing profile in the model is used. The resulting occupants will also be assigned to a new Behavior that has a single Goto Any Exit action (see [Section 5.3.3.27](#)). If there are any objects with **Import Type** Building, the behavior's Initial Delay is set to the distribution defined in the building.

#### 4.3.1.3. Limitations

While the **Generate Model from BIM** action can quickly provide a good starting point for a Pathfinder simulation, it does have some limitations, including:

- Escalators are only supported as Pathfinder stairs. Their speed and direction are not determined during extraction.
- Moving walkways are only supported as Pathfinder rooms. Their speed and direction are not determined during extraction.
- Elevators are not currently supported.
- Generated doors may be slightly wider than the actual door opening. This is because the shape of the door is determined from either the imported door's associated wall opening or the door geometry. In the former case, the actual door opening may be slightly smaller than the wall opening due to the door jam. In the latter case, the door geometry often includes door trim, which may be several inches wider than the actual door opening. In addition, some CAD packages may export a curtain wall door with windows on the side as one door object. In this case, the resulting door may be quite a bit wider than the actual door opening because the wall opening/door geometry would include the side windows.
- Pocket doors might be twice as wide as they should be since their opening may include the space inside the walls.

- Doors might be thicker/deeper than the surrounding wall if the CAD package that exported the IFC file uses deep wall openings rather than openings that are just deep enough to cut out an opening in the wall. This should not significantly impact the simulation unless the opening is extremely deep.
- The presence of a door threshold may prevent a door from generating correctly.
- Pathfinder stairs are only generated from objects marked with the **Import Type, Stair**. Some objects, such as steps in a stadium might look like stairs but are actually marked as floors. In this case, stairs will not be automatically generated, and will look like small, disconnected rooms instead.
- Pathfinder may treat some objects as obstructions that should not participate in the extraction process, such as a stand-in object representing the building envelope. This is caused by limitations in the IFC format or the exporting CAD package, where these objects are marked with the entity type, **IfcBuildingElementProxy**. This type is merely a stand-in data type for objects not inherently supported by the IFC standard. They may or may not be physical objects in the model, and there is currently no way to differentiate, so Pathfinder treats them all as obstructions.

#### 4.3.1.4. Troubleshooting CAD Import

Depending on the imported CAD data, some problems may arise when generating the Pathfinder model. [Table 7](#) below may help determine the source of the problem and resolve it.

**Table 7. CAD Import Problem, Cause and Solution**

Problem	Cause	Solution
There is a wall in the middle of a generated room where there was no obstruction or there is a gap between rooms when they should be connected.	The wall/gap might actually be a very thin gap that exists between adjacent imported slabs. Pathfinder does not currently close these small gaps automatically.	Either draw a door along the edge that bridges the gap or adjust the geometry of the rooms and merge them together.
A generated door should connect two rooms but is an exit instead.	This can happen if the imported door had no associated wall opening, the door's geometry was thinner than the surrounding wall, and there is no slab or other floor object under the door.	Select the door and use the manipulation tool to connect the door to the other side of the door opening.


Problem	Cause	Solution
A generated door appeared where it wasn't expected.	This may happen if a single door was broken into several pieces and/or objects were marked as doors that weren't actually doors. For instance, this might happen if the door trim is modeled as a separate object than the actual door, and the trim is also marked as a door. In this case, the piece of trim is modeled as a separate door.	Delete the door and draw manually instead.
A generated door was too small.	This is usually caused by the same problem as above.	Select the door and change the <b>Width</b> in the property panel to the desired size or delete and redraw the door if this does not work properly.
A generated door was too wide.	This can happen due to limitations in the door extraction algorithm, see <a href="#">Section 4.3.1.3</a> .	Select the door and change the <b>Width</b> in the property panel to the desired size.
A door is missing	The imported object's <b>Import Type</b> was not set to <b>Door</b> , or the door had no walking surface under it, did not have an associated wall opening, and was thinner than the surrounding wall. This might also happen if there is a door threshold.	Draw the door manually.

Problem	Cause	Solution
A room is missing	This may either be because the imported object defining the floor did not have its <b>Import Type</b> set to <b>Floor</b> or the <b>Exclude rooms in solids</b> option was enabled and there is a solid object in the model marked as an obstruction that should not be in the model, such as one representing the building envelope.	Undo the import, ensure the imported object's <b>Import Type</b> is set to <b>Floor</b> , and delete any objects that should not be in the model. Then try generating the model again. If this still does not work, draw the room manually.
A stair is missing	The imported stair object was not marked as a Stair or the step distances exceeded the limits set in the generate settings.	Undo the import, set the <b>Import Type</b> to Stair, and then perform the <b>Generate Model from BIM</b> action again, or draw the stair manually.
An imported stair was broken into several Pathfinder stairs.	This can happen if an imported stair does not conform to a Pathfinder stair, such as with spiral stairs. This might also happen if there are multiple directions from which an occupant could step down off a step of the stair. Another cause could be that the step rise/run changed drastically from one step to another, such as if the top or bottom step does not align exactly with the connecting slabs.	If the stair should have been generated as a single Pathfinder stair, delete the generated stairs and draw the stair manually.

Problem	Cause	Solution
The steps of the generated stair do not match the steps of the imported stair exactly.	This can happen if the steps of the imported stair do not have a consistent step height or tread depth or the imported stair did not line up exactly with the slabs it was connecting. In this case, Pathfinder averages the rise/run of the resulting steps over the entire rise/run of the stair, which may cause the steps to be slightly out of alignment with the imported stair.	If desired, select the Pathfinder stair and change its step rise/run in the property panel.
An elevator is missing.	Generating elevators is not currently supported.	Create the elevator manually, see <a href="#">Section 3.11</a> .
An escalator has extra steps off of the side.	Escalators are generated as stairs. Pathfinder generates stairs using a generic algorithm where any flat surface can be treated as a step. If some of the geometry looks like it might be a step, Pathfinder will create a stair going to it.	Delete the extra stairs.

### 4.3.2. Extract Rooms Individually

While automatic model generation should work for most 3D CAD files, it may not always produce the desired result, the user might want more control over the rooms that are generated, or it might just be too prohibitive to set the **Import Type** for some non-IFC CAD files.

Pathfinder provides an **Extract Room** tool  that can address these issues. This tool allows rooms to be extracted individually using a flood-fill algorithm. While it may take more effort to extract all the necessary rooms, it does not require objects to have an **Import Type** specified unless an object is to be ignored.

To use the **Extract Room** tool:

1. For any imported objects that might come and go throughout the simulation or that should only slow down occupants rather than prevent movement, convert them to Obstacles as



defined in [Section 3.4.1.3](#).

2. Select the **Extract Room** tool from the drawing tool panel on the left of the 3D or 2D view. The tool properties are shown in [Figure 86](#).

### Max Slope

Refers to the maximum grade that a person can walk on. Only imported polygons that have a slope less than this will be included in the result.

### Max Head Height

Refers to the maximum height of all the agents that will be included in the simulation. This parameter is used to subtract overhead obstructions from the resulting room.

### Gap Tolerance

Provides some control in dealing with imperfections in the imported data. If walls are closer than the gap tolerance, Pathfinder will add an extra thin wall in that area, helping to split of rooms so the result does not bleed into undesired areas.

X:	<input type="text" value="1.311977 m"/>	Max Slope:	<input type="text" value="45.0 °"/>	<input type="button" value="Extract"/>
Y:	<input type="text" value="4.155801 m"/>	Max Head Height:	<input type="text" value="1.8288 m"/>	
Z:	<input type="text" value="0.0 m"/>	Gap Tolerance:	<input type="text" value="15.24 cm"/>	

**Figure 86. Property panel for the Extract Room tool**

2. Once the appropriate parameters have been chosen, either enter a location on the floor of the desired room into the property panel or click this point in the 3D or 2D view.
3. If this point does not have any overhead obstructions within **Max Head Height** and the point is on a polygon with a slope less than **Max Slope**, Pathfinder will march out from this point on the imported geometry's polygons until it finds the boundaries of the room. It will also subtract overhead obstructions within **Max Head Height** from the resulting room.

Below is an example of a 3D CAD model [Figure 87](#) and a room extracted from it [Figure 88](#).

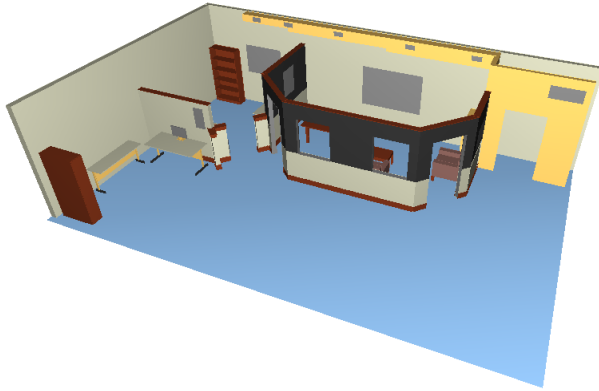


Figure 87. Example CAD model

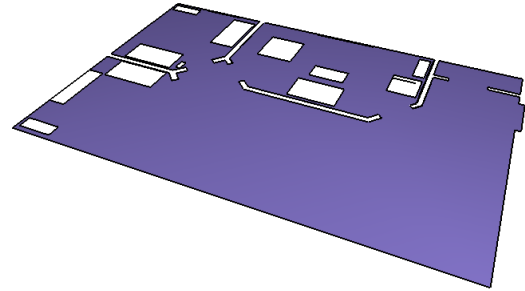


Figure 88. Room extracted from the CAD model


When using the extract room tool, Pathfinder will include all imported geometry with faces, even if hidden. If an object must be excluded from room extraction, the object's **Import Type** must be set to **<ignored>** before performing extraction. Otherwise, this tool completely ignores the **Import Type** of the objects.

### 4.3.3. Working with 2D CAD

2D CAD data can be worked with in two ways:

- It can be used as a guide for drawing rooms similarly to background images, except that drawing tools can snap to 2D CAD data.
- It can have rooms extracted from it similarly to 3D imported geometry as described above.

To sketch the rooms using the built in drawing tools, refer to [Section 3.2](#).

Extracting rooms from 2D CAD data works similarly to extracting rooms from 3D CAD data. As in 3D extraction, the **Extract Room** tool  is used. In addition, the user must click a point in the model with the tool, and one room will be extracted.

The main difference from 3D extraction is that the clicked point must not lie on any 3D imported faces marked for room extraction; instead it must be in empty space (or on the background rectangle imported with a 2D floorplan). The clicked point should also be surrounded by imported 2D lines. These lines will form the boundary of the resulting room. For this reason, any lines that do not contribute to the room's boundary, such as notations, symbols, etc. should be deleted, hidden, or excluded from room extraction prior to clicking the extraction point.

To manually exclude imported geometry from floor extraction, select the geometry and from the property panel, set the **Import Type** to **<ignored>**. When determining which imported geometry to extract rooms from, the 2D room extraction tool will automatically exclude hidden objects and those manually ignored.

Once the desired point is clicked, the surrounding 2D lines looking at the model from the **Top View** will be projected along the Z axis onto the active floor's working Z plane. These projected lines will then be used to define the room around the clicked point. If the surrounding lines do not form a closed boundary as shown in [Figure 89](#), the resulting room will spill outside of the lines and form a room around the bounding box of all the lines as shown in [Figure 90](#). In this case, this outer portion of the room can be separated from the inner portion using the **Thin Wall** tool as discussed in [Section 3.3.2](#). Once separated, the outer portion can be deleted.

When the extraction tool is finished finding a room, the room will lie in the working Z plane of the active floor.

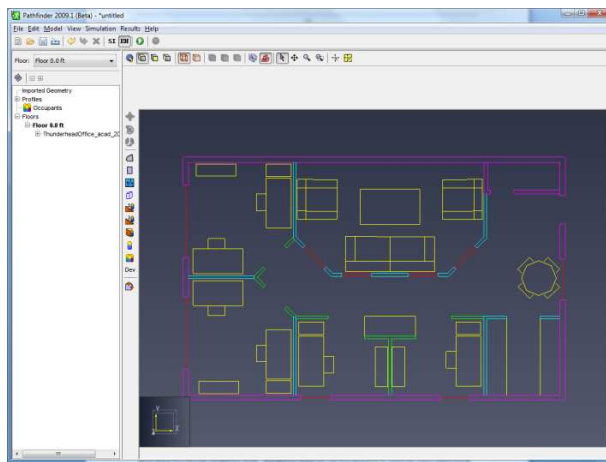


Figure 89. Imported 2D CAD drawing

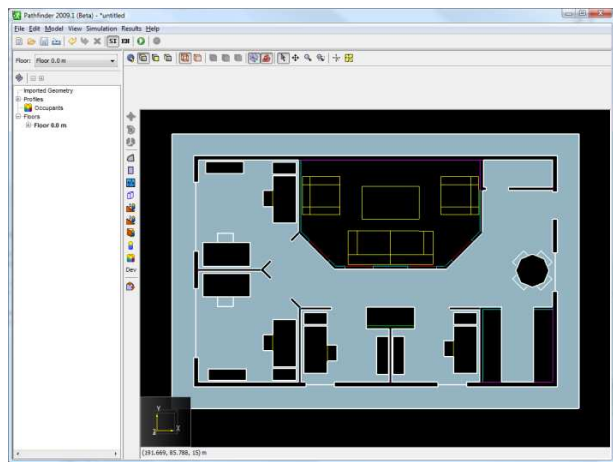


Figure 90. Floor extracted within bounding box.

#### 4.3.4. Working with Images

Working with background images requires the user to draw all rooms, doors, and stairs over the background image ([Figure 91](#)). Because the drawn navigation geometry will cover the background image, it may be preferable to make the navigation geometry transparent. This can be accomplished by selecting the drawn navigation components and lowering the opacity in the property panel. [Figure 92](#) shows a background image with rooms and doors drawn on top, with a lowered opacity for the drawn rooms.

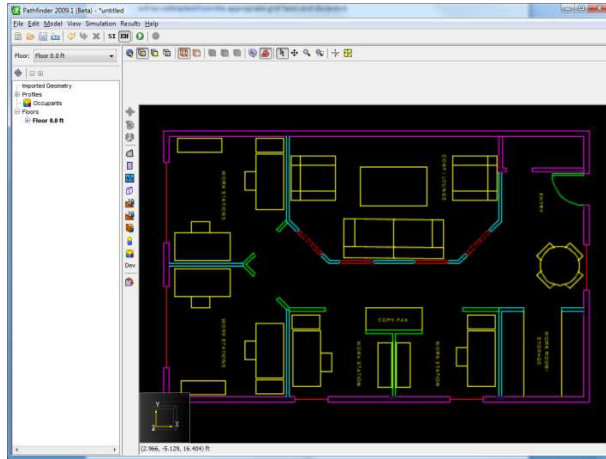


Figure 91. Imported background image

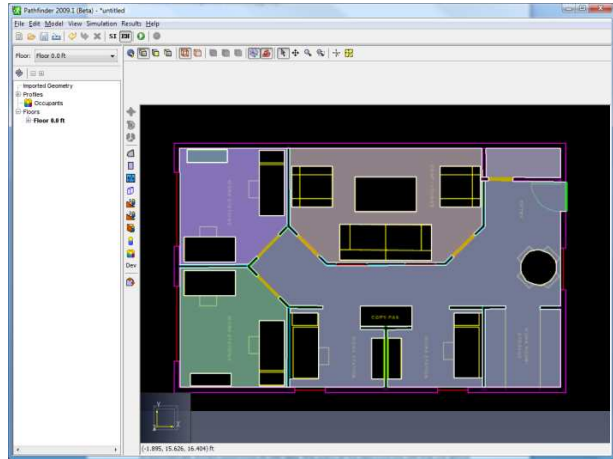


Figure 92. Rooms drawn over image

To draw rooms on top of a background image, refer to [Section 3.2](#).

### 4.3.5. Filling in Missing Pieces

Once rooms have been extracted using the **Extract Room** tool, the model will still be missing doors and stairs ([Figure 93](#)). Doors and stairs must be added manually as discussed in [Section 3.5](#) and [Section 3.6](#) of this guide.

One feature that may be of particular interest to help this process, however, is the internal door feature of the door tool. This feature automatically finds areas within a room that look like potential doorways and can be used to create a thick door in this area.

To use this feature:

1. Select the door tool.
2. In the property panel for the tool as shown in [Figure 55](#).

#### Max Width

Refers to the maximum width of the doorway to search for.

#### Max Depth

Refers to the maximum thickness of the doorway. These numbers may need to be larger than for the normal creation of a door to find potential doorways.

3. Once the appropriate parameters have been entered, move the cursor over the desired doorway ([Figure 94](#)). A door preview will be displayed. If it doesn't, adjust the search parameters in the property panel and try again.
4. If the door appears correctly, left-click the mouse button. The doorway area of the room will

be subtracted from the room, and the thick door will be created in its place (Figure 95).

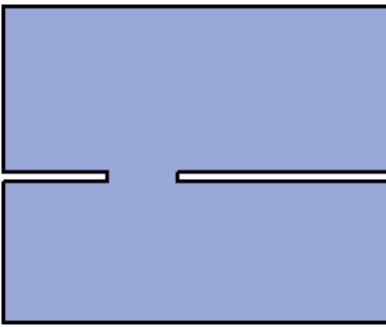


Figure 93. Doorway wall gap

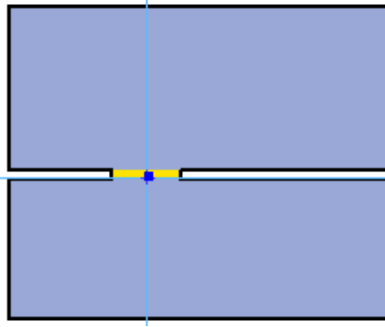


Figure 94. Click inside the gap

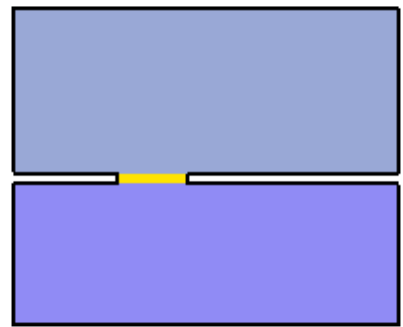


Figure 95. New door in the gap

### 4.3.6. Flattening and Z Location

Sometimes when importing from 3D CAD models, extracted rooms may not be at the proper height or may have some undesired slope. These issues can be corrected either before or after extracting rooms by using the **Set Z** dialog (Figure 96).

To do so, follow these steps:

1. Select the objects that need to be corrected.
2. Right-click the selection and select **Set Z**.
3. This will open the **Set Z** dialog as shown in Figure 96.
4. Set the desired options, and select **OK** to modify the geometry in the selection.

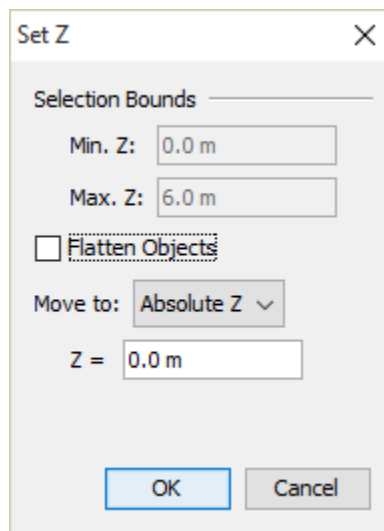


Figure 96. Set Z dialog

The dialog shows the following properties and options:

**Selection Bounds**

Shows the minimum and maximum Z locations of the selected objects.

**Flatten Objects**

If selected, each object in the selection will be flattened so that all of its geometry lies in the same plane. The location of the plane is determined by the **Move to** option.

**Move to**

This specifies how the geometry should be moved in the Z direction. This can be one of the following values:

**Absolute Z**

All the selected objects will be moved to the same Z plane. If **Flatten Objects** is not selected, each object is moved such that its minimum Z touches the specified plane.

**Z Offset**

Each object in the selection is moved by the specified **Z Distance**. If **Flatten Objects** is selected, the object is first flattened to the plane that touches the minimum Z of the object's original geometry, and then moved by the **Z Distance**. [stem  
e037678c24938fb2c2e03319096ae81f]

### 4.3.7. Materials

Materials define advanced display properties that can be applied to faces contained in the imported geometry. They are only shown when the **Realistic** or **Realistic with Outlines** option is selected in the **2D View** or the **3D View**, see [Section 2.3](#). Materials can be shared among faces; when a material is edited, all faces referencing that material are updated.

Materials are extracted from import files in different ways, depending on the file type:

**DWG, FBX, DAE, OBJ, glTF, GLB**

These files have a concept of materials. Each material that is referenced by an object in the file will be imported into Pathfinder.

**FDS and PSM**

Pathfinder materials are constructed from the color and texture settings of the *surfaces* in these file types.

**NOTE**

The use of the word "material" in Pathfinder is different than in PyroSim and FDS. In these applications, a material defines the *physical* properties of a substance. In Pathfinder, a material defines the *visual* appearance of a face.

To see the materials that have been imported from the DWG or PSM file, on the **Model** menu, select **Manage Material Database**. The **Material Dialog** will appear as shown in [Figure 97](#).

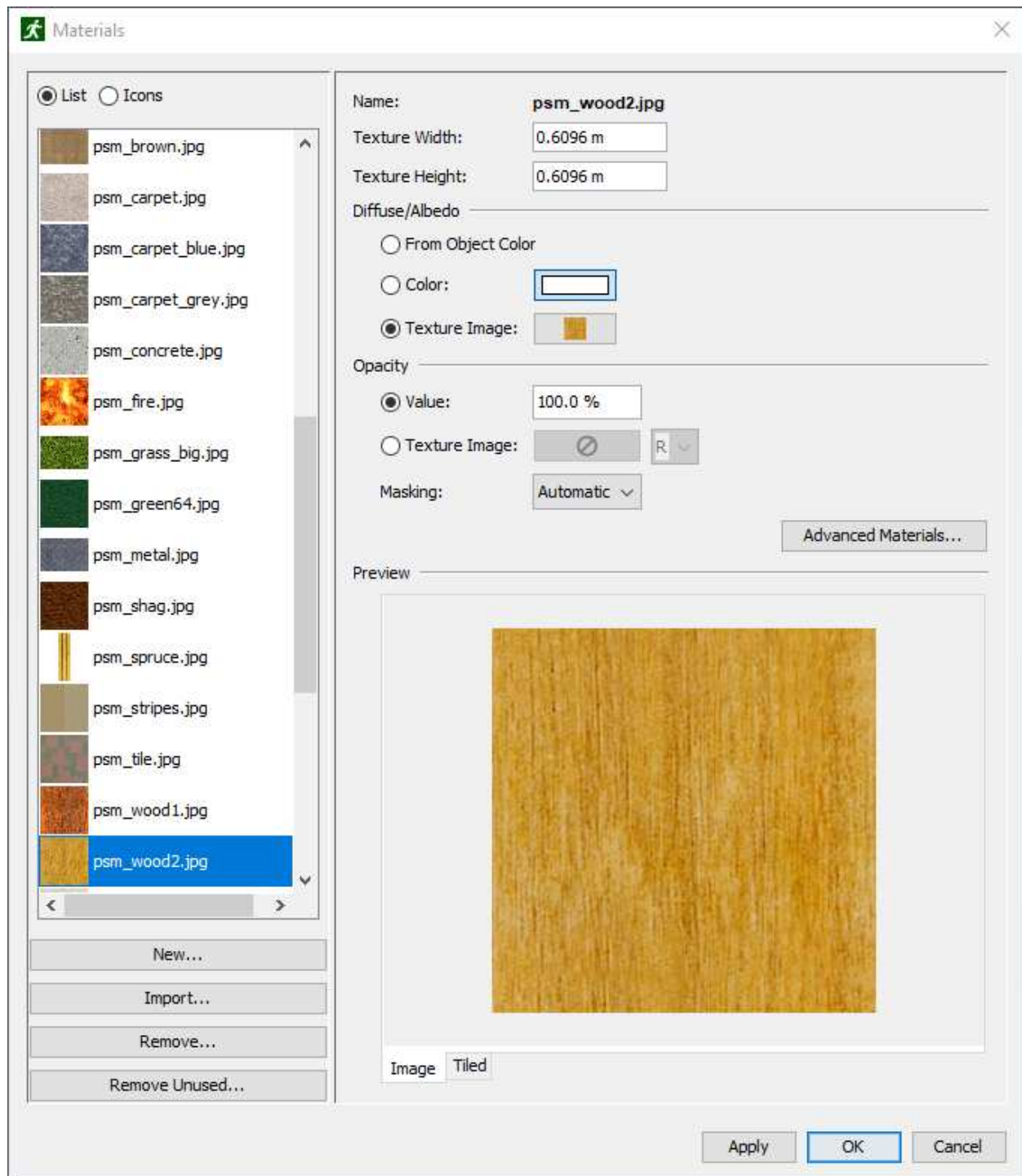


Figure 97. Materials dialog

Pathfinder provides some default database materials. Most of these materials start with the prefix, **psm\_** as in PyroSim. Other materials were either created manually by the user or were imported with the CAD or PyroSim file.

Materials can be added manually by clicking **New** under the material list. Materials can also be



created from an initial texture image on disk by clicking **Import**. The image is copied into the database directory. Newly created materials are added to the database, and can be used across instances of Pathfinder.

Materials can be deleted by clicking **Remove** under the material list. If the material exists in the database, all its associated files in the database directory will also be permanently removed.

The following material properties can be edited from this dialog:

### **Texture Width and Height**

Define the dimensions of the texture image in the model. For instance, if the image depicts a 4x4 array of bricks with no mortar, and each brick is 8"x3", the **Width** would be 32" and the height would be 12".

### **Diffuse/Albedo**

Defines the color of the object.

### **From Object Color**

Uses the color and opacity settings from the objects using this material. Selecting this option allows multiple objects to use the same material and have the same advanced material effects but with different colors. For instance, multiple objects can have the same bumpiness and roughness, but have different colors using one material.

### **Color**

Sets the color of all objects using the material to a specific value.

### **Texture Image**

Causes all objects using the material to display a texture image.

### **Opacity Value**

Defines the opacity of the material.

### **Opacity Texture Image**

Defines a texture which controls the opacity of the material.

### **Masking**

Specifies whether to enable masking for the material. Masking is a feature that treats rendered pixels as either opaque or completely transparent. This can help improve the edges of foliage, text decals, and other opaque, decal-like materials. When enabled, pixels with opacity values  $\geq 0.5$  are drawn as opaque and those with opacity  $< 0.5$  are not drawn. The following masking options are available:

### **Automatic**

Masking will be enabled automatically if the opacity texture or the alpha channel of the diffuse texture contains both opaque and completely transparent pixels. In most cases, this is a safe option to use.

### **Enabled**

Masking is enabled. This may need to be set if the edges of some objects have halos around them or appear fuzzy when using the **Automatic** option.

### **Disabled**

Masking is disabled. This may need to be set if a material that should be translucent appears opaque when using the **Automatic** option.

## **4.3.8. Advanced Material Properties**

Additional material properties can be edited by pressing the **Advanced Materials** button. This will show the **Advanced Materials Dialog** for the currently selected material.

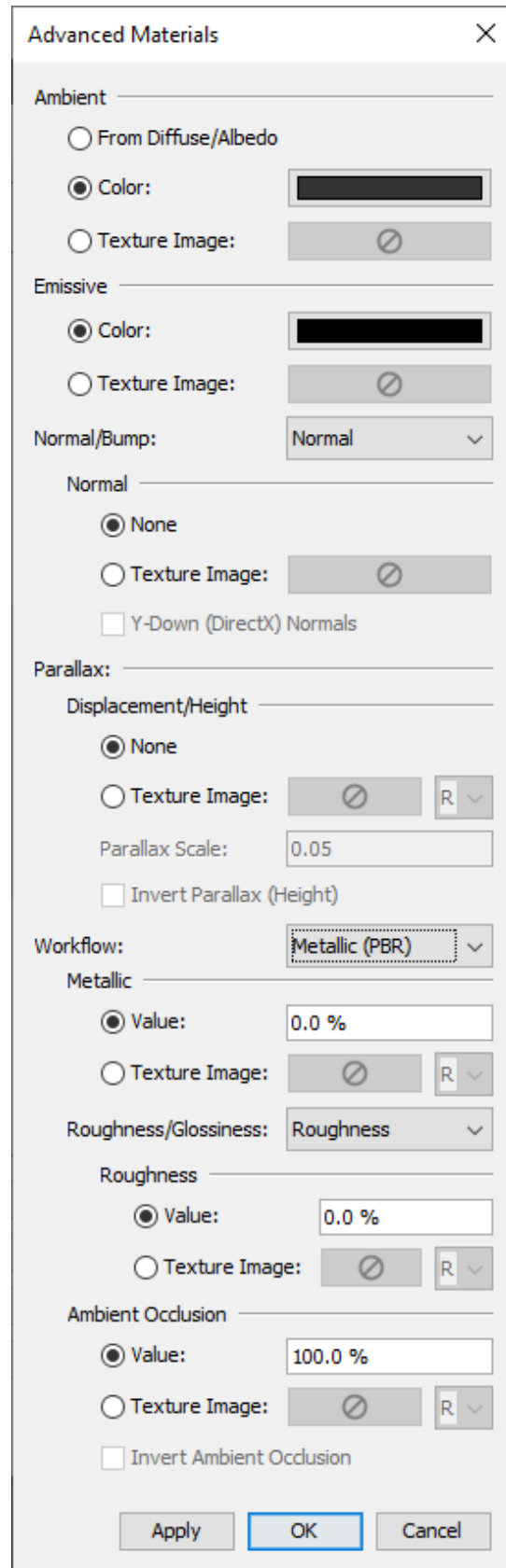


Figure 98. Advanced Materials dialog

Most properties in this dialog can be specified as either a constant color/value or as a texture image. For those properties that represent a single value as opposed to a color, such as the **Roughness** value for PBR workflow, a dropdown next to the texture chooser can be used to pick which color component is the source of the values. This is useful when multiple properties are packed into a single image, but in different color components. For instance, say the PBR parameters, metallic, roughness, and ambient occlusion, are packed into the red, green, and blue color components of a single image. In this case, the same image can be chosen for the metallic, roughness, and ambient occlusion textures. The dropdowns next to each texture would be set to **R**, **G**, and **B**, respectively.

The following material properties can be edited from this dialog:

**Ambient**

Defines the color used to modulate ambient lighting applied to the material. This will only be applied when Image Based Lighting is disabled. Selecting the **From Diffuse/Albedo** option for this channel causes the Diffuse channel value to modulate ambient lighting instead.

**Emissive**

Defines the amount and color of light emitted by the material.

**Normal/Bump**

Whether to use normal mapping or bump mapping for the material.

**Normal**

The normal map used for the material.

**Y-Down Normals**

Whether or not to invert the y-component of the provided normals. This should be enabled for normal maps designed for DirectX. Check any documentation that comes with the material to determine if this applies.

**Bump**

The bump map used for the material.

**Bump Scale**

The strength of the bump map effect.

**Displacement/Height**

The parallax displacement or height map to use.

**Parallax Scale**

The strength of the parallax mapping effect. This unit of this scale is purely visual, but it should be kept low to prevent rendering artifacts.

**Invert Parallax**

Whether or not to invert the displacement/height map. This should be enabled for height maps.

**Workflow**

The material workflow to use. Provided options are Specular (Basic), Metallic (PBR), and Specular (PBR). Materials with a Specular map are designed for either the Specular (Basic) or Specular (PBR) workflow. Specular materials with a Roughness, Glossiness, or Ambient Occlusion map are meant for the Specular (PBR) workflow. Materials with a Metallic map are designed for the Metallic (PBR) workflow.

Depending on which workflow is selected, the following additional properties are provided:

**Specular**

Defines the specular reflectance of the material. This property is only available in the Specular (Basic) and Specular (PBR) workflows.

**Metallic**

Defines whether or not the material is metallic. This property is only available in the Metallic (PBR) workflow.

**Shininess**

Defines how shiny the material's surface is. This property is only available in the Specular (Basic) workflow.

**Roughness/Glossiness**

Determines whether to use roughness or glossiness to define the material's microsurface. This option is only available in the Metallic (PBR) and Specular (PBR) workflows.

**Roughness**

Defines how rough a material's microsurface is. This is the exact inverse of Glossiness. This option is only available in the Metallic (PBR) and Specular (PBR) workflows.

**Glossiness**

Defines how glossy a material's microsurface is. This is the exact inverse of Roughness. This option is only available in the Metallic (PBR) and Specular (PBR) workflows.

### Ambient Occlusion

Provides additional shading information for the material. Note that this property is not related to the scene-wide Ambient Occlusion in Results. This property is only available in the Metallic (PBR) and Specular (PBR) workflows.

#### Invert Ambient Occlusion

Whether to invert the values read from the ambient occlusion texture (`ao_inverted = 1.0 - ao_texture`). This is useful when the ambient occlusion texture specifies the amount of occlusion (increasing values lead to more occlusion) rather than amount of visible light (lower values lead to more occlusion).

### 4.3.9. Reorganizing and Making Fast Edits

Sometimes the imported data may not be organized in a convenient manner. For instance, it might be desirable to change some feature of all the windows, but the windows in the model may not be in the same group, making it difficult to select all of them at once.

In cases such as these, the similar objects may have the same color. If so, right click one of the objects, and choose **Select All by Color**. Alternatively, choose **Select All by Material**. This will find all objects with the same color or material and select them, making it easy to change some shared property or move them into another group for easy selection later.

## 4.4. Importing FDS Output Data

Pathfinder can use the PLOT3D data output from FDS to create time history data for each occupant as they move throughout the simulation. In cases where FDS PLOT3D output data is available for [stem f74275c195a72099e0b06f584fd5489a] Volume Fraction, [stem 8306272097a2ac71e3c568c80d48b7f9] Volume Fraction, and [stem adae461c37eccd2f2a6ed21ebf2c5d08] Volume Fraction; Pathfinder will also output Fractional Effective Dose (FED) for each occupant specified. In cases where FDS PLOT3D output data is available for SOOT Visibility, occupants maximum speed can be reduced to simulate movement through smoke (see [Section 5.1](#)).

Enabling this feature causes simulations to require additional runtime because of the additional processing load relating to reading the FDS output files and mapping PLOT3D data to occupants.

To enable FDS Integration:

- On the **Simulation** menu, click **Simulation Parameters**.
- On the **FDS Data** tab, select **Enable FDS Integration**.
- Click **Edit** and select the SMV file from the FDS simulation of interest.

The dialog (Figure 99) will display information about the attached SMV file and indicate which quantities were found.

#### NOTE

For convenience, the path to the SMV file is stored internally as a relative path. This means that, so long as the *.pth* and *.smv* files are in the same locations relative to each other, you can move the files in your filesystem without affecting the FDS integration.

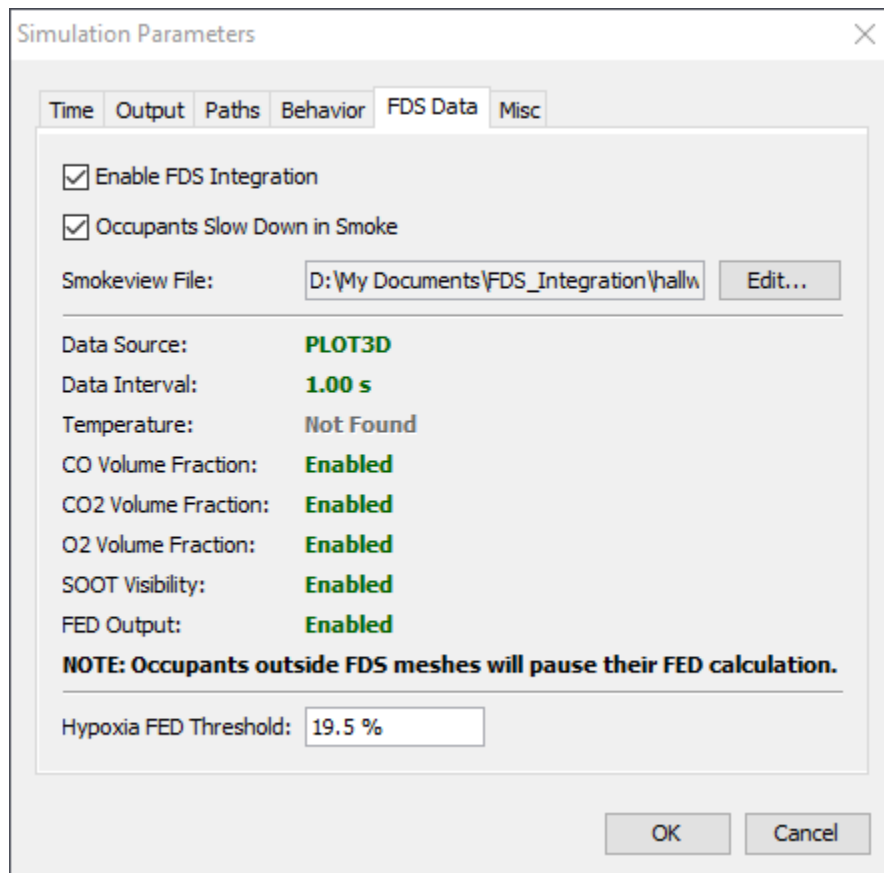


Figure 99. The Solution Parameters dialog after enabling FDS Integration.

### Measuring Occupant FED

The parameter, **Hypoxia FED Threshold**, defines the oxygen concentration at which hypoxia contributes to the FED calculation. When oxygen concentration is above the threshold, low [stem adae461c37eccd2f2a6ed21ebf2c5d08] hypoxia will not contribute to accumulated FED. The default value of 19.5% prevents misleading FED accumulation for occupants that are in safe conditions. (*Pathfinder Technical Reference*, n.d.)

To enable FED and PLOT3D quantity output for one or more occupants:

1. Select one or more occupants

2. In the selection editor, click **More**
3. In the **Additional Occupant Properties** dialog, on the **Output** tab, enable **Output Detailed Data**
4. In the **Output Detailed Data** dropdown, select **Yes**

Once the simulation has completed, output data is available for each specified occupant in the output folder (see [Section 16.9](#)).

For information on how FED is calculated or how its use was verified in Pathfinder, see ([Pathfinder Technical Reference, n.d.](#)) and ([Pathfinder Verification and Validation, n.d.](#)) documents respectively.

## 4.5. Importing Custom Avatars

When viewing occupants as people in either Pathfinder or Pathfinder Results, occupants and vehicles are represented by a 3D avatar. Avatars are chosen in either the **Edit Profiles** dialog (see [Section 5.1](#)) or the **Edit Vehicle Shapes** dialog (see [Section 5.2](#)). While Pathfinder ships with many avatars, they might not cover all modeling scenarios. Custom avatars can be imported into Pathfinder to meet additional requirements.

### 4.5.1. Custom Avatar Requirements

The following file formats are supported for importing custom avatars, listed in order of preference with the most preferred first:

#### Occupant Avatars

GLB, glTF, FBX, DAE

#### Vehicle Avatars

GLB, glTF, FBX, DAE, OBJ

Avatar files should follow these rules if possible:

- There should be one avatar per file.
- When using an FBX file, if possible, the textures should be embedded in the file; otherwise, the textures will have to be manually copied to the avatar directory (see [Section 4.5.5](#) below).
- GLB files are preferable over glTF files, as they usually embed all necessary data in a single file. glTF will require copying referenced texture and data files to the avatar directory manually.
- glTF and GLB are preferred over other formats, as they natively support PBR materials.



- Other file formats require copying textures manually to the avatar directory.

Occupant avatars have the following additional constraints:

- The avatars should be in a T-pose or an A-pose or some variation. For FBX and DAE files, this is accomplished by having at least one "animation" where the avatar is in this pose. There can be other animations in the file, but this will adversely affect loading performance.
- The avatars must be "rigged", meaning there is a joint hierarchy specified in the file. The rig must be human-like.
- Joint names in the file should follow standard conventions used by industry-wide software for best compatibility.

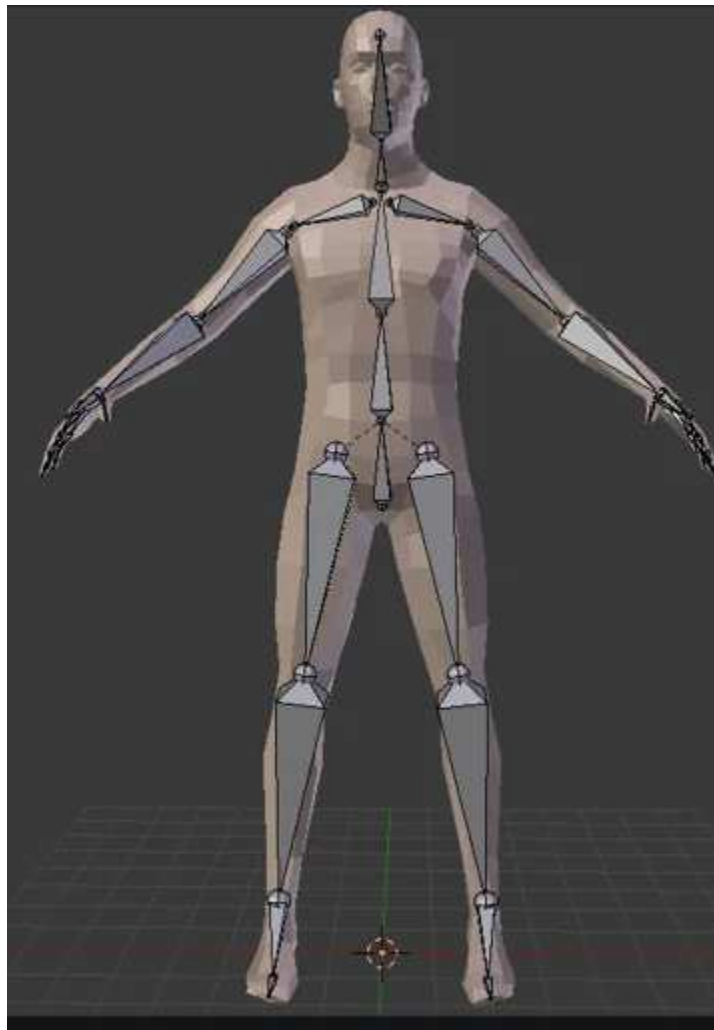


Figure 100. Example of a "rigged" human character model in the "A" pose in Blender.

### 4.5.2. Where to Find Custom Avatars

There are a variety of sources for custom avatars. They can either be purchased or downloaded online from third-party vendors specializing in 3D models or they can be created using third-party tools. Pathfinder has been successfully tested with avatars provided from the following sources:

#### MakeHuman Community

An open source tool for creating 3D characters. Available at <http://www.makehumancommunity.org/>.

#### Adobe Mixamo

A free online source for high quality character models and animations. This service provides mostly fantasy characters, but it has some more realistic options as well. Available at <https://www.mixamo.com/>.

### 4.5.3. Importing a Custom Occupant Avatar

To import a custom occupant avatar, perform the following steps:

1. On the **Model** menu, click **Edit Profiles**.
2. In the **Edit Profiles** dialog, click the list of avatars next to **3D Model** to bring up the **3D Model** dialog.
3. In the **3D Model** dialog, click the **Import** button.
4. Choose the desired file that contains the occupant avatar to import. This will copy the avatar file along with some other supporting files into `%APPDATA%/Pathfinder/models/md5/avatarname`, where `avatarname` is the name of the avatar file.

#### NOTE

By default, imported avatars are only available to the current user. See [Section 4.5.7](#) for more information on sharing these avatars.

### 4.5.4. Importing a Custom Vehicle Avatar

To import a custom vehicle shape avatar, perform the following:

1. On the **Model** menu, click **Edit Vehicle Shapes**.
2. In the **Edit Vehicle Shapes** dialog, click the list of avatars next to **3D Model** to bring up the **3D Model** dialog.
3. In the **3D Model** dialog, click the **Import** button.
4. Choose the desired file that contains the vehicle avatar to import. This will copy the avatar file

along with some other supporting files into `%APPDATA%/Pathfinder/models/props/avatarname`, where `avatarname` is the name of the avatar file.

**NOTE**

By default, imported avatars are only available to the current user. See [Section 4.5.7](#) for more information on sharing these avatars.

### 4.5.5. Fixing Avatar Issues

After importing a custom avatar, it may need further adjustments in order to display correctly. The following is a list of recommendations for fixing avatar issues:

**Table 8. Custom Avatar Issues**

Problem	Cause	Fix
Avatar has missing textures.	Avatar file was an FBX without embedded textures, or an OBJ, DAE, or glTF file.	Copy image and mesh data files from original avatar location to target avatar location under <code>%APPDATA%/Pathfinder/models</code> .
Some or all of occupant avatar body parts do not move during animations.	The file might not be rigged or might not follow common joint naming conventions.	If the file is not rigged, there is currently no fix. If it is rigged, however, apply a joint name map in the avatar's JSON file as described in <a href="#">Section B.3.3</a> .
Avatar is facing the wrong direction.	Avatar in originating file was not facing the front.	Apply a <code>mesh::transform::rotate</code> transform in the avatar's JSON file as described in <a href="#">Section B.1</a> .
Avatar is too big/too small.	Avatar was not modeled with realistic units or the units were wrong/unspecified in the avatar file.	Adjust the <code>mesh::transform::scale</code> factor in the avatar's JSON file as described in <a href="#">Section B.1</a> .
Avatar is offset from the actual occupant location.	Avatar was not located at the origin in the source avatar file with feet on the ground.	Apply a <code>mesh::transform::offset</code> transform in the avatar's JSON file as described in <a href="#">Section B.1</a> .
Avatar appears to slide/skate when walking.	Avatar has different proportions than those assumed by the importer.	Override Pathfinder's provided move animation in the avatar's JSON file and adjust the <code>move_clip::naturalSpeed</code> property as described in <a href="#">Section B.1</a> .

Problem	Cause	Fix
Other	Avatar type is unsupported by Pathfinder.	Contact <a href="mailto:support@thunderheadeng.com">support@thunderheadeng.com</a> to report.

#### 4.5.6. Avatar Performance Considerations

The avatars that ship with Pathfinder have been optimized for use in Pathfinder Results, allowing many thousands of occupants to be displayed onscreen at once with good performance. Custom avatars are *not* optimized for Pathfinder Results, as they contain no level-of-detail information and often contain much more detail than is necessary for displaying thousands of occupants. Using custom avatars will likely result in reduced results display performance compared to using avatars supplied by Pathfinder.

Currently the only ways to mitigate this problem are to either limit the number of occupants using the custom avatars or use custom avatars that are designed with low polygon counts and/or low-resolution textures.

#### 4.5.7. Making Avatars Available to Other Pathfinder Users

When avatars are imported into Pathfinder, they are copied into the `%APPDATA%/Pathfinder/models/` location, which limits their use to the current user on the current machine. If the results of a Pathfinder model containing these custom avatars are loaded by another user or on another machine, the occupants using the custom avatars will look like the Mannequin model that ships with Pathfinder. To make these custom avatars available to others users, perform the following from the computer and user account from which the avatars were imported:

1. In File Explorer, navigate to the following location: `%APPDATA%/Pathfinder/models`
2. If copying occupant avatars, navigate to `md5`. If copying vehicle avatars, navigate to `props`.
3. Copy the desired avatar folders.
4. On the machine on which the avatars are to be made available, in File Explorer, navigate to the following: `%PROGRAMDATA%/Pathfinder/models`. If this location does not yet exist, create it.
5. If copying occupant avatars, create/navigate to `md5`. If copying vehicle avatars, create/navigate to `props`.
6. Paste from the clipboard.

#### NOTE

If you want to make all custom avatars available to other users, copy the `%APPDATA%/Pathfinder/models` location into `%PROGRAMDATA%/Pathfinder` on the target computer.

## 4.6. Importing Custom Animations

When viewing occupants as people in Pathfinder Results, occupant motion is driven by skeletal animations. Animations are chosen based on tags specified in the **Edit Profiles** dialog (see [Section 5.1](#)). While Pathfinder ships with basic animations including walking and standing idle, these might not cover all modeling scenarios. Custom animations can be imported into Pathfinder to meet additional requirements.

### 4.6.1. Custom Animation Requirements

Typically, animations are designed in 3D modeling software for a specific avatar or are created from motion capture from an actor in a studio. The avatar/motion capture model is typically represented as a skeleton with a joint hierarchy. The animation data is simply transformations of the skeleton's joints over time.

Animations can be stored in various file formats, such as **FBX**, usually exported from 3D modeling software. Depending on the file format, the animation file may be able to contain a single animation or multiple and may optionally include the avatar for which the animation was created. Repositories of animations can be found online, such as <https://www.mixamo.com/>.

While each animation is designed for a specific avatar, once it is imported into Pathfinder it can be used with *any* avatar. This happens through a process called **retargeting**, where Pathfinder will analyze the animation along with some information about the originating avatar, and then adjust the animation to better match the target avatar, even if the target has vastly different limb proportions or features than the original.

In order for this retargeting process to work well, animation files should follow these rules if possible:

- Animations must be stored in an **FBX** file.
- There should be one animation per file. If there are more than one, only the first animation containing at least 2 frames of animation data is used.
- The avatar for which the animation was created must be "rigged", meaning there is a joint hierarchy specified in the file. The rig must be human-like.
- Joint names in the file should follow standard conventions used by industry-wide software for best compatibility.
- The animation file may optionally include the originating avatar as well as the animation. If it does, the animation file is the only file necessary to import into Pathfinder; however, the file may be quite large. If the file does *not* contain the avatar, a separate file containing the avatar will also be necessary to import into Pathfinder.

**NOTE**

If importing multiple animations, it may be desirable to save the animation's avatar separately in its own file, and only store the animations (without avatar) in the animation files. This can help reduce file size and initial load times for the animations.

### 4.6.2. Where to Find Custom Animations

There are a variety of sources for custom animations. They can either be purchased or downloaded online from third-party vendors specializing in 3D animations or they can be created using third-party tools. Pathfinder has been successfully tested with animations from the following sources:

**Adobe Mixamo**

A free online source for high quality character models and animations. This service provides mostly fantasy characters, but the supplied animations are quite diverse. Available at <https://www.mixamo.com/>.

**NOTE**

When using this service, it's recommended that the avatar/character be downloaded once, and then animations downloaded **Without Skin**. This will help reduce the file size and load times for the animation files.

### 4.6.3. Importing a Custom Animation

To import a custom animation into Pathfinder, perform the following steps:

1. On the **Model** menu, click **Manage Animation Database**.
2. In the **Animations** dialog, click **Import** and choose the animation file to import.
3. In the **Import Animation** dialog, enter the required properties as defined below.
4. Click **OK** to import the animation. The animation **Clip** and **Base Pose** files are copied into **%APPDATA%/Pathfinder/models/anim/clips** and **%APPDATA%/Pathfinder/models/anim/meshes**, respectively.

**NOTE**

By default, imported animations are only available to the current user. See [Section 4.6.4](#) for more information on sharing these animations.

In the [Animation Datatabase Dialog](#) animations can be imported, renamed, and deleted. By default, the list on the left shows all imported animations; however, the field above the list can be used to filter the list by tag. For instance, enter **upright** and press **ENTER** to see all animations containing the **upright** tag. Add additional tags to further refine the search.

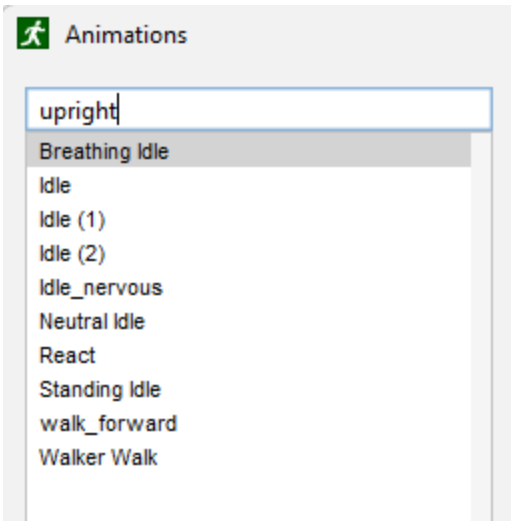


Figure 101. Filtered Animation List

The following properties can be entered in the [Import Animation Dialog](#) and the [Animation Database Dialog](#):

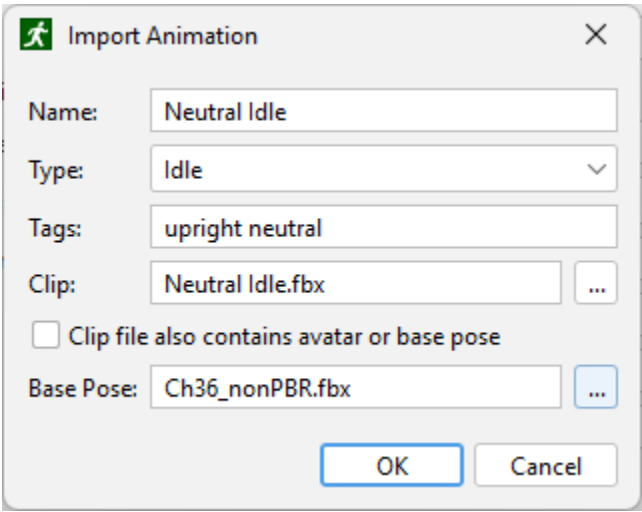


Figure 102. Import Animation Dialog

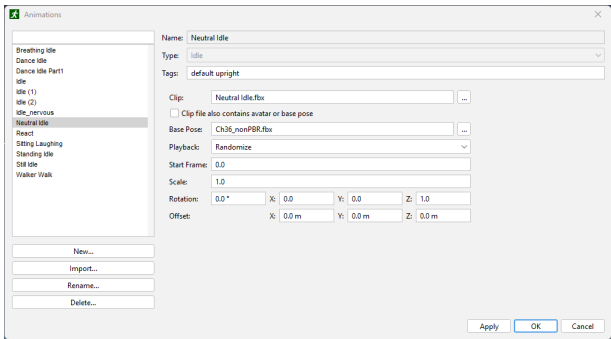


Figure 103. Idle Animation Dialog

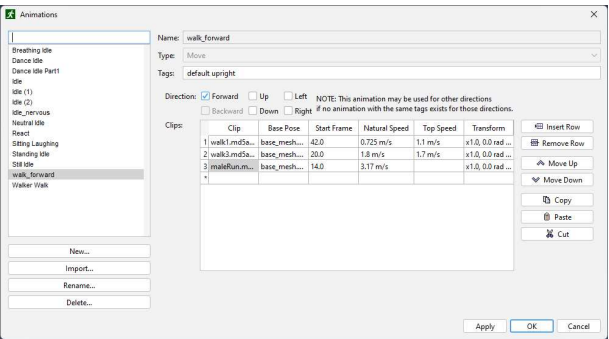


Figure 104. Move Animation Dialog

**Name**

The name of the animation. When importing an animation, this is matched to the name of the animation clip, but it can be any useful name, as long as it is unique.

**Type**

The animation type, which may be one of the following values:

**Idle**

An animation that can be used when an occupant is idling (not actively seeking a destination).

**Move**

An animation that can be used when an occupant is seeking a destination.

**Tags**

A list of labels used to identify the animation that are matched against an occupant's **Animation** property (see [Section 5.1.5](#)). When multiple tags are specified, the animation is associated with all combinations of the tags. For instance, if an animation was tagged with **upright**, **bored**, and **fidgeting**, it could be matched with the following tag combinations:

- **upright**
- **bored**
- **fidgeting**
- **upright,bored**
- **upright,fidgeting**
- **bored,fidgeting**
- **upright,bored,fidgeting**

**Clip**

The file containing the animation.

**Clip file also contains avatar or base pose**

Indicates that the **Clip** file also contains a base pose for the animation's avatar. A base pose is one in which the avatar is standing, facing the -Y axis in either a T-pose or A-pose (see [Section 4.5.1](#)). This pose can be defined in one of two ways:

- The animation file also contains the avatar itself.
- The animation file contains two animations. The first is a single-frame animation defining the base pose. The second is the actual animation.



**Base Pose**

Defines a separate file containing the base pose for the animation's avatar. Usually, this is a file containing the avatar itself. This is only needed if **Clip file also contains avatar or base pose** is unchecked (i.e. the **Clip** doesn't also contain the base pose).

**Scale**

Defines a scaling transform to be applied to the animation's joint locations so that the animation's units are in meters. Typically this can be left at **1.0**, as the joints will be automatically scaled to match each avatar, but this may need to be modified if the animation was authored for a very small or large avatar and the animation is unacceptable in Results.

**Rotation**

Defines a rotation transform for the animation about the specified axis using the right-hand rule. This should be adjusted such that the avatar is facing the -Y axis. For instance, if avatars appear to be rotated 90 degrees to the right of their expected orientation when using this animation, specify a rotation of **90°** about the **+Z** axis (i.e. **Rotation=90°, X=0,Y=0,Z=1**).

**Offset**

Defines an offset to apply to the animation. This should be adjusted such that the animation's skeleton is centered at the origin in X and Y and is located at the proper Z height. For instance, if avatars appear slightly lower than expected when using this animation, such as their feet are below the floor when standing, increase the Z offset.

**4.6.3.1. Idle Animations**

Idle animations are used when an occupant is still, such as when waiting in a queue or using a [Wait Behavior Action](#). They have the following additional properties as seen in [Figure 103](#):

**Playback**

Defines how the idle animation is played. This can be one of the following values:

**Randomize**

The animation will start at a random frame and repeat indefinitely. This may be useful for idle animations when you want the animation to loop, and it doesn't matter at which frame the animation starts.

**Play Once**

The animation will play once from the beginning and then hold the last frame. This may be useful for animations representing specific non-repeating actions, such as a customer paying for a good.

**Repeat**

The animation will play from the beginning and repeat indefinitely. This may be useful for animations that should loop, and the starting frame is important, such as an avatar clapping their hands at a performance.

**Start Frame**

Remaps the specified frame as the first frame of the animation. This is important if the **Playback** is **Play Once** or **Repeat**, as it defines the starting frame.

**4.6.3.2. Move Animations**

Move animations are used when an occupant is actively moving toward a destination. They are defined slightly different than **Idle** animations, as they also have associated speed and direction.

Each move animation represents the occupant moving in a single direction, relative to the occupant's orientation. Only a single direction needs to be specified for a given set of tags, but more can be defined to improve realism in Results. When a direction is defined, the same animation (but reversed) will be used for the opposite direction if the opposite direction isn't explicitly defined. For instance, if **Forward** is defined, but **Backward** is not, the reverse of the **Forward** animation will be used for **Backward**. For all other missing directions, the direction is filled in by the nearest non-missing direction. For instance, if **Left** is not defined, but **Forward,Left** is, then **Forward,Left** will also be used for **Left**. When an avatar is matching an animation clip to their current relative movement direction, they will choose the clip closest to their direction.

While only one direction needs to be defined, it's recommended that at least the following be defined (all with the same set of tags) to provide acceptable animation in Results:

**Forward**

Use an animation where the occupant is moving directly forward.

**Forward,Up**

Use an animation where the occupant is moving up stairs.

**Forward,Down**

Use an animation where the occupant is moving down stairs.

Each move animation can be defined as a set of animation clips, each clip defining a different speed range for the move animation's direction. For instance, you might specify a walking animation clip for speeds ranging from 0 to 1.8 m/s and a running animation clip for speeds > 1.8 m/s. Each animation clip is represented as a single row of the **Clips** table as shown in [Figure 105](#), and its speed range spans from the previous clip's **Top Speed** to the current clip's **Top Speed**. Clips must be listed in order of increasing **Top Speed**. The **Top Speed** for the last clip can be left

blank. When an occupant chooses a move animation clip in Results, they will choose the one whose range spans the occupant's current speed.

The following additional properties can be defined for move animations and their animation clips:

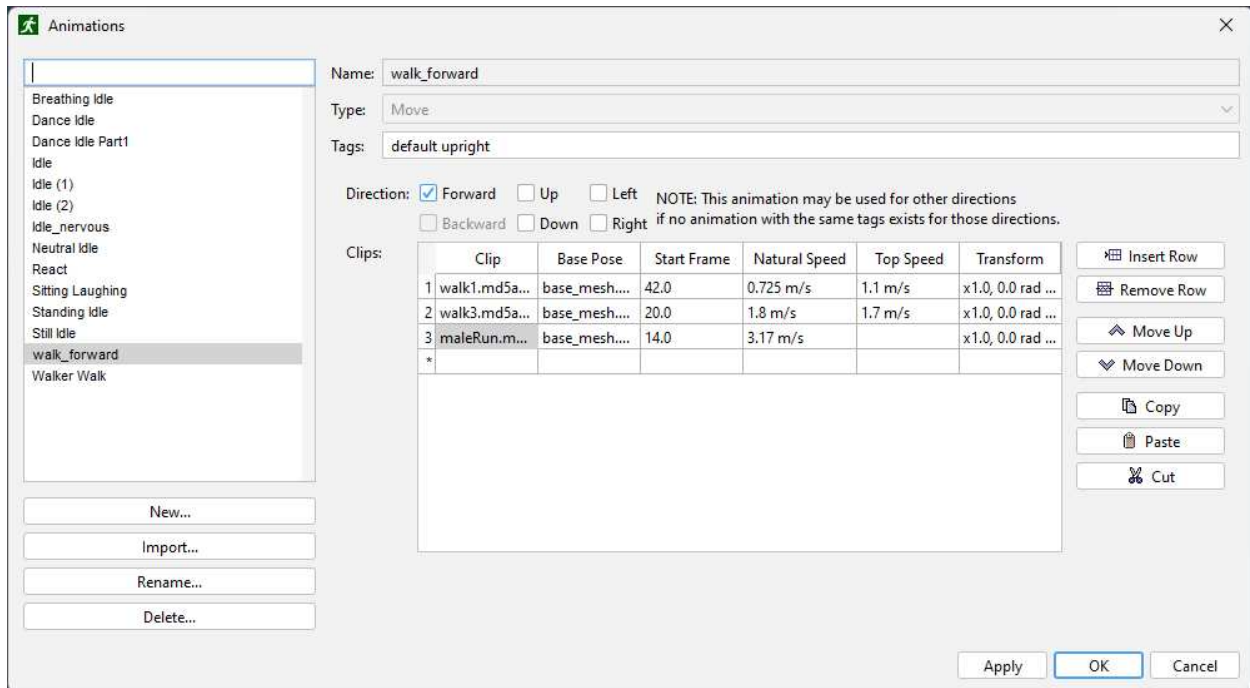


Figure 105. Move Animation Dialog

## Direction

Defines the direction of movement for the animation, relative to the direction the occupant is facing. Directions can be combined if desired. For instance, if the animation represents an occupant walking diagonally forward and to the right of the direction they're facing, select the directions, **Forward** and **Right**.

## Start Frame

Remaps the specified frame as the first frame of the animation. For walking/running animations, this should be the frame at which the avatar's legs are together and the left foot is about to step forward.

## Natural Speed

The approximate speed that an avatar would be moving if they played the animation at **1x** speed. For instance, the **Natural Speed** of a walking animation clip representing a single walk cycle might be calculated as  $2 * \text{foot\_stride} / \text{clip\_time}$ , where **foot\_stride** is the distance between the avatar's heels at full stride, and **clip\_time** is the total clip time. When playing the animation clip in Results for a particular occupant, the playback speed of the clip is adjusted

automatically according to the **Natural Speed** of the animation and the movement speed of the occupant. For instance, if the **Natural Speed** is 1.1 m/s, but the occupant is moving at 1.32 m/s, the animation will play at 1.2x speed ( $1.32/1.1$ ).

### Top Speed

Defines the upper speed range for the clip. An occupant will use this clip if their speed is between the **Top Speeds** of the previous and current clips.

## 4.6.4. Making Animations Available to Other Pathfinder Users

When animations are imported into Pathfinder, they are copied into `%APPDATA%/Pathfinder/models/anim`s, which limits their use to the current user on the current machine. If the results of a Pathfinder model using these custom animations are loaded by another user or on another machine, Results may fail to match the animation tags, causing it to display the avatars in their T- or A-poses instead. To make these custom animations available to others users, perform the following from the computer and user account from which the animations were imported:

1. In File Explorer, navigate to the following location: `%APPDATA%/Pathfinder/models/anim`s
2. Copy the desired animation files. This may involve copying the JSON file and files in `clips` and `meshes`.
3. On the machine on which the avatars are to be made available, in File Explorer, navigate to the following: `%PROGRAMDATA%/Pathfinder/models/anim`s. If this location does not yet exist, create it.
4. Paste from the clipboard.
5. Restart Pathfinder on the target computer.

#### NOTE

If you want to make all custom animations available to other users, simply copy the entire `%APPDATA%/Pathfinder/models/anim`s directory into `%PROGRAMDATA%/Pathfinder/models` on the target computer.

# Chapter 5. Occupants

In Pathfinder, occupants are defined in two parts: *Profiles* and *Behaviors*.

## Profile

Defines characteristics of the occupants, such as maximum speed, radius, avatar, and color. These profile properties can optionally be overridden per occupant or changed through the course of the simulation by their **Behavior**.

## Behavior

Defines a sequence of actions the occupant will take throughout the simulation, such as moving to a room, waiting, and then exiting.

## 5.1. Profiles

Pathfinder uses an *occupant profile* system to manage distributions of parameters across groups of occupants. This system helps you control the occupant speed, size, and visual distributions. To edit occupant profiles, you can use the **Edit Profiles** dialog ([Figure 106](#)).

To open the **Edit Profiles** dialog: on the **Model** menu, click **Edit Profiles**, or double-click the word **Profiles** in the Navigation View.

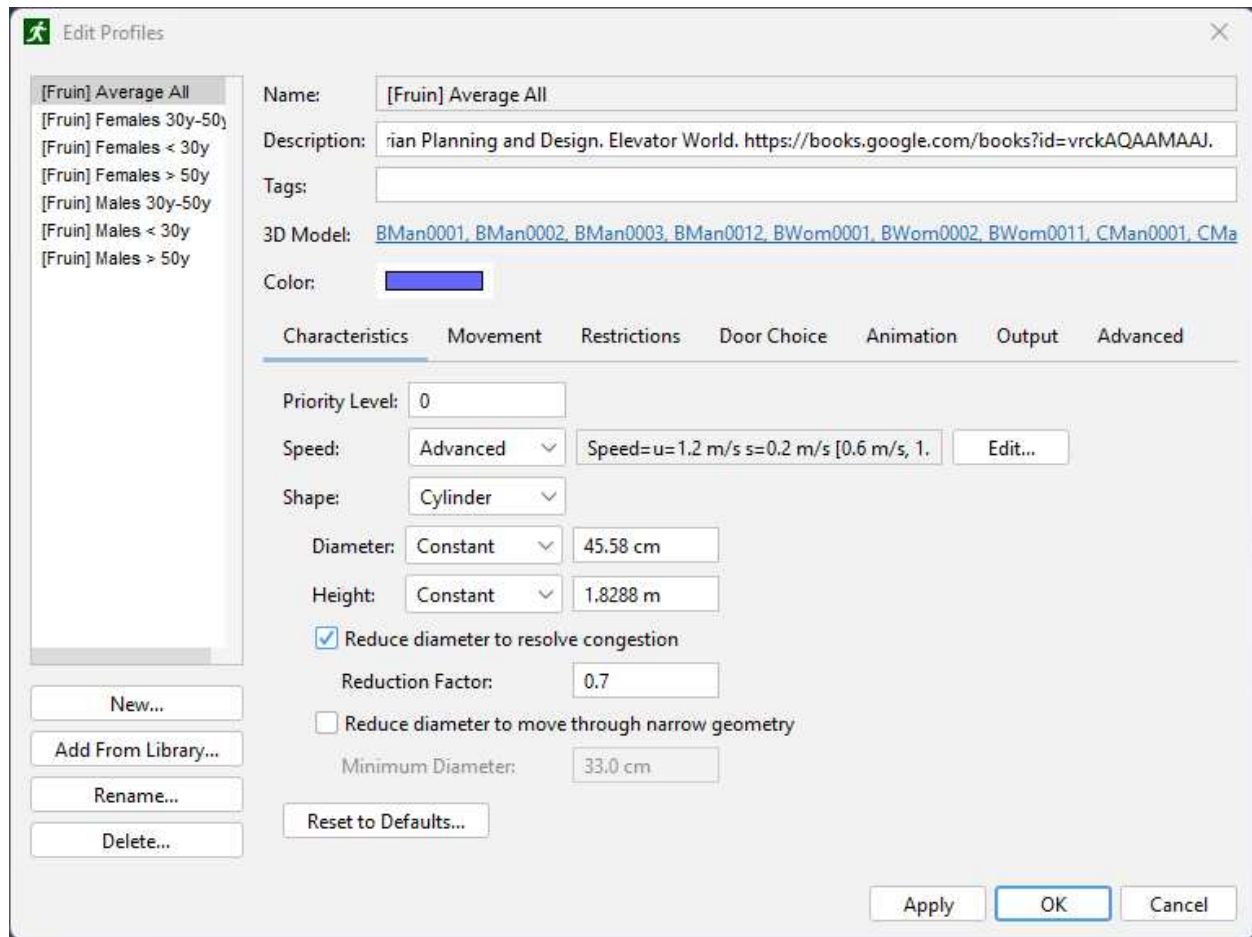


Figure 106. The Edit Profiles dialog

## Description

Provides a place to enter descriptive text. This value is not used outside the **Edit Profiles** dialog.

## Tags

Provides a list of tags that are applied to occupant at the beginning of the simulation or when they are first added to the simulation from an occupant source. Individual tags are separated by spaces (see [Section 5.8](#)).

## 3D Model

Provides a way to use a specific set of 3D human models for an occupant profile. When rendering occupants as 3D human models belonging to the current profile, Pathfinder will choose one of the 3D models selected in the **3D Model** dialog.

To select 3D models:

1. Click **Edit** on the **3D Model** row. This will open the **3D Models** dialog (see [Figure 107](#)).

- To enable or disable a particular model, click the model's icon. If the **<hidden>** choice is selected, no 3D Model will be shown for the occupant.

By clicking the **Import** button in the **3D Models** dialog, custom avatars can be imported into Pathfinder (see [Section 4.5](#)).

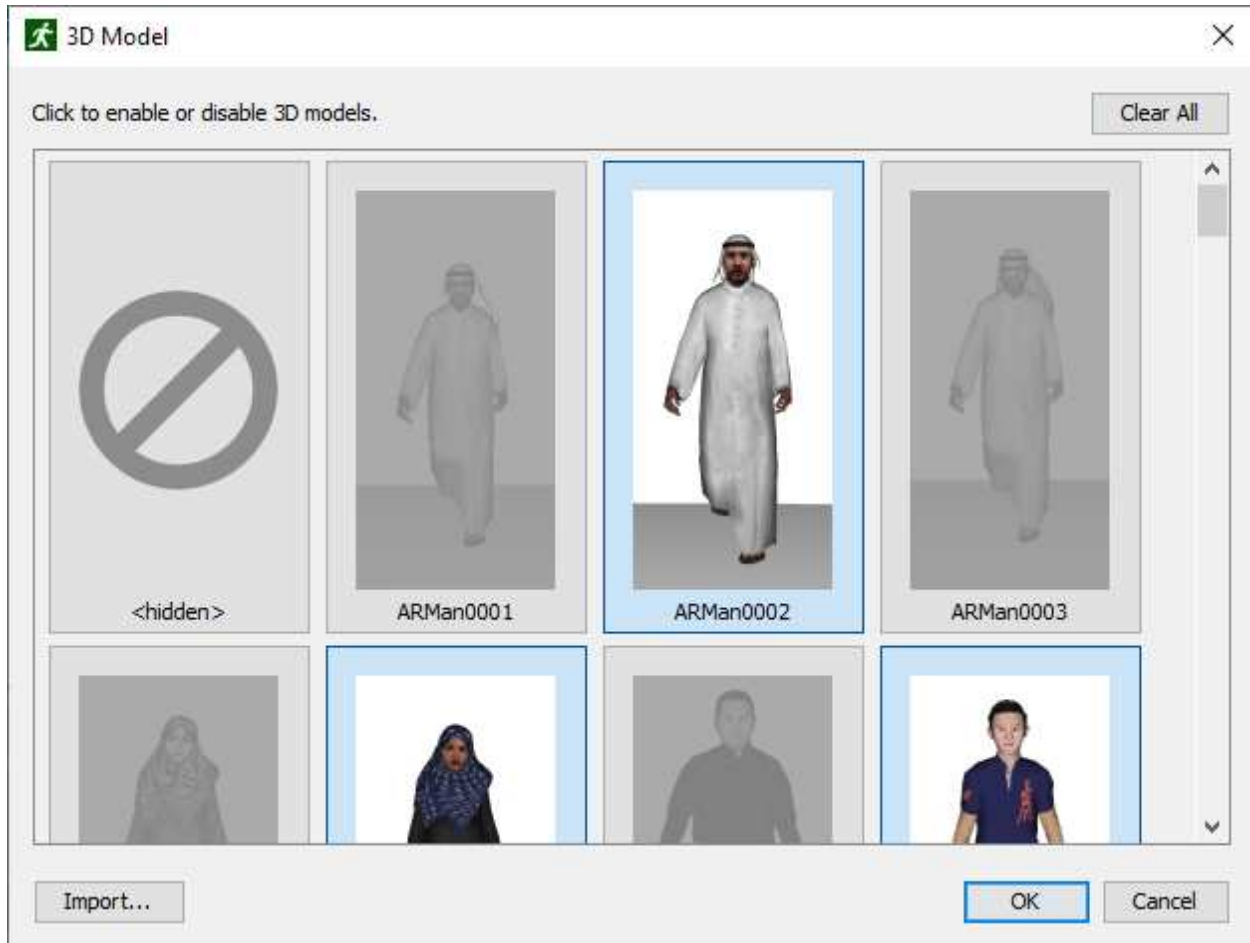


Figure 107. An example of the 3D model dialog.

### 5.1.1. Characteristics Tab

#### Priority Level

Completely relative and higher values indicate a higher priority. For instance, if three occupants meet with assigned priorities of 4, 6, and 12, they will behave the same as if their priorities were 0, 1, and 2, respectively. This allows occupants of lower priority to move out of the way of those of higher priority. This would be useful when simulating first responders that must be able to move easily through a crowd of occupants.

#### Shape

The occupant can be either a cylinder or a polygonal shape. In the case of a **Cylinder**,

**Diameter, Height, Reduction Factor** and **Minimum Diameter** can be specified.

### **Diameter**

The 'shoulder width' of the cylinder representing the occupant. It is used for collision testing and path planning during the simulation. This value will also affect how many occupants can be added to a room without overlapping. The default value of **45.58 cm** is based on the average of measurements of male and female persons from nine countries (Pheasant and Haslegrave 2005). An earlier edition is referenced by G. Keith Still in his Ph.D. thesis (Still 2000). As shown in the Pathfinder Validation and Verification Manual, this size (and Interpersonal Distance) results in movement that matches the SFPE and experimentally measured fundamental diagrams. Caution should be used in changing the default Diameter value.

### **Height**

specifies the height of the cylinder used for inter-occupant collisions. This is useful for limiting collisions that might occur between occupants on different floors when the floors have been modeled close together.

### **Reduction Factor**

[optional] Is enabled by default and is useful in helping occupants to "squeeze" by one another in congested areas. This factor is unitless and should be selected to be between **0** and **1**. It can be considered a way to get the benefits of elliptical occupant shape (i.e. it gives the occupants a radius corresponding to turning sideways) despite the simulator's exclusive use of circular occupant diameters. When occupants cannot otherwise move due to occupant-occupant collisions, this factor is multiplied against the shoulder width to shrink the collision circle and potentially allow occupants to ease through congestion.

### **Minimum Diameter**

[optional] Is designed for models that have narrow seating areas and aisles. In cases where there is some detail of the geometry that makes it difficult for full-width occupants to move, this option can be enabled to alter the occupant-geometry collision test to accommodate the narrower geometry. The occupant-occupant collision test is not affected by this feature. Occupants will reduce their diameter to this value only if the geometry would otherwise prevent them from further following their path. The default value of 33.00 cm is based on the maximum depth of a human body for 95<sup>th</sup> percentile of measured individuals (Pheasant and Haslegrave 2005).



**NOTE**

If **Reduce diameter to move through narrow geometry** is checked, the occupant will use the **Minimum Diameter** to plan their path through the model. This means that if there are any narrow gaps  $\geq$  **Minimum Diameter**, such as behind a chair or shelf that results in a shorter path, the occupant will try to navigate through that space. When using this feature, it is best to either manually close off small gaps like this or use the **Close Gaps** feature to ensure occupants don't use these areas.

**Polygon**

If this option is selected, a dropdown menu is used to select the vehicle shape. Vehicles are used to model an occupant using an alternate mode of transportation, such as a wheelchair or bed as used in assisted evacuation. See [Section 5.2](#) for more information on creating vehicle shapes or [Chapter 7](#) for more information on assisted evacuation. The **Edit** button can be used to access the **Vehicle Shapes** dialog. The optional **Reduction Factor** can be used in the same way as for the cylinder shape, because occupants with vehicle shapes sometimes temporarily use cylinder shapes in their movement calculations to avoid stuck situations.

**5.1.2. Movement Tab**

The **Movement** tab provides parameters related to how occupants use their surroundings.

**Initial Orientation**

Specifies an angle in degrees counter-clockwise from the positive x-axis which the occupants will use as their orientation in the beginning of the simulation. Occupants can also be made to orient on a specific point. This can be done outside of the **Edit Profiles** dialog by right clicking selected occupants, and selecting **Orient on Point** from the context menu. See [Section 9.3](#) for more information about how this action works with Occupant Targets.

**Requires Assistance to Move**

Specifies whether the occupant requires assistance from another occupant in order to move. This is useful when modeling assisted evacuation as described in [Chapter 7](#). This option is recommended for occupants that are unable to move under their own power (e.g. in a bed or other carrying device).

**Ignore One-way Door Restrictions**

Whether the occupant will ignore the direction specified for one-way doors.

**Escalator Preference**

Specifies how the occupant will stand or walk on escalators and moving walkways. The following options are available:

**Stand anywhere**

Occupants will get on the escalator and stand at any position. The occupant will stand still and travel on the stair at the stair's speed constant.

**Stand left**

Occupants will get on the escalator and stand at the left side. The occupant will stand still and travel on the stair at the stair's speed constant.

**Stand right**

Occupants will get on the escalator and stand at the right side. The occupant will stand still and travel on the stair at the stair's speed constant.

**Walk**

The escalator's speed constant will be added to the occupant's desired speed on the stair to determine the occupant's final speed, and they will attempt to walk around standing occupants.

**Trigger Susceptibility (Seeking)**

Defines the occupant's susceptibility to triggers while the occupant is seeking (actively trying to reach a destination). This is only used when a trigger's **Ignore Occupant Susceptibility** property is **unchecked**. In this case, the occupant's **susceptibility** is multiplied by the trigger's **influence** to determine the probability that the occupant will use the trigger. See [Section 10.3.4](#) for more information about how occupants choose to use triggers.

**Trigger Susceptibility (Waiting)**

Defines the occupant's susceptibility to triggers while the occupant is waiting.

**Allowed Triggers**

Defines which triggers the occupant is allowed to use. This is combined with each trigger's **Allowed Occupants** property to determine which occupants can use each trigger (see [Section 10.3.1](#)). The following values can be specified:

**All**

The occupant can use any trigger.

**None**

The occupant cannot use triggers.

**From List**

The occupant can only use the specified triggers (see [Section 2.6.1](#) for help using the Object Sets dialog). When using this option, the list can be specified in one of the following ways:

**Reject**

The occupant will not use any of the specified triggers, but can use those not listed.

**Accept**

The occupant can only use one of the specified triggers.

**5.1.3. Restrictions Tab**

The **Restrictions** tab specifies which components the occupant can use during path planning. For each component type, the following options are available:

**All**

The occupant can use all components of that type.

**None**

The occupant cannot use any components of that type.

**Up Only**

[escalators only] The occupant can only use escalators that move occupants up.

**Down Only**

[escalators only] The occupant can only use escalators that move occupants down.

**Based on Behavior**

[elevators only] The occupant will only use elevators when their behavior contains a **Go To Elevators** action. (see [Section 5.3.3](#)). Otherwise, the occupant can use all non-restricted elevators at any time.

**From List**

The occupant can only use certain components (see [Section 2.6.1](#) for help using the Object Sets dialog). When using this option, the list can be specified in one of the following ways:

**Reject**

The occupant will not use any of the specified components, but can use any other not listed.

**Accept**

The occupant can only use one of the specified components of that type.

**5.1.4. Door Choice Tab**

The **Door Choice** tab provides parameters related to how occupants choose doors to exit from in each room. For more information about door choice see the ([Pathfinder Technical Reference, n.d.](#)).

### Current Room Travel Time

The cost factor that affects the cost of travelling to a door in the occupant's current room, ignoring all other occupants. Higher values increase the door's cost in this category, making the Current Room Travel Time relatively more important.

### Current Room Queue Time

The cost factor that affects the cost of waiting in a queue at a door in the occupant's current room. Higher values increase the door's cost in this category, making the Current Room Queue Time relatively more important.

#### NOTE

The status of a door (open/closed) affects the current room queue time. See the Pathfinder Technical Reference ([Pathfinder Technical Reference, n.d.](#)) for more details.

### Global Travel Time

The cost factor that affects the cost of travelling from a door to an exit or the occupant's next goal, ignoring all other occupants. Higher values increase the door's cost in this category, making the Global Travel Time relatively more important. The following are factored into the global travel time:

- Walking distance and occupant speed
- Door wait times, including those for local doors (see [Section 3.5.3](#))
- Room speed modifiers (see [Section 3.2.5](#))
- Elevator travel time, based on its Timing Model (see [Section 3.11.5](#))

### Elevator Wait Time

The time during which the occupant will prefer to use an elevator. This is done by pretending that there is no queue in front of the elevator, and that the elevator is ready to pick up the occupant. After the elevator wait time runs out, the occupant will be more likely to choose the stair. The elevator wait time is reset every time the occupant starts a new behavior action.

### Current Door Preference

The value used to make occupants stick to their currently chosen doors, preventing excessive door switching. A value of 100% will cause occupants to never switch doors once an initial door is chosen, and a value of 0% will allow occupants to freely change their selected doors.

### Current Room Distance Penalty

The value used to exponentially increase the cost associated with travelling based on how far the occupant has travelled in the current room. This causes the occupant to prefer shorter routes over faster routes the further they have travelled in the current room. Every time the

occupant travels this distance in the current room, the travel time cost will have doubled. Setting this value to zero will disable this feature.

### 5.1.5. Animation Tab

The **Animation** tab provides parameters that control the animations used by occupants in the 3D Results when occupants are viewed as people. Pathfinder provides several basic animations, including walking and idling, but allows custom animations to be imported as well (see [Section 4.6](#)).

The animation used by an occupant depends on their current movement state. If they are idling (not actively moving toward a destination), they will use one of the **Idle Animations**. Otherwise, they will use one of the **Move Animations**.

An occupant's animation can be overridden by a vehicle if they are using a vehicle shape (see [Section 5.2](#)).

#### NOTE

While an occupant's animation can be set in the profile properties, in many cases it might make more sense to temporarily change it as part of their behavior using the [Change Profile Property](#) and [Reset Profile Property](#) behavior actions. For instance, you might want to show an occupant going to a vendor and then paying for a good. In this case, instead of setting their animation in the profile to a **pay** animation, which would play every time they were idle, it would make more sense to represent the payment as the following behavior actions:

1. Change <Idle Animations> to [pay]
2. Wait (15.0 s)
3. Reset <Idle Animations>

In order to define an occupant's animations, Pathfinder uses a tagging system. Each animation is associated with one or more tags (see [Section 4.6.3](#)). When specifying which animations to use in the occupant profile, either a pre-defined animation can be selected or a custom animation can be matched with any combination of the animation's tags. For instance, if a custom animation **AnimA** has the tags **upright** and **alert** and **AnimB** has the tags **upright** and **bored**, specifying an **Idle Animation** of **upright** will match with both **AnimA** and **AnimB**. However, specifying an **Idle Animation** of **upright bored** will only match **AnimB**. Enter fewer tags to match generically or more tags to match more specifically; however, be careful not to enter a combination of tags that cannot be matched by any animation.

If a matching animation cannot be found, the occupant will be displayed in the 3D Results in a T- or A-pose. This is a visual indication that the match was not made.

**NOTE**



**Figure 108. Avatar T-Pose**

To edit the animations, click the underlined text next to **Idle Animations** or **Move Animations**. The **Animation Tags** dialog will appear.

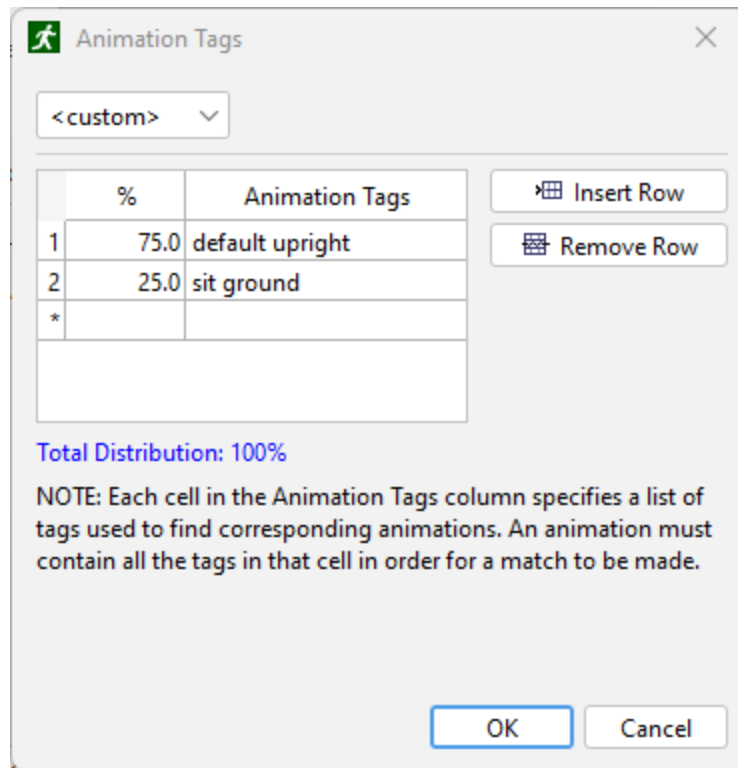


Figure 109. Animation Tags Dialog

The following options are available depending on the animation type:

### Idle Animations

#### <default>

Occupants idle in a standing position. Tags: **default,upright**

#### <custom>

Allows a distribution of animations to be specified, matched by tags. For example, in [Figure 109](#), approximately **75%** of occupants will use animations with the tags, **default** and **upright**. About **25%** of occupants will use animations with the tags, **sit** and **ground**. For any given row, if multiple animations match the specified tags, those animations will be distributed evenly. For instance, if two animations match **default,upright**, approximately **37.5%** of occupants will use one of the animations, and **37.5%** will use the other.

#### NOTE

Any of the pre-defined animations can be included in the distribution by entering the tags associated with the pre-defined animation. For example, **Sit (ground)** can be included by entering the tags, **sit,ground**.

### Sit (ground)

Occupants sit on the ground. Tags: **sit,ground**

**Lie (injured)**

Occupants lie on their side while injured. Tags: **lay,side,injured**

**Lie (back)**

Occupants lie on their back facing upward. Tags: **default,lay,supine**

**Move Animations****<default>**

Occupants walk normally. Tags: **default,upright**

**<custom>**

See **Idle Animations**.

**Lie (back)**

Occupants lie on their back facing upward. Tags: **default,lay,supine**

**NOTE**

This would not normally be used. It exists to support the (uncommon) case of an occupant lying in a bed that moves on its own without helpers.

**Wheelchair**

Occupants move as if they are operating a wheelchair. Tags: **default,wheelchair**

## 5.1.6. Output Tab

The **Output** tab provides the **Output Detailed Data** option. When checked, additional output data files are generated for each occupant using this profile. The files contain data for each time step, such as occupant speed, location, etc. (see [Section 16.9](#)).

**NOTE**

Enabling this feature may cause the simulation to use significantly more resources, including CPU and disk space. It may be better to only enable this feature for specific occupants (see [Section 5.1.10](#)).

## 5.1.7. Advanced Tab

The **Advanced** tab provides the following parameters:

**Acceleration Time**

Is a Steering Mode parameter that specifies the amount of time it takes for the occupant to reach maximum speed from rest or to reach rest from maximum speed. The resulting forward acceleration of each occupant is [stem eb9c27c75009d0aa1831f5987bce4867]. The occupant uses a separate reverse acceleration of [stem 50c6c1635babf023970caf011006f52a] and a



separate lateral acceleration of [stem fde79c59b4226eeeb764719c2866a97b].

### Persist Time

The amount of time an occupant will maintain an elevated priority when trying to resolve movement conflicts. See the ([Pathfinder Technical Reference, n.d.](#)) for more details.

### Collision Response Time

The distance ahead at which an occupant will start recording a cost for colliding with other occupants when steering. It is multiplied by an occupant's current speed to calculate the recording threshold. As an example, with the default value of **1.5 s** and the default maximum speed of **1.19 m/s**, the occupant will look ahead **1.785 m** from their current position to detect potential collisions and calculate costs.

### Slow Factor

Specifies a fraction of the occupant's speed at which they are considered to be slow. A slow occupant will consider backward directions to separate with others, while a fast moving occupant has a tighter, more focused direction.

### Wall Boundary Layer

Specifies the distance that occupants try to maintain with walls and other static obstructions.

### Personal Distance

The desired distance one occupant will try to maintain with others in a queue. This may be entered explicitly as a distance, from queue area, or from queue density. When entered as a distance, it is defined as the gap distance between the occupants' shapes rather than their center-to-center distance. If entered as an occupant area, the personal distance is calculated based on sphere packing: [stem 16ca23b213c657741726ba93e526ab4e] Where [stem 3e18a4a28fdee1744e5e3f79d13b9ff6] is personal distance, [stem 44bc9d542a92714cac84e01cbbb7fd61] is the occupant area, and [stem 2103f85b8b1477f430fc407cad462224] is the occupant's shoulder width. Personal distance is calculated similarly when entered as occupant density, except that [stem c2be5eb22f6ed300bc45b6e849e51b8c] where [stem 6dec54c48a0438a5fcde6053bdb9d712] is occupant density.

#### NOTE

When personal distance is entered in terms of occupant area or density, this does not guarantee that occupants will maintain this area or density, but they should be fairly close when occupants are queued. It is also not accurate for maintaining large distances between occupants (**> 1 m**, center-to-center). If a larger separation distance is desired, use **Social Distancing** instead.

## Social Distance Occupants

Enables or disables Pathfinder's social distancing model for the occupant profile and indicates which occupants should be used to apply **Social Distance**. This can be one of the following values:

### Accept All

Enables social distancing with all other occupants.

#### NOTE

By default, occupants in movement groups do not perform social distancing with other members of their group. To enable social distancing with other group members, check the **Enforce social distancing between group members** box in the **Movement Groups** property panel or the **Edit Movement Group Template** dialog (see [Section 8.1](#)).

### Reject All

Disables social distancing.

### From List

Allows social distancing with specific occupants. With this option, you can choose to either **Accept** or **Reject** occupants with the specified tags (see [Section 5.8](#)). For example, in the image below, the profile's occupants will enable social distancing with other occupants who have the both the tags, **suspicious** and **stranger**.

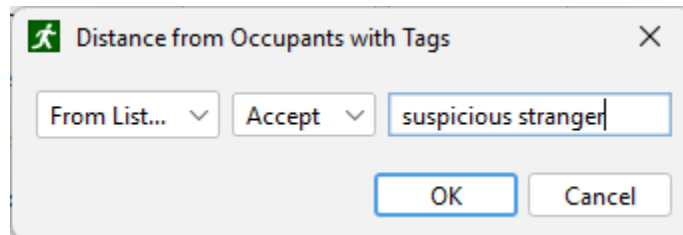


Figure 110. Social Distance Occupants Dialog

## Social Distance

If social distancing is enabled, specifies the desired distance between occupants. The value can be set as a constant or a distribution of values, enabling occupants to have different desired social distances.

#### NOTE

Social distance is defined differently than **Personal Distance** in that it is the center-to-center distance between occupants, whereas the personal distance is the gap distance between their shapes. In addition, social distance is enforced more strictly than personal distance and is more useful for maintaining larger distances between occupants ( $> 1$  m).

## Speed in Smoke

When FDS Output integration is enabled (see [Section 4.4](#)) and PLOT3D output for SOOT Visibility is present, this option controls how much smoke will limit the maximum velocity of occupants. Disabling this option will prevent visibility data from limiting maximum occupant velocity. By default, Pathfinder calculates maximum speed in smoke on a per-occupant basis using the following function of smoke-free maximum speed ([stem e5da456f085b8d5894d3a152a085d834] in m/s) and visibility ([stem 97b67e81dff22dc41a325344c07666b] in m) presented in Fridolf et al. ([Fridolf et al. 2019](#)) as the the individualized representation (method 3): [stem 66e14040fce2818391134f7c1bb70f05].

Each of these parameters (except on/off parameters) can be set using a **constant** value, a **uniform distribution** between two values, or either a **normal** (Gaussian) or **log-normal distribution** (see [Section 5.1.8](#)).

Each occupant in the Pathfinder model is linked to one profile. Profile parameters can be edited in the profiles dialog at any time and the occupants using that profile will be automatically updated. Occupants' profiles can be set when adding the occupants or by selecting the occupants after being created and editing the **Profile** box in the property panel.

### 5.1.7.1. Advanced Speed Properties

In most cases, users only need to enter the maximum speed of an occupant in the **Characteristics** tab of the **Edit Profiles** dialog. The actual speed of the occupant throughout a simulation will vary according to this speed and a set of assumptions from the *Engineering Guide to Human Behavior in Fire* ([SFPE 2019](#)), which takes into account the type of terrain being traversed (stairs, ramps, etc.) and the density of surrounding occupants (see ([Pathfinder Technical Reference, n.d.](#))).

Pathfinder allows for fine-grained control over the speed of the occupant, however, and the SFPE assumptions can be customized or replaced with other assumptions.

To do so, follow these steps:

1. In the **Model** menu, click **Edit Profiles** to open the **Edit Profiles** dialog.
2. Click the **Characteristics** tab.
3. Next to **Speed**, click the drop-down box and select **Advanced** as shown in [Figure 111](#).

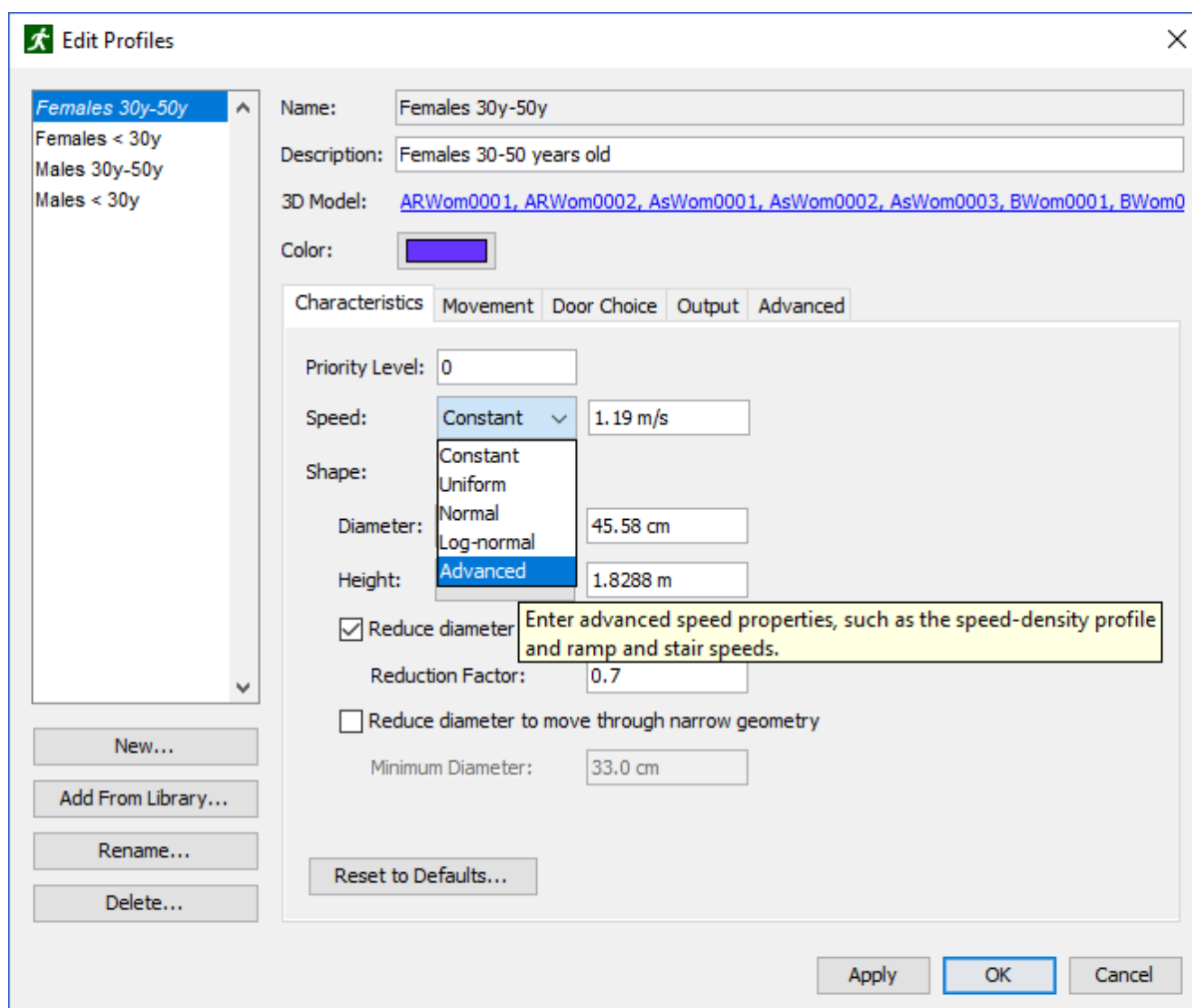


Figure 111. Editing advanced speed properties

This will open the **Advanced Speed Properties** dialog as shown in [Figure 112](#).

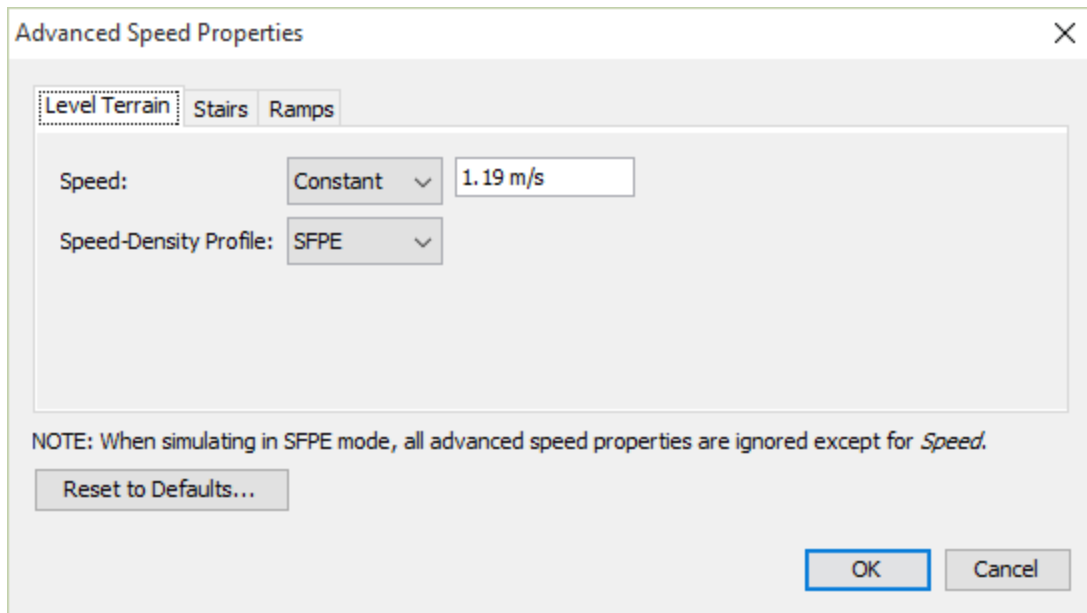


Figure 112. Advanced Speed Properties dialog

As noted in the **Advanced Speed Properties** dialog, when simulating in SFPE mode, only the maximum speed property is used. All others in the **Advanced Speed Properties** dialog are ignored.

Clicking **Reset to Defaults** in the **Advanced Speed Properties** dialog will reset all advanced speed properties to their default values in Pathfinder.

Each tab in the dialog allows the customization of occupant speed on each terrain type in Pathfinder, including [Level Terrain](#), [Stairs](#), and [Ramps](#).

#### 5.1.7.1.1. Level Terrain Tab

The **Speed Density Profile** can be used to set the occupant's speed as a function of the surrounding occupant density, also known as the fundamental diagram. One of three options may be chosen for the profile:

##### SFPE

The fundamental diagram as specified in the *Engineering Guide to Human Behavior in Fire* (SFPE, 2003) is used to adjust occupant speed.

##### Constant

The maximum speed is multiplied by a constant factor to determine the occupant's speed. In most cases, the occupant will either try to move this speed or stop.

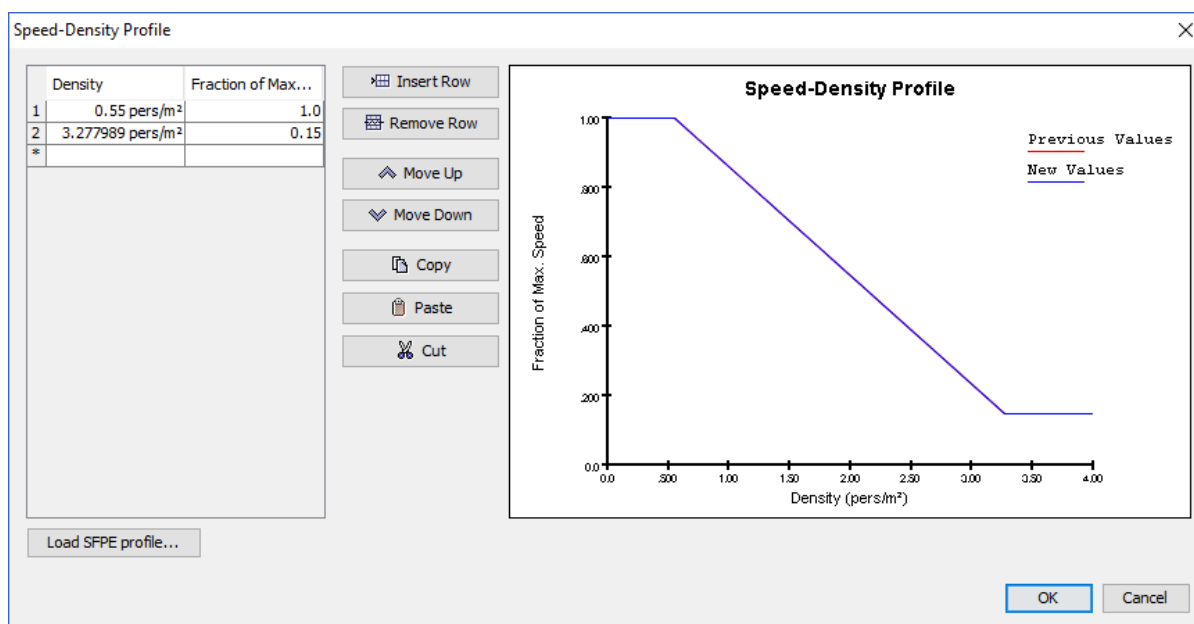
##### From Table

The speed is entered as a fraction of the occupant's maximum speed as a function of

surrounding occupant density. The values are entered in the **Speed-Density Profile dialog** as shown in [Figure 113](#). The profile is defined as a piece-wise linear function, where sample points are entered into a table. For densities within the entered range, the speed fraction is interpolated. For densities outside the range, the speed fraction is equal to that entered for the nearest specified density. A preview of the fundamental diagram is shown to the right of the table. The SFPE profile, which is the default profile, may be loaded by clicking **Load SFPE profile** below the table.

#### NOTE

The loaded SFPE profile has a minimum value of 15% of the occupant's maximum speed. This ensures that occupants will not become stuck at high densities.



**Figure 113. Speed-Density Profile Dialog**

#### 5.1.7.1.2. Stairs Tab

The following properties may be entered in the **Stairs** tab:

##### Speed Fraction Up

Defines a speed factor when the occupant travels up stairs.

##### Speed-Density Up

Defines the speed-density profile to use when the occupant travels up stairs.

##### Speed Fraction Down

This specifies the speed fraction when the occupant travels down stairs.

## Speed-Density Down

This specifies the speed-density profile to use when the occupant travels down stairs.

The above parameters define a factor that is multiplied by occupant's maximum speed to determine their speed up or down stairs.

The factor may be specified in one of three ways:

### SFPE

The assumptions from the SFPE *Engineering Guide to Human Behavior in Fire* are used to determine the speed fraction up stairs. This option uses the rise and run of the stair to determine the speed.

### Constant

A constant factor is applied to the maximum speed.

### From Table

The speed factor can be entered as a function of stair slope. Stair slope is defined as step rise/step run. The function is entered as a piece-wise linear function.

Additionally, the **Speed-Density Up** and **Speed-Density Down** parameters have the **From Level Terrain** option. This option forces the occupant to use the same speed-density profile as specified on the **Level Terrain** tab when the occupant travels up or down stairs.

#### 5.1.7.1.3. Ramps Tab

The **Ramps** tab has nearly identical properties available as on the **Stairs** tab. The only difference is that **Speed Fraction Up** and **Speed Fraction Down** are entered in terms of the ramp's geometric slope rather than step slope. The geometric slope is determined per triangle in the resulting navigation mesh and depends on the triangle's normal direction.

## 5.1.8. Stochastic Parameters

Many parameters, including speed, can be specified by a constant value or in terms of a probability distribution.

Pathfinder supplies inputs for the following distributions:

**Table 9. Distribution Options**

Distribution	Description
Constant	Specifies a constant value.

Distribution	Description
Uniform	Generates random values that are uniformly distributed between the specified minimum and maximum parameters.
Normal	Generates random values that are normally distributed from the specified mean and standard deviation, as shown in the following equation: [stem 229309b0d2ea9ffb194750e226e00822] Where [stem 07617f9d8fe48b4a7b3f523d6730eef0] and [stem 8cda31ed38c6d59d14ebefa440099572] are the mean and standard deviation, respectively, and $x$ is a random number from a standard normal distribution. Generated values are bounded by the minimum and maximum parameters.
Lognormal	Generates random values that are lognormally distributed from the specified location and scale parameters, as shown in the following equation: [stem 2ac109f297c7de380238592e2c9ac640] Where [stem 07617f9d8fe48b4a7b3f523d6730eef0] and [stem 8cda31ed38c6d59d14ebefa440099572] are the location and scale parameters, respectively, and $x$ is a random number from a standard normal distribution. Generated values are bounded by the minimum and maximum parameters.

### 5.1.9. Seeds

Each occupant has a unique random seed that determines the specific values generated from a profile distribution. Each of these occupant-specific values can be seen by selecting an individual occupant. These specific values will never change unless the distribution is changed in the profile or a new seed is manually generated for the occupant. This ensures that two simulation runs with the same input model will give the same answer. New seeds can be generated for occupants by right-clicking the occupants and selecting **Randomize**.

For an example of the effects of the changing the seed or profile, consider the following scenario:

1. A profile has been created using a uniform distribution of speed on the range 1 m/s to 2 m/s.
2. An occupant is created using this profile.
3. Using the occupant's unique random seed, Pathfinder assigns the occupant a speed of 1.6 m/s based on the occupant's profile.
4. The simulation is run several times, and each time the occupant has a maximum speed of 1.6 m/s.
5. The occupant's profile is changed so that its speed range is .5 m/s to 1 m/s.
6. Pathfinder assigns the occupant a new maximum speed of .6 m/s, which is used for all



subsequent simulations.

7. The user randomizes the occupant, and Pathfinder assigns a new maximum speed of **.91 m/s** to the occupant.

### 5.1.10. Customizing Occupants

When occupants are selected, their property panel appears as shown in [Figure 114](#). Occupants can be given custom profile data once they are added to the model. This can be done by selecting a set of occupants and checking the box next to the parameter to be customized.

Percentage-based distributions can also be used to assign occupant profiles and behaviors to a group of occupants, see [Section 5.5](#).

Customized properties will be overwritten if an occupant switches their profile or changes a property during simulation, see [Section 5.3.3.12](#) and [Section 5.3.3.13](#).

#### NOTE

In versions of Pathfinder prior to 2012.1, individual parameters could not be customized. If one parameter was to be customized, all had to be customized.

When using custom profile data, only constant values can be used for occupant parameters. In addition, once a parameter is customized, any changes to that parameter in the profile will not affect the customized value.

Figure 114. Using a custom occupant profile

Occupants with individually customized parameters can easily be found by right-clicking all or a sub-set of occupants and selecting **Select Customized Occupants** from the right-click menu if any exist in the selection.

### 5.1.11. Profile Libraries

Profiles can be saved and reused using profile libraries ([Figure 115](#)). Profile libraries are managed in the **Profile Libraries** dialog. To open the **Profile Libraries** dialog, on the **Edit Profiles** dialog, click **Add From Library**. Alternatively, the **Profile Libraries** dialog can also be opened by clicking **Edit Profile Libraries** on the **Model** menu.

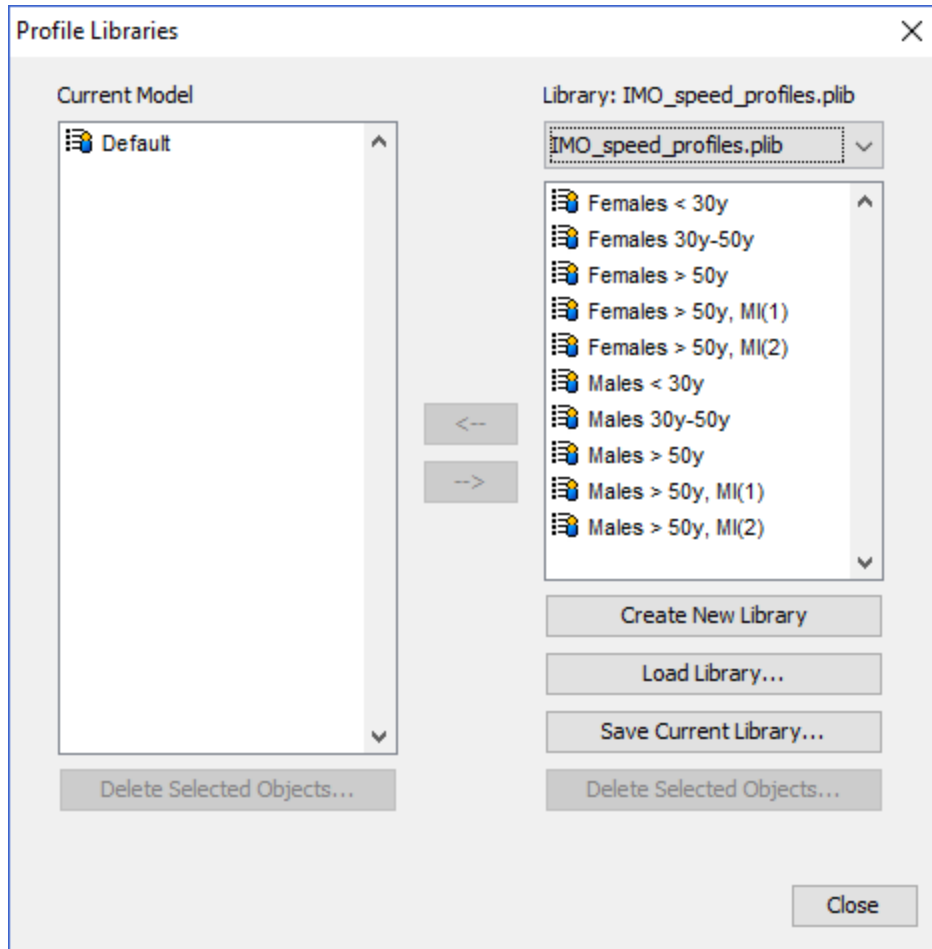


Figure 115. Profile Libraries Dialog

The list on the left side of the dialog shows all profiles contained in the current Pathfinder model. The list on the right side of the dialog shows all profiles stored in the profile library. Profiles can be moved between the lists using the buttons in the middle of the dialog. **Create New Library** clears the list of library profiles so that a new library can be created. **Load Library** opens a profile library from file. Pathfinder supports loading profiles from profile library files (**PLIB**) and from standard Pathfinder model files (**PTH**).

To create a Pathfinder library file (**PLIB**) based on profiles in the current model:

1. Click **Create New Library** to create a new empty library.
2. Use the arrow buttons to copy profiles from the current model to the library.
3. Click **Save Current Library**

The dropdown menu on the right side of the dialog can be used to load predefined libraries. To fill the predefined libraries box, Pathfinder scans two locations: a folder in the Pathfinder installation shared by all users and a folder in the current user's **APPDATA** path, see [Table 10](#).

**Table 10. Pathfinder Library Locations**

PLIB File Location	Suggested Use
C:\Program Files\PyroSim 20XX\lib\profiles (or alternate install folder)	Built-in libraries included with Pathfinder. Administrator access is required to modify this location. Changes to PLIB files from this location will affect all users sharing the computer.
%APPDATA%\Pathfinder\profiles	Recommended location for storing PLIB files. Non-admin users have write access to this folder. Changes to PLIB files in this location will affect only the current user.

All predefined profile libraries are opened as read-only to prevent accidental modification. To modify a library, you will need to resave to overwrite the existing PLIB file. See [Appendix A](#), for more information on the occupant profiles provided with Pathfinder.

## 5.2. Vehicle Shapes

An occupant can be assigned a vehicle to use during the simulation. When using a vehicle, an occupant uses the shape of the vehicle rather than the default cylinder shape when performing collision detection. Any convex polygon can be used as the vehicle shape.

Vehicle shapes are not intended to be used as shapes of regular occupants as vehicle movement is fundamentally different from occupant movement. Unlike regular occupants, vehicles are not allowed to move sideways. Consequently, the vehicles follow a different path, which accounts for this movement restriction. Vehicles also avoid collisions with other occupants and walls differently from regular occupants. Finally, vehicle movement uses more complex calculation, which can increase the run time of the simulation.

The vehicles can be created and customized in the **Edit Vehicle Shapes** dialog ([Figure 116](#)). To open the **Edit Vehicle Shapes** dialog: on the **Model** menu, click **Edit Vehicle Shapes**.

**Edit Vehicle Shapes**

Default Bed  
Default Wheelchair

Name: Default Wheelchair

Description:

Height: 1.0 m

3d Model: Wheelchair

Occupant Animation: Wheelchair

Occupant Offset: X: 0.0 m Y: 0.0 m Z: 0.0 m

Points:

	X	Y
1	-0.66 m	-0.38 m
2	0.66 m	-0.38 m
3	0.66 m	0.38 m
4	-0.66 m	0.38 m
*		

Insert Row  
Remove Row  
Move Up  
Move Down

Pivot:

X	Y
0.0 m	0.0 m

Positions of attached occupants:

X	Y
-0.71 m	0.0 m
*	

Remove Row

New...  
Rename...  
Delete...

Shape Area: 1.0032 m²

Corresponding occupant count: 6

Apply OK Cancel

Figure 116. The Edit Vehicle Shapes Dialog

The **Description** box provides a place to enter descriptive text. This value is not used outside the **Edit Vehicle Shapes** dialog.

### Height

Specifies the height of the vehicle shape.

### 3D Model

Specifies the avatar to display for the vehicle when viewing occupants as people. This 3D model is shown in addition to the 3D model specified in the occupant's profile. If **<shape>** is chosen as the 3D model, an extruded polygon is used as the 3D model for the vehicle. The color of the extruded polygon will match the color of the occupant. Custom avatars can also be imported as described in [Section 4.5](#).

### Occupant Animation

The animation that should be used for the occupant using the vehicle. This overrides the animation specified in the occupant's profile and can be one of the following values:

**Default**

Uses the occupant's profile animation as if they are not using a vehicle.

**Wheelchair**

Makes the occupant look like they're operating a wheelchair.

**Bed**

Makes the occupant appear to be lying flat on their back.

**<custom>**

Allows animation tags to be entered manually, enabling imported animations to be used (see [Section 5.1.5](#) for more information about animation tags).

**Occupant Offset**

Specifies how far the occupant's 3D model should be moved to line up with the vehicle's 3D model. A value of  $(0, 0, 0)$  will put the occupant's 3D model at the vehicle's origin as shown in the vehicle preview.

**Points**

Specify the shape of the vehicle as projected onto the ground. The shape must be specified as a convex polygon. All changes in this list are displayed in the vehicle preview to the right.

**Pivot**

Specifies the position of the pivot, which is a point around which the vehicle shape rotates. The pivot can lie inside or outside of the vehicle shape.

**Positions of attached occupants**

Specify the locations where an assistant can attach in order to move the vehicle. This is only needed for assisted evacuation and is ignored if the occupant using the vehicle never requests assistance (see [Chapter 7](#)). All changes in this list are displayed in the vehicle preview to the right.

**Shape Area**

Shows the area of the vehicle shape. This area is used for occupant density calculations.

**Corresponding Occupant Count**

Shows how many regular occupants fit inside the vehicle shape. This amount is used to determine whether occupants will fit in rooms that they are about to enter. For example, if an elevator has a nominal load of 10 people, then a bed, which corresponds to 12 people, will not be able to fit inside. Note that the corresponding occupant count is not used for occupant density calculation in SFPE mode.

The **Vehicle Preview** shows a graphical preview of the projected shape of the vehicle and provides additional capability to add, remove, and change points of the vehicle shape, the pivot location, and the positions of attached agents.

The dashed lines depict the horizontal and vertical axes. Their intersection (by default point  $x=0$ ,  $y=0$ ) is the pivot of the vehicle, around which the vehicle body rotates. The red arrow points in the direction of the movement of the vehicle. The pivot can be moved both inside and outside of the vehicle shape.

A point can be added to the vehicle shape or positions of attached occupants by right clicking inside the graphical editor. When moving the cursor close to a point, the point changes its color to orange indicating that it can be selected. A selected point changes its color to yellow. This selection is also displayed in the point editors on the left side of the dialog.

Any point can be moved by selecting and dragging, and it can be deleted by right clicking and selecting the **Delete Point** option. Mouse wheel can be used to zoom in and out, and mouse dragging can be used to move around. **Reset View** button resets the view so that all points are visible. The **Undo/Redo** buttons can be used to undo/redo any action within the 2D Graphical Editor.

A warning will appear at the bottom of the dialog in case constraints like minimum number of points in the vehicle shape, convexity, distance of assisting occupants from the vehicle shape etc. are violated.

For more information about the navigation of polygonal shapes in Pathfinder see the Pathfinder Technical Reference ([Pathfinder Technical Reference](#), n.d.).

## 5.3. Behaviors

Behaviors in Pathfinder represent a sequence of *actions* the occupant will take throughout the simulation. Once they occupant has completed all actions, they are removed from the simulation. Actions may be added that can make the occupant wait or travel to a destination, such as a room, point, or exit.

The last destination for the occupant can be thought of as the occupant's *sink*. By default, there is one behavior in the model called "Goto Any Exit". This behavior simply makes the occupant move from their starting position to any exit present in the model by the fastest route.

As with profiles, any number of occupants can refer to a single behavior. Any changes to the behavior will be reflected in referring occupants.

### 5.3.1. Creating a New Behavior

To create a new behavior:

1. Right click the **Behaviors** node from the Navigation View, and from the right-click menu, click **Add a Behavior**, which will open the New Behavior dialog shown in [Figure 117](#).
2. In the New Behavior dialog, enter a behavior name, and optionally specify an existing behavior to base the new behavior on. Using this option will copy all the actions from the existing behavior.

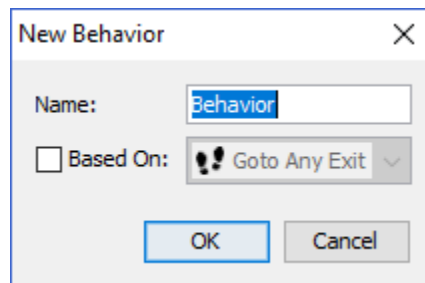


Figure 117. New Behavior dialog

With the new behavior selected, the behavior property panel will appear as shown in [Figure 118](#).



Figure 118. Behavior Property Panel

#### Initial Delay

Specifies an initial delay that makes the occupant wait at their starting position before moving to the next action. If this link is clicked, it will show a dialog where different distribution curves can be entered for the delay, similar to those discussed in Profiles.

### 5.3.2. Adding Actions

Additional actions can be added to any behavior, such as going to a room, a waypoint, an elevator, or simply waiting in place. To add an action, select a behavior or existing behavior action. The property panel ([Figure 119](#)) will show a drop-down button with the description of an action that can be added. To add the currently shown action, simply click the button. To add a different action, click the down-arrow shown to the right of the button and select the desired action from the *behavior actions list*.

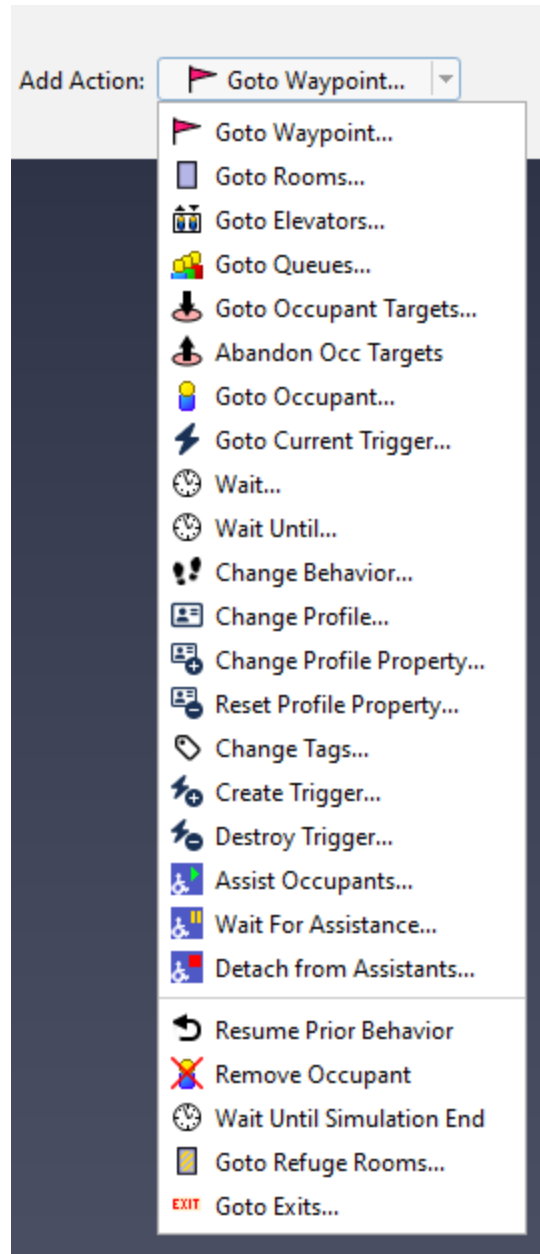


Figure 119. Behavior actions list

Once the desired action is clicked, a creation panel will be shown above the 3D/2D View depending on the action. Enter the desired parameters in the creation panel as discussed in the following sections, and then click **Create** to create the action and append it to the behavior. If the behavior itself was selected when adding the action, then the new action will be appended to the end of the list. If, instead, an action was selected when the new action was created, then the new action is inserted directly after that selected action. If the action is an ending action (includes all actions below the separator in the *behavior actions list*), the action will always be added at the end. If the behavior already has an ending action, it will replace the existing ending action.



Actions always occur in the order shown in the Navigation View. For instance, as shown in [Figure 120](#), an occupant using "Behavior1" would first go to any elevator, then go to "Room00", then wait for 20 seconds, then go to "Room09". After reaching "Room09", the occupant will be removed from the simulation. The actions can be reordered at any time by dragging and dropping an action in the list in the Navigation View.



Figure 120. Example of action order for a behavior

#### NOTE

All actions must end with an ending action. These can be identified in the *behavior actions list* as those below the separator.

### 5.3.3. Behavior Action Types

Table 11. Behavior Actions Summary

Action	Ending Action ?	Description
<a href="#">Goto Waypoint</a>	No	Specifies that an occupant should go toward a specific point on the navigation mesh.
<a href="#">Goto Rooms</a>	No	Specifies that an occupant must select a room out of a set, and go to it.
<a href="#">Goto Elevators</a>	No	Tells an occupant to use elevators.
<a href="#">Goto Queue</a>	No	Specifies that an occupant should join a designated Queue.
<a href="#">Goto Occupant Targets</a>	No	Tells an occupant to reserve an Occupant Target out of a set and travel to the target.
<a href="#">Abandon Occ Targets</a>	No	Tells an occupant to abandon Occupant Targets they have previously reserved.
<a href="#">Goto Occupant</a>	No	Tells an occupant to go to another occupant with a specified tag.
<a href="#">Goto Current Trigger</a>	No	Tells an occupant to go to the location of their currently used Trigger.
<a href="#">Wait</a>	No	Tells an occupant to wait in their current location for a specified amount of time.

Action	Ending Action ?	Description
Wait Until	No	Instructs an occupant to delay their movement until a specified simulation time passes.
Change Behavior	No	Instructs an occupant to change a behavior to a new behavior picked randomly from a behavior distribution.
Change Profile	No	Instructs an occupant to change to a different profile picked randomly from a profile distribution.
Change Profile Property	No	Instructs an occupant to change one of their properties to a specified value.
Reset Profile Property	No	Instructs an occupant to reset a property back to their profile value.
Change Tags	No	Instructs an occupant to change their Tags.
Look At	No	Instructs an occupant to orient their body towards another occupant.
Look Ahead	No	Instructs an occupant to orient their body toward their goal.
Create Trigger	No	Creates triggers that may influence other occupants.
Destroy Trigger	No	Destroys triggers created by an occupant.
Assist Occupants	No	Instructs the occupant to join an assisted evacuation team and begin assisting occupants who request assistance from that team.
Wait For Assistance	No	Indicates that the occupant should wait for assistance from other occupants.
Detach from Assistants	No	Detaches a client from their assistants, allowing the assistants to continue helping other clients.
Resume Prior Behavior	Yes	Instructs the occupant to continue using the behavior prior to the current behavior.
Remove Occupant	Yes	The occupant will be removed from the simulation.
Wait Until Simulation End	Yes	The occupant will wait in their current location until the end of the simulation.
Goto Refuge Rooms	Yes	Instructs the occupant to go to one of a set of rooms marked as <i>refuge areas</i> .
Goto Exits	Yes	Instructs an occupant to take the fastest route to a set of exits.

### 5.3.3.1. The Goto Waypoint Action

A *Goto Waypoint* action specifies that an occupant should go toward a specific point on the navigation mesh. Once they arrive within a certain radius of the point, they will move on to the next action in their behavior.

To add one of these actions:

1. Click the **Goto Waypoint** button from the *behavior actions list*.
2. Enter the **Location** or click a point on the model.
3. Enter the **Arrival Radius** or drag the mouse after you click the location to specify the radius.

These parameters can be entered manually in the creation panel or be filled in by clicking a point on the navigation mesh in the 3D or 2D View or click-dragging to specify the location+arrival radius. When clicking or click-dragging, the action is created when the mouse button is released.

### 5.3.3.2. The Goto Rooms Action

A *Goto Rooms* action specifies that an occupant must select a room out of a set, and go to it. Once they cross a door into the room, they are considered to be in the room and can move on to the next action in their behavior. If multiple rooms are specified for the action, the occupant will go to the one that is fastest to reach.

To add a *Goto Rooms* action:

1. Click the **Goto Rooms** button from the *behavior actions list*.
2. Click the **Rooms** link to specify the rooms with a dialog (see [Section 2.6.1](#) for help using the Object Sets dialog), or left-click the desired rooms in the 3D or 2D View.
3. Right-click in the 3D/2D View to finish selecting the rooms and create the action, or click **Create**.

### 5.3.3.3. The Goto Elevators Action

A *Goto Elevators* action tells an occupant to use elevators. When using this action, an occupant will go to a specified elevator, call it, wait for it to arrive, enter it, and then wait for it to reach their target floor. Once they reach their target floor, they can begin their next action. If multiple elevators are specified for the action, the occupant will use the one that allows them to reach their target floor fastest.

To add a *Goto Elevators* action:

1. Click the **Goto Elevators** button from the *behavior actions list*.
2. Either click the **Elevators** link to specify the desired elevators with a dialog (see [Section 2.6.1](#)

for help using the Object Sets dialog), or left-click the desired elevators in the 3D or 2D View.

3. Select a **Target Floor** for the occupant to target for discharging. The floors available will reflect floors reached by the elevator selection, and will default to the discharge floor property of the elevator traveled in if no selection is made.
4. Right-click in the 3D/2D View to finish selecting the elevators and create the action, or click **Create**.

#### 5.3.3.4. The Goto Queue Action

A *Goto Queue* action specifies that an occupant must select a Queue out of a set, and go to it. When using this action, the selected Queue will direct the occupant's movement through available queue paths and service points and then releasing the occupant to the next behavior action once it is complete.

To add a *Goto Queue* action:

1. Click the **Goto Queues** item from the *behavior actions list*.
2. Click the **Queue** link to select the queues with a dialog (see [Section 2.6.1](#) for help using the Object Sets dialog). Both queues and queue groups can be selected. If a queue group is selected, the queues in that group do not need to be selected. Alternatively, left-click the desired target queues in the 3D or 2D View.
3. Click the **Create** button

#### 5.3.3.5. The Goto Occupant Targets Action

A *Goto Occupant Targets* action tells an occupant to reserve a single Occupant Target out of a set of targets and then navigate toward the reserved target (see [Chapter 9](#)). Once the occupant has reserved a target, they will hold the reservation indefinitely as they move toward it and then perform other behavior actions. They will even maintain the reservation after they have exited the model or have otherwise been removed from the simulation. Holding a reservation to a target prevents other occupants from using that target and allows the occupant to return to it if they have been distracted by a Trigger.

The occupant will only give up the reservation if they encounter an **Abandon Occ Targets** action in their behavior list that indicates they should do so. This action can either be part of their assigned behavior or can be part of a Trigger behavior (see [Chapter 10](#)).

The occupant reaches their chosen target and is finished with the **Goto Occupant Targets** action when their body shape intersections a circular area surrounding the target location with a **0.5 m** radius.

To add a *Goto Occupant Targets* action:

1. Click the **Goto Occupant Targets** item from the *behavior actions list*.
2. Click the **Occ Targets** link to specify the targets with a dialog (see [Section 2.6.1](#) for help using the Object Sets dialog), or left-click the desired targets in the 3D or 2D View.
3. Enter the desired preferences as described below.
4. Right-click in the 3D/2D View to finish selecting the targets and create the action, or click **Create**.

An occupant uses their *Preference* parameters to decide the order in which they attempt to reserve targets.

### Priority Preference

Defines how the occupant will account for a target's priority when choosing a target. This preference is always considered before the **Distance Preference**, and may be one of the following values:

#### None

The occupant does not consider target priority, and will only rely on the **Distance Preference** to choose a target.

#### Lower

The occupant prefers available targets with the lowest priority value.

#### Higher

The occupant prefers available targets with the highest priority value.

### Distance Preference

Defines how the occupant will take their travel distance to each target into account when choosing a target. This preference is secondary to the **Priority Preference**. The following preferences are available:

#### None

The occupant does not care about their distance to the target. This effectively allows an occupant to choose an available target at random.

#### Nearest

The occupant prefers available targets nearest to them, according to travel distance.

#### NOTE

In order to have an occupant choose an available target at random, set both the **Priority Preference** and **Distance Preference** to **None**.

For more information on the Occupant Target reservation system, how occupants choose targets, and how conflicting requests are resolved see [Section 9.6](#).

#### 5.3.3.6. The Abandon Occupant Targets Action

The *Abandon Occupant Targets* action cancels one or more of an occupant's previous Occupant Target reservations. Abandoning a target allows other occupants to reserve the target using a **Goto Occupant Targets** action.

To add an *Abandon Occ Targets* action:

1. Click the **Abandon Occ Targets** item from the *behavior actions list*.
2. Select which reservations should be abandoned by choosing an option for **Targets**. This can be one of the following values:

##### **All**

The occupant will abandon all previously reserved targets.

##### **Most Recent**

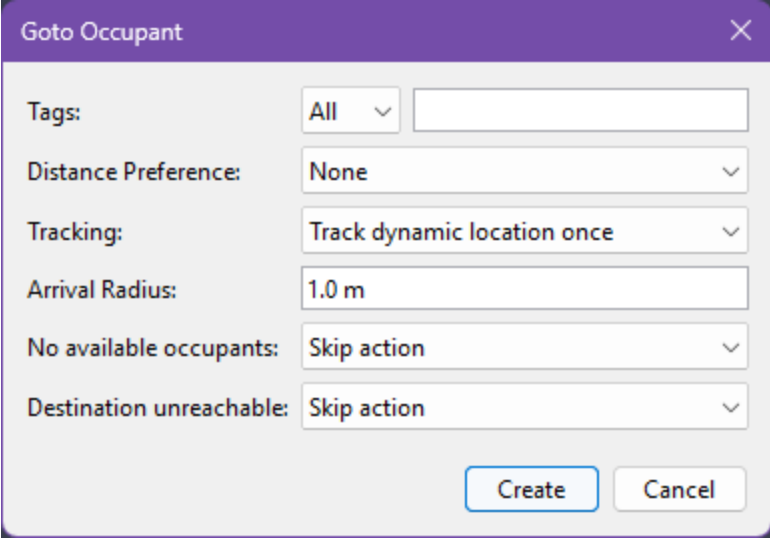
The occupant will only abandon the most recently reserved target.

3. Click **Create**.

#### 5.3.3.7. The Goto Occupant Action

A *Goto Occupant Action* tells an occupant to select another occupant and seek to them.

1. Click the **Goto Occupant** item from the *behavior actions list*.
2. Enter the desired parameters in the **Goto Occupant** dialog as described below and shown in [Figure 121](#).
3. Click **Create**.



The 'Goto Occupant' dialog box has a purple title bar with a close button. It contains several configuration options:

- Tags:** A dropdown menu set to 'All' and an adjacent empty text input field.
- Distance Preference:** A dropdown menu set to 'None'.
- Tracking:** A dropdown menu set to 'Track dynamic location once'.
- Arrival Radius:** A text input field containing '1.0 m'.
- No available occupants:** A dropdown menu set to 'Skip action'.
- Destination unreachable:** A dropdown menu set to 'Skip action'.

At the bottom right, there are two buttons: 'Create' (highlighted with a blue border) and 'Cancel'.

Figure 121. Goto Occupant Dialog

A source occupant uses the following parameters to determine which destination occupant to select and how to seek to them:

### Tags

The set of tags to require the destination occupant to have (see [Section 5.8](#)).

#### All

Requires the destination occupant to have all tags in the set.

#### Any

Requires the destination occupant to have at least one tag in the set.

### Distance Preference

Defines how the occupant will take their travel distance to each occupant into account when choosing a destination occupant.

#### None

The occupant does not care about their distance to the occupant. The destination occupant is chosen randomly from those available.

#### Nearest

The nearest available occupant is chosen.

### Tracking

Defines how the source occupant will track the destination occupant.

**Go to occupant**

The source occupant seeks the destination occupant until they are within the **Arrival Radius** and have a clear line-of-sight.

**Follow occupant**

The source occupant seeks and follows the destination occupant as long as the destination occupant is still in the simulation and has the desired tags.

**Go to initial location**

The source occupant records the location of the destination occupant when the action is started and then seeks that point until they are within the **Arrival Radius** and have a clear line-of-sight to the point. The destination occupant may not still be at the point when the source occupant arrives.

**Arrival Radius**

The radius within which the source occupant is considered to have arrived at the destination occupant.

**No available occupants**

Defines what to do if there are no reachable occupants matching the **Tag** criteria when the action is started.

**Skip action**

The source occupant will skip this action.

**Wait**

The source occupant will wait until a destination occupant can be found.

**NOTE**

The source occupant will wait a small amount of simulation time before searching again, as each search may be an expensive operation. The default delay is set to **.25 s** but can be changed using the **Goto Occupant Search Interval** in the [Miscellaneous Simulation Parameters](#).

**Destination unreachable**

Defines how the occupant will behave if the destination occupant becomes unreachable.

**Skip action**

The source occupant will skip this action.

**Wait**

The source occupant will wait until the destination occupant becomes reachable again.



### Restart action

The action is restarted so a new destination occupant can be selected.

#### 5.3.3.8. The Goto Current Trigger Action

A *Goto Current Trigger* action tells an occupant to seek to their currently used [Trigger](#). If the occupant is not using a trigger, this action will be skipped.

To add a *Goto Current Trigger* action:

1. Click the **Goto Current Trigger** item from the *behavior actions list*.
2. Enter the desired parameters in the **Goto Current Trigger** dialog as described below.
3. Click **Create**.

The following properties are available for the **Goto Current Trigger** action:

#### Tracking

Defines how the occupant will seek to the trigger.

##### Go to trigger

The occupant seeks their current trigger until they are within the **Arrival Radius** and there is a clear line-of-sight to the trigger. If the trigger is destroyed before the occupant reaches it, the occupant will skip this behavior action.

##### Follow trigger

The occupant seeks and follows the trigger until the trigger is destroyed.

##### Go to initial location

The occupant records the location of their current trigger when the action is started, and then seeks that point until they are within the **Arrival Radius** and there is a clear line-of-sight to the point. The trigger might not still be at the point when the occupant arrives.

#### NOTE

The occupant will continue to seek the location even if the trigger is destroyed.

#### Arrival Radius

The radius within which the occupant is considered to have arrived at the trigger.

#### Destination unreachable

Defines how the occupant will behave if the location specified by **Tracking** is unreachable.

**Skip action**

The occupant will skip this action.

**Wait**

The occupant will wait for the location to become reachable again.

**5.3.3.9. The Wait Action**

A *Wait* action tells an occupant to wait in their current location for a certain amount of time. Once that time has elapsed they will begin their next action.

The manner in which they wait varies depending on the destination of their most recent seek action and the **Wait Mode**, which can be the following values:

**Avoid Others**

The waiting occupant will move out of the way of occupants headed toward a destination unless the destination overlaps with the waiting occupant's most recent destination.

**Wait in place with collisions**

The waiting occupant will stand still. Others will attempt to go around them, if possible.

**Wait in place without collisions**

The waiting occupant will stand still. Others make no attempt to avoid them and can pass through them. This may be helpful in situations such as tight seating rows in a theater or stadium, where the waiting occupant is actually sitting in a seat and should not be causing flow restrictions in the row.

The following provides more detail about how an occupant will wait for the various types of previous seek actions and **Wait Modes**:

**Table 12. Waiting Behavior**

Prevoius Seek Action	Wait Mode: <i>Avoid Others</i>	Wait Mode: <i>Wait in place with collisions</i>	Wait Mode: <i>Wait in place without collisions</i>
<a href="#">Goto Waypoint</a>	The occupant will try to stay close to the center of the waypoint while avoiding others.	The occupant will move to the center of the waypoint without avoiding others, where they will stay once reached. Others will still avoid them.	The occupant will move to the center of the waypoint without avoiding others, where they will stay once reached. Others can pass through them.

Prevoius Seek Action	Wait Mode: <i>Avoid Others</i>	Wait Mode: <i>Wait in place with collisions</i>	Wait Mode: <i>Wait in place without collisions</i>
<a href="#">Goto Rooms</a>	The occupant will try to move toward the farthest point in the room away from all active doors, allowing other occupants to enter the room.	The occupant will move slightly into the room and then wait without moving. Others will still avoid them.	The occupant will move slightly into the room and then wait without moving. Others can pass through them.
<a href="#">Goto Elevators</a>	The occupant will stay in the elevator they traveled in once it reaches the occupant's target floor.	Should not be used.	Should not be used.
<a href="#">Goto Queue</a>	The occupant waits in the room of the used service in the same manner as <b>Goto Rooms</b> .	The occupant stands still at their used service while others try to avoid them. Others can still use this service, however, which may cause excessive congestion.	The occupant stands still at their used service, while others continue to use the service and can pass through them.
<a href="#">Goto Occupant Targets</a>	The occupant uses the Occupant Target reservation system as described in <a href="#">Section 9.6</a> to ensure they have a reserved target. If they have reached the target, they will wait at it in the same manner as the <b>Goto Waypoint</b> action. If they haven't reached it, they will travel to the target and then wait.	The occupant uses the Occupant Target reservation system, travels to their reserved target, and then stands still at the target. Others try to avoid them.	The occupant uses the Occupant Target reservation system, travels to their reserved target, and then stands still at the target. Others can pass through them.

Prevoius Seek Action	Wait Mode: <i>Avoid Others</i>	Wait Mode: <i>Wait in place with collisions</i>	Wait Mode: <i>Wait in place without collisions</i>
<a href="#">Assist Occupants</a>	The occupant waits in the room where they detached from the assisted occupant in the same manner as <b>Goto Rooms</b> .	The occupant waits in the room where they detached from the assisted occupant and then stands still. Others must avoid them.	The occupant waits in the room where they detached from the assisted occupant and then stands still. Others can pass through them.

**NOTE**

If the occupant is being assisted, cannot move on their own, and their prior action left them in a target room, the assistants will first move the client away from all doors in that room, even if the doors are inactive, and then the assistants will disconnect from the client (see [Chapter 7](#)).

To add a *Wait* action:

1. Click **Wait** from the *behavior actions list*.
2. Specify the **Wait Time**.

#### 5.3.3.10. The Wait Until Action

A *Wait Until* action instructs an occupant to delay their movement until a specified simulation time passes. This action is useful for synchronizing movement of many occupants at a time. Occupants fill space during a *Wait Until* action per the same rules described for the *Wait* action. *Wait Until* actions can be specified using three different approaches: occupants can wait until a specific time, the next in a list of times, or the next time in a periodic function.

To add a *Wait Until* action:

1. Click **Wait Until** from the *behavior actions list*.
2. Click the time to edit the value, which will show the **Wait Until Time** dialog as shown in [Figure 122](#).
3. Choose a sub-action from the drop-down.
4. Specify the required parameters.
5. Click OK on the **Wait Until Time** dialog.
6. Click **Create**.

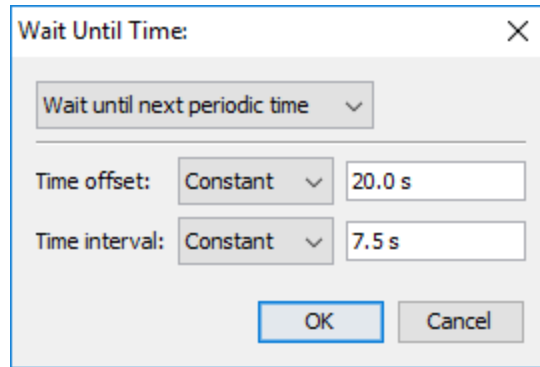


Figure 122. Wait Until Time dialog

Within the Wait Until Time dialog, there are three sub-actions available:

### Wait until a specific time

This action tells occupants to wait in their current location until the simulation time passes the specified time. If the time has already passed when the occupant begins this action, they will immediately proceed to their next action.

### Wait until next scheduled time

This action allows the user to specify a list of wait times. When an occupant begins the wait action, they compare the simulation time to the list of specified times, and wait until the first time greater than the simulation time. If the simulation time is greater than any specified time, the occupant immediately proceeds to the next action.

### Wait until next periodic time

This action sets up a periodic list of times.

#### Time offset

Specifies the first entry in the time list.

#### Time interval

Specifies the delay between times in the list.

The list of times is then generated from these parameters. For example, if **Time offset** is set to 60 s, and the **Time interval** is set to 10 s, the list of times will be [60, 70, 80, 90, 100, ...]. Similarly to the *Wait until next scheduled time* action, when the occupant begins this action, they will pick the next time in the list and wait until that time passes.

Both the **Time offset** and **Time interval** may be specified as distributions. In this case, each occupant will be given a unique list of times to choose from. For instance, if the **Time offset** is set to a uniform distribution of [50,60] and the **Time interval** is set to be constant at 10 s, one occupant may get the list, [52.3, 62.3, 72.3, 82.3, ...], and another occupant may get the list

[59.2, 69.2, 79.2, 89.2, ...].

#### 5.3.3.11. The Change Behavior Action

This action causes an occupant to switch its behavior. After this action, the occupant will start performing actions from a different behavior. This new behavior is randomly selected from a given behavior distribution. Besides other behaviors, the same behavior and *No Change* can also be selected. If the occupant changes behavior to its current behavior, the behavior will be restarted. If the occupant selects to make *No Change*, the **Change Behavior** action will have no effect. It is possible to create loops in the behavior references. That way the occupant might keep performing the actions forever (until the end of the simulation). Note that depending on the specific setup, it is possible that actions that come after a **Change Behavior** action will not be performed by the occupant.

See [Section 2.6.2](#) for help specifying behavior distributions.

#### 5.3.3.12. The Change Profile Action

This action causes an occupant to change their profile, which will override all parameters settable in an occupant profile ([Section 5.1](#)). This can be used to change an occupant's speed, shape, avatar, etc., throughout a simulation. The chosen profile is selected from a list of possible profiles, including a **No Change** item that will prevent the occupant's profile from changing.

##### NOTE

This action will override all occupant parameters as defined in the chosen profile, including those that have been customized for a specific occupant (see [Section 5.1.10](#)).

For profile parameters that are specified as distributions (e.g. max speed), a given occupant will always select a value from the same part of the distribution for that parameter. For instance, if a given occupant starts with a profile whose max speed is a normal distribution, and the occupant's speed is chosen from the 25<sup>th</sup> percentile, any subsequent profile changes will also lead to the occupant's speed being chosen from the 25<sup>th</sup> percentile of the new speed distribution. This ensures that if an occupant's profile is changed, that parameter will remain consistent for the occupant. In addition, if the parameter does not change with a profile change, the occupant will maintain the previous value. For value sets, such as occupant avatars, it is guaranteed that the occupant will maintain the same value only if the new profile uses the exact same set of values as the previous.

Profile changes can also be used to change an occupant's shape. It can change from a normal cylinder shape to a vehicle shape. It can also change from one vehicle shape, such as a bed, to another shape, such as a wheelchair. Because vehicles can be used with assisted evacuation (see [Chapter 7](#)), take care when switching between shapes during assistance. If an occupant is to change vehicles during assistance, for correct functionality it is best to detach from assistants before changing shapes using a **Detach from Assistants** action and then re-attach after changing

shapes using a **Wait for Assistance** action.

See [Section 2.6.2](#) for help specifying profile distributions.

### 5.3.3.13. The Change Profile Property Action

This action causes an occupant to override a single parameter settable in an occupant profile ([Section 5.1](#)). This can be used to change an occupant's speed, shape, avatar, etc., throughout a simulation.

To add a *Change Profile Property* action:

1. Click **Change Profile Property** from the *behavior actions list*.
2. Choose a property to change from the **Property** drop-down in the **Change Profile Property** dialog.
3. Specify the value of the property to change to. Many properties can be specified as either a distribution or a constant value.
4. Click **Create**.

For properties that are specified as distributions, a given occupant will always select a value from the same part of the distribution for that property. For instance, if a given occupant starts with a profile whose max speed is a normal distribution, and the occupant's speed is chosen from the 25<sup>th</sup> percentile, any subsequent speed property changes will also lead to the occupant's speed being chosen from the 25<sup>th</sup> percentile of the new speed distribution. For value sets, such as occupant avatars, it is guaranteed that the occupant will maintain the same value only if the new property has the exact same set of values as the previous.

Property changes can also be used to change an occupant's shape. It can change from a normal cylinder shape to a vehicle shape. It can also change from one vehicle shape, such as a bed, to another shape, such as a wheelchair. Because vehicles can be used with assisted evacuation (see [Chapter 7](#)), take care when switching between shapes during assistance. If an occupant is to change vehicles during assistance, for correct functionality it is best to detach from assistants before changing shapes using a **Detach from Assistants** action and then re-attach after changing shapes using a **Wait for Assistance** action.

#### NOTE

In the simulator, it takes exactly 1 time step to process a single Change Profile Property action. If you string together many of these actions in a row, this may introduce a delay in the occupant's movement. If you want to change many properties, it may be more effective to use the [Change Profile](#) action instead.

#### 5.3.3.14. The Reset Profile Property Action

This action causes an occupant to reset a parameter to the value defined in their occupant profile ([Section 5.1](#)). This can be used to reset changes previously performed by the [Change Profile Property](#) action.

To add a *Reset Profile Property* action:

1. Click **Reset Profile Property** from the *behavior actions list*.
2. Choose a property to reset from the **Property** drop-down in the **Reset Profile Property** dialog.
3. Click **Create**.

#### 5.3.3.15. The Change Tags Action

This action causes an occupant to change their tags, optionally replacing the occupant's current tags. See [Section 5.8](#) for other ways tags can be applied to occupants.

To add a *Change Tags* action:

1. Click **Change Tags** from the *behavior actions list*.
2. Specify how to change the tags by choosing an option for **Operation**. This can be one of the following values:

**Add (+)**

The specified tags will be added to the occupant.

**Remove (-)**

The specified tags will be removed from the occupant.

**Set**

The occupant's current set of tags will be replaced with the specified set.

3. Specify the set of **Tags** to perform this operation with. Each tag should be separated by a space.
4. Click the **Create** button.

#### 5.3.3.16. The Look At Action

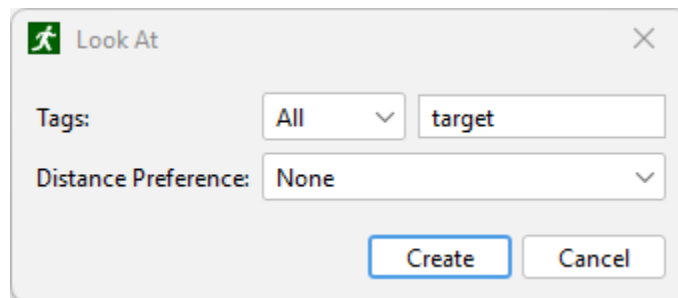
This action causes an occupant to orient their body toward another occupant and continue doing so while performing their next behavior actions until they encounter a [Look Ahead](#) action. While the occupant has a look-at target, they will orient their body as long as they have a clear line-of-sight to the destination occupant's center, according to the navigation mesh. If they do not have



clear line-of-sight, they will orient their body toward their goal instead.

**NOTE** Occupants using a vehicle shape will skip this action.

1. Click the **Look At** item from the *behavior actions list*.
2. Enter the desired parameters in the **Look-at** dialog as described below and shown in [Figure 123](#).
3. Click **Create**.



**Figure 123. Look-at Dialog**

A source occupant uses the following parameters to determine which destination occupant to select:

### Tags

The set of tags to require the destination occupant to have (see [Section 5.8](#)).

#### All

Requires the destination occupant to have all tags in the set.

#### Any

Requires the destination occupant to have at least one tag in the set.

### Distance Preference

Defines how the occupant will take their travel distance to each occupant into account when choosing a destination occupant.

#### None

The occupant does not care about their distance to the destination occupant. The destination occupant is chosen randomly from those visible.

#### Nearest

The nearest visible occupant is chosen.

#### 5.3.3.17. The Look Ahead Action

This action causes an occupant to orient their body toward their goal. This can be used after a [Look At](#) action in order to reset their orientation.

#### 5.3.3.18. The Create Trigger Action

This action creates new [Triggers](#) based on [Trigger Templates](#). This can be used to influence the behavior of other occupants in the simulation. It can also be used to create moving triggers.

To add a *Create Trigger* action:

1. Click **Create Trigger** from the *behavior actions list*.
2. Click the **Triggers** link to specify the trigger templates that should be created with a dialog (see [Section 2.6.1](#) for help using the Object Sets dialog).
3. Specify where the triggers should be created by choosing an option for **Location**. This can be one of the following values:

##### **Follow Occupant**

The triggers will be created at the occupant's current location and move with the occupant from there.

##### **Current Location**

The triggers will be created at the occupant's current location and remain there.

##### **Fixed Location**

The triggers will be created at a specified location. When this option is selected, you can specify the location by clicking a point on the model.

4. Click the **Create** button.

Triggers created by this action do not influence the occupant that created them. They will be automatically destroyed when that occupant is removed from the simulation.

#### 5.3.3.19. The Destroy Trigger Action

This action destroys triggers previously created by [Create Trigger](#).

To add a *Destroy Trigger* action:

1. Click **Destroy Trigger** from the *behavior actions list*.
2. Specify the trigger templates which should be destroyed in the displayed **Triggers** dialog (see [Section 2.6.1](#) for help using the Object Sets dialog).

3. Click the **Create** button in the dialog.

All triggers with the selected templates previously created by the occupant will be destroyed by this action.

#### 5.3.3.20. The Assist Occupants Action

This action is used in assisted evacuation models as discussed in [Chapter 7](#). This action instructs the occupant to join an assisted evacuation team and begin assisting occupants who request assistance from that team. When an occupant joins a team, they become an *assistant*. Once all occupants requesting assistance have completed their actions for which they wanted assistance, the assistant will begin their next action.

To add an *Assist Occupants* action:

1. Click **Assist Occupants** from the *behavior actions list*.
2. Specify the assisted evacuation **Team** that the occupant will join.
3. Click **Create**.

#### 5.3.3.21. The Wait For Assistance Action

This action indicates that the occupant should wait for assistance from other occupants as discussed in [Chapter 7](#). This action requires that the occupants using it have a vehicle shape with at least one attached occupant location as discussed in [Section 5.2](#). When an occupant begins this action, they are considered to be a *client*, and assistants will approach the client. The client waits according to the action's **Wait Mode** as described in [Section 5.3.3.10](#). Once all attachment positions are filled by assistants, the client begins their next action. The assistant will stay attached to the client until either the client has completed all subsequent actions or the client begins a **Detach from Assistants** action as described below.

To add a *Wait for Assistance* action:

1. Click **Wait for Assistance** from the *behavior actions list*.
2. Specify the **Assisting Teams** that will assist the client (see [Section 2.6.1](#)).
3. Click **Create**.

#### 5.3.3.22. The Detach From Assistants Action

This action indicates that the occupant should detach from their assistant as discussed in [Chapter 7](#). When an occupant begins this action, they are considered to be a *client*, and currently being assisted in their movement. Once detached, the occupant can proceed to their next behavior without assistance, only if they do not require assistance to move. This action also allows the

assistant to move on to the next occupant requiring assistance.

To add a *Detach From Assistance* action:

1. Click **Detach from Assistant** from the *behavior actions list*.
2. Click **Create**.

#### 5.3.3.23. The Resume Prior Behavior Action

This action resumes the Behavior prior to the current behavior. It can only be used for behaviors that are the target of a **Change Behavior** action. For instance, suppose **Behavior A** has the following actions:

1. Goto RoomA
2. Change Behavior <Behavior B>
3. Exit <any>

And **Behavior B** has the following actions:

1. Wait 20 s
2. Resume prior

The **Resume Prior Behavior** action will cause the occupant to follow this sequence of actions:

1. Goto RoomA (from **Behavior A**)
2. Wait 20 s (from **Behavior B**)
3. Exit <any> (from **Behavior A**)

#### NOTE

If a behavior ends with a **Resume Prior Behavior** action, the behavior cannot be directly referenced by an occupant or occupant source. It can only be referenced by a **Change Behavior** action.

To add a *Resume Prior Behavior* action:

1. Click **Resume Prior Behavior** from the *behavior actions list*.

#### 5.3.3.24. The Remove Occupant Action

Removes the occupant from the simulation. The occupant will no longer be visible or interact with any other remaining occupants.

To add a *Remove Occupant* action:

1. Click **Remove Occupant** from the *behavior actions list*.

#### 5.3.3.25. The Wait Until Simulation End Action

The occupant will wait in their current location until all other occupants finish their behavior actions or also start a **Wait Until Simulation End** action. The manner in which the occupant waits is similar to the [Wait](#) and [Wait Until](#) actions and depends on the action's **Wait Mode**.

##### NOTE

As with the **Wait** action, if the occupant is being assisted, cannot move on their own, and will be left in a room, the assistants will first move the client away from all doors, even if the doors are inactive, and then the clients will disconnect, allowing assistants to help other clients. The client will then wait without assistance for the remainder of the simulation (see [Chapter 7](#)).

To add a *Wait Until Simulation End* action:

1. Click **Wait Until Simulation End** from the *behavior actions list*.

#### 5.3.3.26. The Goto Refuge Rooms Action

This action instructs the occupant to go to one of a set of rooms marked as *refuge areas* as discussed in [Section 3.2.5](#). Like the **Goto Exits** action, this action must be the last action in the behavior. When an occupant reaches the refuge area, they will remain in the simulation and wait for the simulation to complete. Their behavior while waiting is described in the **Wait** action above. Additionally, the occupant will be tagged with "refuge\_reached", as reported in the *Occupant Summary* and *Occupant History* output files as discussed in [Section 16.8](#) and [Section 16.9](#).

This action is created similarly to the [Goto Rooms](#) action, except that only rooms marked as *refuge areas* may be chosen.

##### NOTE

If an occupant using this action is being assisted by others, the assistants will first move the client to a parking location and then detach. It is only at this point, that the client is marked with "refuge\_reached" and considered finished with the action.

#### 5.3.3.27. The Goto Exits Action

This action causes an agent to take the fastest route to a set of exits. Like the **Goto Refuge Rooms** action, this action must be last in the behavior. Once an agent goes through an exit, they are removed from the simulation and reported as having exited the model in the *Occupant Summary* and *Occupant History* output files as discussed in [Section 16.8](#) and [Section 16.9](#).


To add a *Goto Exits* action:

1. Click **Goto Exits** from the *behavior actions list*.
2. Either click the **Exits** link to specify the desired exits with a dialog (see [Section 2.6.1](#) for help using the Object Sets dialog), or left-click the desired exits in the 3D or 2D View.
3. Right-click in the 3D/2D View to finish selecting the exits and create the action, or click **Create**.

## 5.4. Generating Occupants

Occupants can be added to the simulation in a number of ways. The simulation can either be pre-seeded with any number of occupants or it can have occupants continuously generated by occupant sources. When pre-seeding the model, occupants can be placed individually in the 3D or 2D view, distributed in a rectangular region of a particular room, or distributed throughout the entire area of a room.

### 5.4.1. Individual Placement

Individual occupants can be added to the model with the Add Occupant tool . Occupants can only be placed in pre-existing rooms, stairs, and ramps and cannot overlap other occupants or room boundaries. Left-click a desired position with the mouse, or enter an x-y-z coordinate and press the **Create** button from the property panel to place an occupant ([Figure 124](#)).

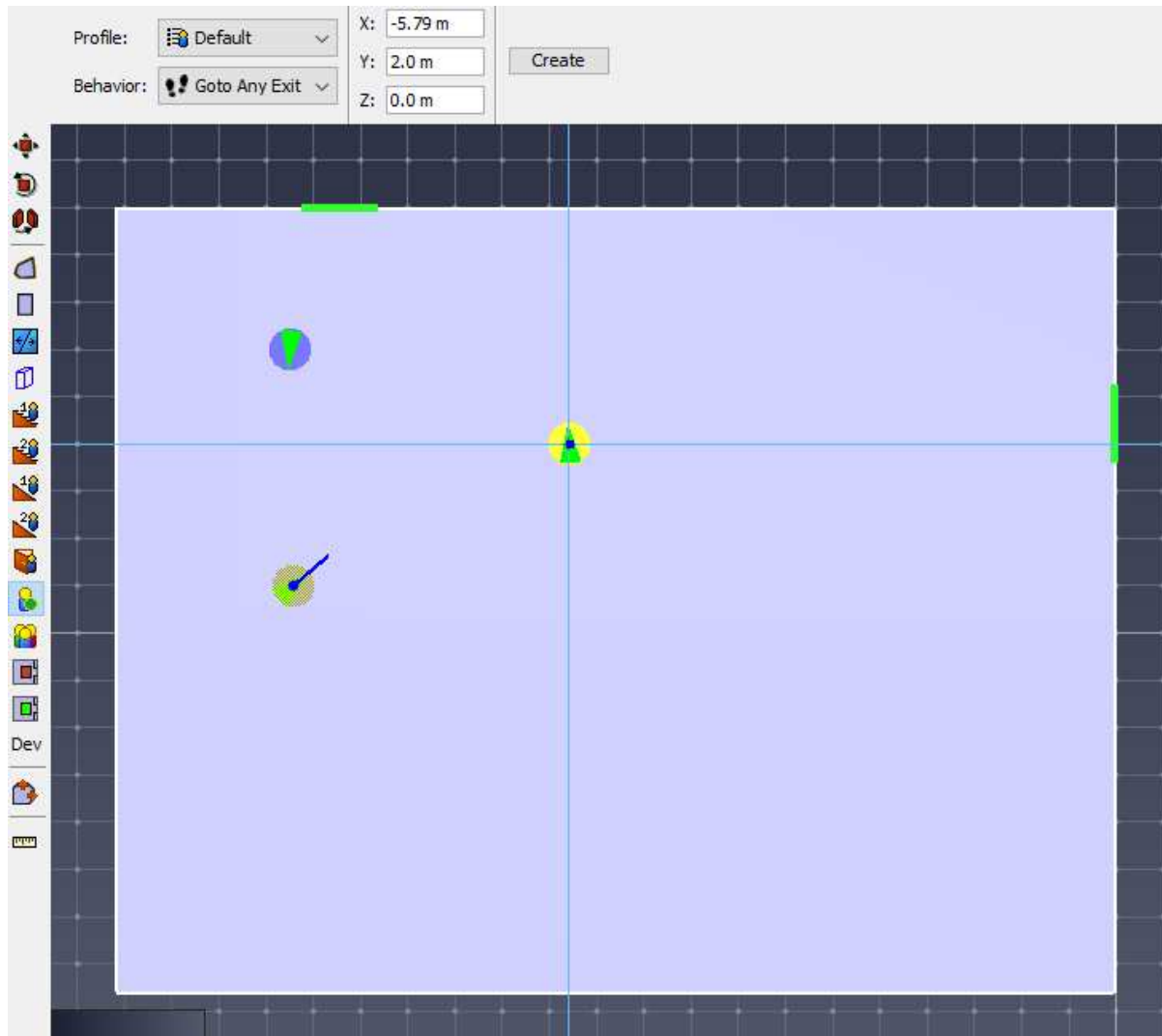



Figure 124. Adding occupants individually

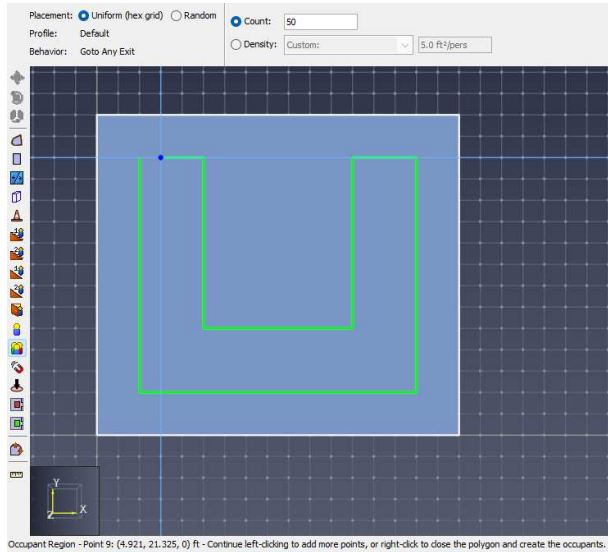
### 5.4.2. Group Placement

Groups of occupants can be added to the model with the **Add Occupants to a Region** tool . The occupants are distributed throughout the region using parameters in the property panel as shown in [Figure 127](#).

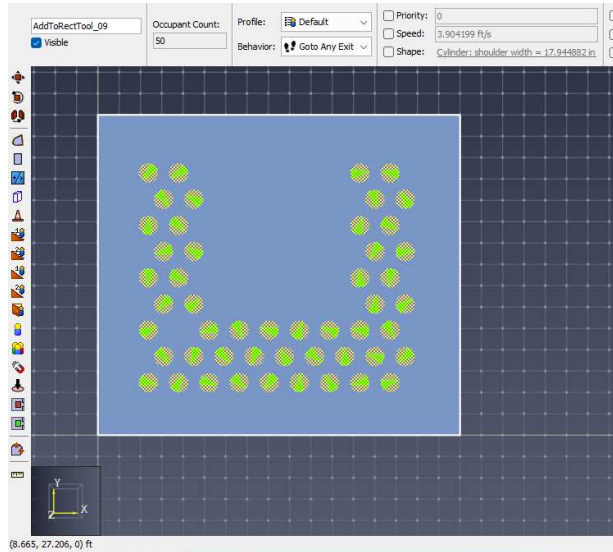
Once the properties have been specified, there are two ways to draw the placement region:

- Click+drag the left mouse button to place them in an axis-aligned rectangular region.
- Left-click multiple points to specify a polygonal region and right-click to finish the polygon.

When finished with the tool, the occupants will be placed within the designated region ([Figure 126](#)).



**Figure 125. Drawing a polygonal region for adding occupants**



**Figure 126. Occupants added to the polygonal region**

When an occupant is selected, the property panel allows the occupant's name, profile, and behavior to be edited. All profile parameters are also displayed, showing exactly what value will be used for that specific occupant. Any of these parameters can be overridden by checking the box next it and entering the desired value (see [Section 5.1.10](#)).



**Figure 127. Add Rectangular Group of Occupants Property Panel**

#### 5.4.2.1. Random or Uniform Placement

Random placement places occupants randomly within the designated area, such that no overlapping of occupants occurs. If the number of desired occupants is too great to accomplish this, a prompt will ask whether or not to continue with overlapping occupants. Uniform placement places occupants in an orderly hex pattern, allowing greater occupant densities before overlap occurs. Again, a prompt will ask whether to continue with overlap if the density is too great.

#### 5.4.2.2. Count or Density

This option specifies whether to place a set number of occupants or place enough occupants to achieve a certain density. Several template densities are provided, and **Custom** can be selected from the densities drop-down menu to enter a new value.



### 5.4.2.3. Profile

This option allows a distribution of profiles to be set for the occupants, such as specifying that 25% of the added occupants should be females < 30 years old, 30% children, etc. The label shows the currently set distribution, and if there is more than one profile defined in the model, the value can be clicked to edit the distribution. See [Section 2.6.2](#) for help specifying distributions.

### 5.4.2.4. Behavior

This option allows the distribution of behaviors to be set and is set by specifying percentages of each behavior.

## 5.4.3. Room Placement

In addition to distributing occupants in placement regions, occupants can be distributed throughout entire rooms.

To do this:

1. Select the desired rooms and choose **Add Occupants** from the **Model** menu or the right-click menu as shown in [Figure 128](#). This will bring up the Add Occupants dialog ([Figure 129](#)). For an explanation of the dialog's options, please see [Section 5.4.2](#).
2. Click the **OK** button after selecting the desired options to place occupants and exit the dialog.

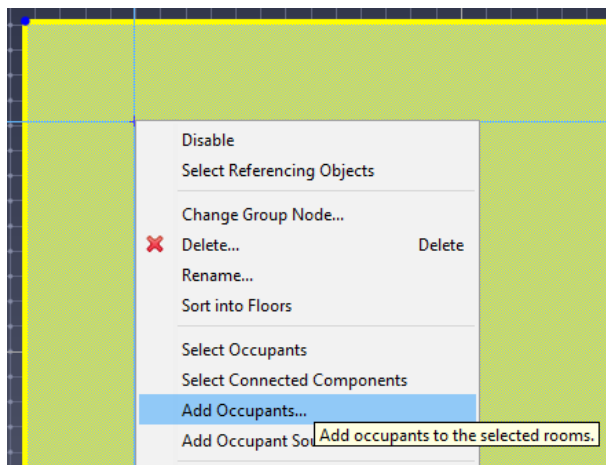


Figure 128. Add occupants in context menu

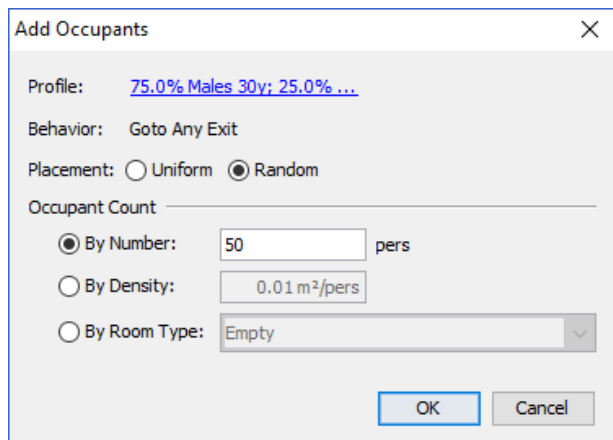


Figure 129. Add occupants dialog

### 5.4.4. Creating from Occupant Targets

Occupants can also be created from existing Occupant Targets (see [Chapter 9](#)). To do so, perform the following:

1. Select the desired Occupant Targets where occupants are to be placed.
2. Right-click the selected targets and choose **Create occupants from targets**. If there are multiple occupant profiles or behaviors available in the model, a dialog will appear asking for the desired distribution of each. Enter the desired distributions and click **OK**.
3. A dialog will appear asking to delete the selected occupant targets. Selecting **Yes** will delete the targets, effectively converting them into occupants.

Occupants will then be generated based on the selected distribution of profiles and behaviors:

**NOTE**

Resulting occupants may overlap each other, but are not allowed to overlap room boundaries. This may result in some Occupant Targets remaining empty. If this is the case, a warning message will indicate how many targets cannot be filled.

### 5.4.5. Importing from a CSV File

Occupants can also be created by importing their locations from a CSV file. In the CSV file, each row represents a single occupant. The file must contain three columns, specifying the X, Y, and Z coordinates in meters for each occupant location.


An optional occupant name can be specified in the fourth column. If a name is not specified in the occupant's row then a name will be generated for them.

To import the occupants, perform the following:

1. Add the desired [occupant profiles](#) and [behaviors](#) to your model.
2. In the **File** menu, select **Import**.
3. Select the CSV file containing the occupant locations.
4. In the import dialog, specify a distribution of occupant profiles and behaviors to apply to the generated occupants. See [Section 2.6.2](#) for help specifying distributions.
5. Click **Finish** to add the occupants.

### 5.4.6. Occupant Sources

Occupant sources define areas in which occupants are generated dynamically throughout the simulation.

To add an occupant source, use the Add Occupant Source tool  or right click on a door or a room in the model, and select **Add Occupant Source**. If an occupant source that is attached to a component is later deleted from the model the user will be notified and the occupant source will either be attached to a different component or deleted.

There are three options for specifying the physical location of the occupant source within the model, as shown in [Figure 130](#).

**Figure 130. New Occupant Source Options**

An occupant source can be defined by a rectangular region or attached to a component in the model.

### From Rectangle

Use this option to draw a box that defines a placement region. This region is intersected with the navigation mesh, and new occupants are created at random positions in the intersected areas.

### From Door

Use this option to attach the new occupant source to an existing door. The door can be selected either from the drop-down menu or by clicking on the component in the model. New occupants will be created at random positions along the door.

- If the door is one-way, the occupants will be placed slightly off of the door toward the one-way direction.
- If the door is an exit, the occupants are placed slightly inside the model.
- If the door is attached to an elevator, occupants are placed slightly outside the elevator.

### From Room

Use this option to attach the new occupant source to an existing room. The room can be selected either from the drop-down menu or by clicking on the component in the model. New occupants will be created at random positions inside the room.

Once the location of an occupant sources is defined, additional parameters that relate to the occupants that are generated can be specified. [Figure 131](#) below shows parameters that can be specified for an existing occupant source.

**Figure 131. Parameters of an Occupant Source**

**Occupant Count (read-only)**

The approximate total number of occupants that will be created by the selected occupant sources.

**Flow Rate**

Rate at which new occupants will be created ([Section 5.4.6.2](#)).

**Enforce Flow Rate**

Specifies whether the occupant source should keep generating occupants regardless of how crowded its area is ([Section 5.4.6.3](#)).

**Occupant Locations**

The locations of generated occupants ([Section 5.4.6.4](#)).

**Profile**

Specifies a profile distribution. See [Section 2.6.2](#) for help specifying distributions. See [Section 5.4.6.1](#) for more information on randomizing occupant source distributions.

**Behavior**

Specifies a behavior distribution. See [Section 5.4.6.1](#) for more information on randomizing occupant source distributions.

**Component**

Shows which door or room the occupant source is attached to, if any.

**Movement Group Template**

Specifies the distribution of movement groups for occupants that are part of a movement group. An *Ungrouped* option is available for occupants that will not be part of any movement group. (See [Section 8.4](#))

**Emit at Max. Velocity**

Whether occupants should be traveling at their maximum velocity when they are generated by the source. If this is set to **No**, the occupants will start with a speed of 0 and have to accelerate to their maximum velocity.

**NOTE**

This parameter only affects steering mode, as SFPE occupants have infinite acceleration.

**Initial Orient**

The initial orientation of the occupants when they are generated ([Section 5.4.6.5](#)).

#### 5.4.6.1. Occupant Source Seed

Each occupant source contains a non-editable seed used in the randomization of profile, behavior, and flow rate distributions. This ensures that every time a simulation is run with the same seeds, the results are deterministic. In order to generate a new set of results, however, the seeds can be re-generated. To do so, perform the following:

1. Select the occupants sources that are to be randomized.
2. Right-click the selected sources and select **Randomize**.

#### NOTE

Random flow rate distributions are only available when using the **Scheduled** option with a **Poisson** or **Random** distribution (see [Section 5.4.6.2.4](#)).

#### 5.4.6.2. Flow Rate

The occupant source **Flow Rate** is the rate at which new occupants will be created. There are a number of methods for specifying the flow rate as discussed below.

##### 5.4.6.2.1. Constant

Specifies the flow rate as a constant value in persons / second. The occupant source will continue flowing until the simulation end time ([Section 15.1.1](#)).

##### 5.4.6.2.2. From Table

Specifies the flow rate as a function from a table, where the flow rate can vary over time. The flow rate table editor is shown in [Figure 133](#) below. During the simulation, at any given time, the number of occupants that are released from the occupant source is calculated based on the integral of the resulting function up to that time, minus the number of occupants who have already been released.

#### NOTE

The **From Table** option requires transition periods between varying flow rates and does not support random insertion times. If this does not meet the desired needs, the **Scheduled** flow rate option may be a better fit.

The editor can also be used to automatically construct a periodical step function. To do so, click the **Step Function** button. The **Create Step Function** dialog is shown in [Figure 132](#) below. The editor with a generated step function is shown in [Figure 133](#) below.

The following are the parameters of the step function that can be set in the **Create Step Function** dialog:

**Flow Rate**

Flow rate value when occupant generation is ON.

**Initial Delay**

The amount of time at the beginning of the simulation until the step function begins.

**On Duration**

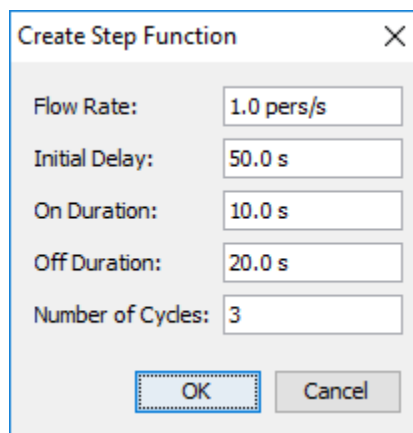
Amount of time in each period for which occupant generation is turned ON.

**Off Duration**

Amount of time in each period for which occupant generation is turned OFF.

**Number of Cycles**

The number of ON/OFF pairs in which occupants are generated. After this, the occupant source will stop generating occupants.

A screenshot of a software dialog box titled "Create Step Function" with a close button (X) in the top right corner. The dialog contains five input fields, each with a label and a value: "Flow Rate:" with "1.0 pers/s", "Initial Delay:" with "50.0 s", "On Duration:" with "10.0 s", "Off Duration:" with "20.0 s", and "Number of Cycles:" with "3". At the bottom of the dialog are two buttons: "OK" and "Cancel".

Flow Rate:	1.0 pers/s
Initial Delay:	50.0 s
On Duration:	10.0 s
Off Duration:	20.0 s
Number of Cycles:	3

OK Cancel

**Figure 132. Create Step Function Dialog**

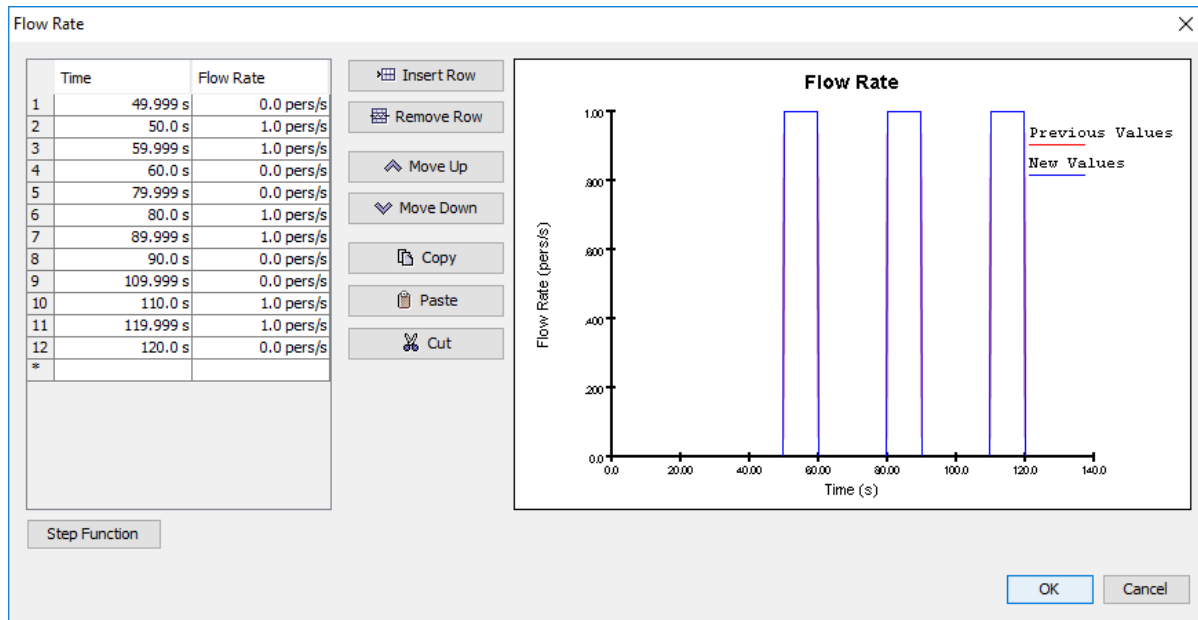


Figure 133. Occupant Source Flow Rate Editor

#### 5.4.6.2.3. By Time

This option allows the specific occupant entry times to be specified in a table. This may be useful in re-creating observed data. The number of non-empty entries in the table determines the number of occupants who will be generated from the occupant source.

#### 5.4.6.2.4. Scheduled

With the **Scheduled** flow rate option, the insertion times can be specified using timing intervals. Unlike the **From Table** option, the **Scheduled** option allows for random insertion times and instantaneous transitions between varying flow rates.

**Flow Rate**

Timing Distribution: Poisson

Insertion Intervals

	Time	Duration	Num. Occupants
1	0.0 s	10.0 s	13
2	10.0 s	10.0 s	10
3	20.0 s	10.0 s	0
4	30.0 s	10.0 s	5
*			

Buttons: Insert Row, Remove Row, Move Up, Move Down, Copy, Paste, Cut

Step Function

Total Occupants: ≈28

OK Cancel

Figure 134. Occupant Source Scheduled Flow Rate Editor

Each row of the table represents one insertion interval. The **Time** column indicates the start of that insertion interval. The **Duration** column indicates how long that insertion interval will last. The end time of the interval is **Time + Duration**. Intervals may not overlap. The **Num. Occupants** column indicates how many occupants will be released during that interval. The approximate flow rate for the interval is **Num. Occupants / Duration**.

**NOTE**

In some cases, there may be intervals where no occupants should be generated. These intervals can either be left out of the table or the **Num. Occupants** can be set to 0.

The **Timing Distribution** indicates how the occupant insertion times will be distributed during each interval. The following options are available:

**Uniform**

Occupants are distributed at uniformly-timed intervals during each insertion interval. The total number of occupants is guaranteed as long as the flow rate is enforced and a valid location can be found on the occupant source.



### Poisson

Occupants are inserted at random times during each insertion interval using a Poisson distribution. The total number of occupants is not guaranteed, though the number should be close to the approximation as long as the flow rate for each interval is sufficiently high enough, the flow rate is enforced, and a valid location can be found on the occupant source.

### Random

Occupants are inserted at random times during each insertion interval using an approximately uniform distribution. Longer intervals produce more randomness, but the total number of occupants is guaranteed as long as the flowrate is enforced and a valid location can be found on the occupant source.

#### NOTE

When using the **Poisson** and **Random** distributions, the timing distributions can be re-randomized between simulations by generating a new seed (see [Section 5.4.6.1](#)).

As with the [From Table](#) option, the **Scheduled** option can be used to create a step function with the **Step Function** button.

#### 5.4.6.3. Enforce Flow Rate

When **Yes** is selected, the occupant source will keep generating occupants regardless of how crowded its area is, which can result in overlapping occupants but will help guarantee the number of generated occupants. When **No** is selected, the occupant source will only generate an occupant if the new occupant will not overlap with another occupant. Otherwise, the occupant source will wait until the next time step to try to generate an occupant again.

#### 5.4.6.4. Occupant Locations

The occupant source **Occupant Locations** table can be used to enter known locations for occupants when they enter the simulation. This might be useful, for instance, to compare a simulation to experimental data, where the exact initial positions of the occupants need to be reproduced.

In the table, enter the initial locations for each generated occupant in each row. The order of the rows matches the order in which occupants are generated by the occupant source, as determined by the [Flow Rate](#). For instance, the first row is the location of the first generated occupant. The second row is for the second occupant, etc. If a row is left blank or there are fewer entries in the table than generated occupants, some occupants will not have a corresponding entry in the table. In this case, those occupants will be assigned a randomly generated location within the occupant source region.

**NOTE**

The locations in the table do *not* have to be within the occupant source region. They can be any valid location on the navigation mesh.

For example, suppose the occupant locations table contains the following values:

**Table 13. Example Occupant Locations Table**

X	Y	Z
1	0	5
2	-5	3

Also, suppose the occupant source flowrate produces 5 occupants. The first occupant will be generated at (1,0,5), the second occupant will be generated at a random location within the occupant source region, the third occupant will be generated at (2,-5,3), and the third and fourth occupants will also be assigned random starting locations.

#### 5.4.6.5. Initial Orientation

The **Initial Orientation** property specifies the direction an occupant will be facing when they are generated. If this box is unchecked, the initial orientation of the occupant is determined by the **Initial Orientation** property of the occupant's profile (see [Section 5.1.2](#)). If the box is checked, however, the orientation can either be set to <automatic> or to a distribution of orientation angles. The orientation angle is specified as an angle going counter-clockwise from the positive X axis.

If the **Initial Orientation** is set to <automatic>, the orientation is determined as follows:

1. If the occupant source is an exit, the orientation is set to the direction into the model.
2. If the source is a door with a one-way direction, the orientation is the same as the one-way direction.
3. If the source is a door without a one-way direction, the orientation is a direction away from the door that aligns with the seek direction of the most recently released occupant.
4. If the occupant source is a region, the orientation is the travel direction of the most recently released occupant.

**NOTE**

Because the last two options rely on the most recently-released occupant, the initial orientation might be incorrect for the first generated occupant. If this causes issues, it may be more desirable to explicitly set an initial orientation rather than relying on the <automatic> option.

## 5.5. Redistributing Profiles and Behaviors

Once occupants have been created, the distribution of profiles and behaviors can be reshuffled among them.

1. Select occupants, right-click one of the selected occupants either in the Navigation view or Model view, and then select **Properties** from the right-click menu. To edit a single occupant or group, double-click the occupant or group in the Navigation view. The Edit Selected Occupants dialog will appear as shown in [Figure 135](#).
2. If more than one profile exists in the model, the **Profile** link can be clicked to edit the distribution as discussed in Group Placement.
3. If more than one behavior exists in the model, the **Behavior** link can be clicked to edit its distribution.

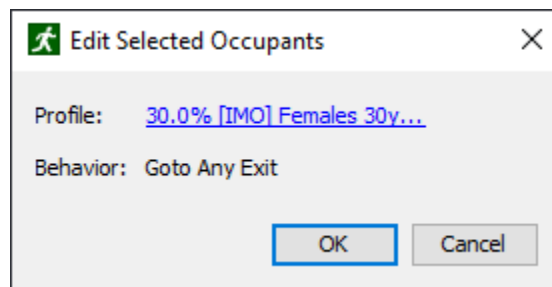


Figure 135. Edit Selected Occupants dialog

### NOTE

Changing the distribution of profiles or behaviors will not change the number of occupants in the group. It will just change which profile/behavior is assigned to each occupant to match the specified distribution as closely as possible.

## 5.6. Randomizing Occupants' Positions

After occupants have been created, their positions can be changed to new random positions.

To randomize occupants' positions, select one or multiple rooms, and from the right-click menu select **Randomize Occupants' Positions**. The **Randomize Occupants' Positions** dialog appears ([Figure 136](#)), providing the option to change room and position settings.

- **Reposition within Current Room** will make each occupant stay in its current room, while **Reposition within Any Selected Room** will move occupants freely within selected rooms.
- **Random** and **Uniform** options will redistribute occupants randomly or uniformly, respectively.

After clicking **OK**, all occupants in the selected rooms will be moved to their newly generated

positions. Occupants' orientation will change too, unless it is defined locally for specific occupants or set to a constant angle in the occupants' profile.

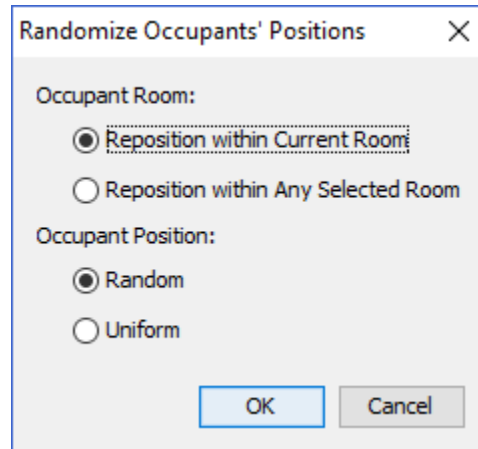


Figure 136. Randomize Occupants' Positions Dialog

## 5.7. Reducing Population

It is sometimes necessary to reduce the number of occupants in the model. The number of occupants can be decreased in each room, stair, ramp, or occupant group automatically.

To reduce the number of occupants in a component or in an occupant group, right click on the specific object and select **Reduce Population**. A dialog will appear with the option to set the new occupant count. The deletion method can also be set.

Pathfinder can either keep random occupants, first occupants in the selection, or last occupants in the selection.

## 5.8. Occupant Tags

Occupants can have multiple tags applied to them. These tags act as a kind of label that can help identify related occupants during the simulation. They can be used for the following purposes:

- Choosing a destination occupant for the [Goto Occupant](#) behavior action.
- Choosing a destination occupant for the [Look At](#) behavior action.
- Specifying the occupants with which to perform social distancing (see **Social Distance Occupants** in [Section 5.1.7](#)).
- Determining which occupants can use a trigger (see [Section 10.3.1](#)).
- Reporting in results files (see [Section 16.8](#) and [Section 16.9](#)).

There are multiple ways that tags can be applied to occupants, including the following:

- Setting the **Tags** profile/occupant property. These tags are applied to the occupants at the beginning of the simulation or when they are first added to the simulation (see [Section 5.1](#)).
- Setting the **Occupant Tags** property for rooms, stairs, or ramps. These tags are applied to the occupant when they enter the room and removed when they exit. (see [Section 3.2.5](#)).
- Setting the **Occupant Tags** property for doors. These tags are applied to the occupant when they cross the door (see [Section 3.5.3](#)).
- Using a [Change Tags](#) behavior action to change the tags during the simulation. This can be used to add to, remove from, or replace the occupants' current set of tags as part of their behavior sequence.

# Chapter 6. Queues

In Pathfinder, users can define occupant-organizing structures called Queues. These can be defined as an object that consists of the following:

1. One or more [Services](#)
2. One or more [Paths](#), consisting of [Path Nodes](#)

Occupants who join a queue will move immediately into an available service, where they will stay for a user-defined amount of time. If all services in a queue are occupied, occupants will begin to form a line along the path(s). When occupants finish their wait at the service, they will be released to their next behavior action while any occupants behind them will move forward in the queue.

As the model runs, queues will attempt to fill their service(s) as soon as they are able, filling in from occupants who have joined the structure first. If there are multiple Paths waiting for the Service(s), queues will choose from them evenly.

When it is desired that a particular path always leads into a particular service, it is important to model the scenario using a single queue. If an occupant has a choice between multiple queues with a similar structure of paths and services, it is best to specify all of the potential queues in the [Goto Queue](#) behavior action.

## NOTE

Full support for queues with movement groups is not yet available. Occupants in movement groups may become stuck when using queues.

## 6.1. Services

Services are the destinations of a Queue, defined by a point on the navigation mesh. When an occupant reaches one, they will wait at this point and then be released to their next behavior action; associated parameters designate the time for the occupant to wait before the queue releases them.

### Service Time

Amount of time an occupant waits at a service before being released by the queue. The wait time can be specified by a constant value or as a user-defined distribution.

## 6.2. Paths

Paths are the routes of a Queue that occupants follow and wait at to reach a service. They are linear sets of Nodes on the navigation mesh, and occupants will form a line along the path between the nodes, moving forward to the exit end of the path and to the service when they are

able.

Once created, paths have the following parameters:

**Follow Path**

Specifies whether an occupant follows the given path from beginning to end. When checked, a joining occupant will follow every point on the path before getting to their place or service.

When unchecked, an occupant will go directly to their place in line or at a service.

## 6.3. Path Nodes

A path's Nodes are the points in space that defines the route. The points are ordered from top to bottom in the tree, representing the nodes by distance from a service. They may be shuffled around or deleted in any order once created. Additional Nodes may be added to or deleted from a path after the initial path is created. Path Nodes can be moved to modify the path through the model.

# Chapter 7. Assisted Evacuation

Pathfinder supports assisted evacuation scenarios, in which some occupants may assist other occupants. This is particularly useful in hospital evacuations and other scenarios where there may be some disabled occupants who need help for at least part of their journey.

The following is some terminology used in Pathfinder:

## **Assistant**

An occupant who helps other occupants [Section 7.1.1](#).

## **Client**

An occupant who is helped by assistants [Section 7.1.2](#).

## **Team**

A group of assistants [Section 7.3](#).

Assisted evacuation is supported through the behavior system of Pathfinder, allowing a wide range of scenarios to be investigated.

Some possibilities include:

- Assistants help clients through their entire evacuation, allowing the client to visit multiple intermediate waypoints or rooms or wait at a location while being assisted.
- Assistants help clients for only part of their evacuation, such as an occupant in a wheelchair who only needs assistance to descend stairs.
- Assistants help in stages. For example, one team of assistants moves clients to one location and then another team moves the occupants to another location.

Assisted evacuation is also supported by the occupant source system (see [Section 5.4.6](#)), which allows clients and assistants to be generated by occupant sources.

## 7.1. Assistance Process Overview

Technical details of the algorithm for assisted evacuation are available in the Pathfinder Technical Reference ([Pathfinder Resources](#)), but a summary from both the assistants' and the clients' perspectives are given here.

### 7.1.1. Assistants

Initially, the occupant becomes an assistant of a team based on their **Assist Occupants** action ([Section 5.3.3.20](#)).



- The assistant checks whether any current clients or future clients (from an occupant source) will ever need assistance from them, and, if not, continues on with their next behavior action.
- If the assistant determines clients are available, they will send an offer to assist a client.
- When the client accepts the request, the assistant attaches to the client and becomes passive until the client detaches.
- When there are no more clients available, the assistant moves on to the next behavior action.

### 7.1.2. Clients

When the simulation begins, the occupant becomes a client of a team or set of teams based on the client's **Wait for Assistance** action ([Section 5.3.3.21](#)) and then the client waits indefinitely for an assistant from one of those teams to offer assistance. When the client receives an offer, they choose the closest assistants to assign to all open positions on their vehicle (any extra offers are rejected).

All subsequent behaviors are processed with assistants attached until one of the following happens (depending on the terminating action):

- The client encounters a **Goto Exits** action, in which case they detach from all assistants shortly before going through the exit and then proceed to go through the exit on their own, even if they cannot move without assistance. They are then removed from the simulation.
- The client encounters a **Goto Refuge Rooms** action, in which case the client will go to one of the rooms. Upon entering the room, the client will proceed to the back of the room, detach from the assistants, and wait until the simulation completes.
- The client encounters a **Wait Until Simulation End** action, in which case the client will immediately detach from assistants before waiting for the end of the simulation. This frees their assistants to help with other clients.
- The client encounters a **Detach from Assistants** action, in which case the client detaches from all assistants and continues on to the next action on their own. (Example: The client is in a wheelchair and needed assistance down stairs.)

## 7.2. Preparing Clients

There are some considerations to take into account when preparing clients. [Table 14](#) outlines some important client-specific parameters that can help achieve a variety of evacuation outcomes.

**Table 14. Important parameters to define client behavior**

Parameter	Location	Usage
<a href="#">Wait for Assistance Action</a>	Behaviors panel	Turns occupant into a "client"

Parameter	Location	Usage
<a href="#">Detach from Assistants Action</a>	Behaviors panel	Allows client to detach from assistants and make part of their evacuation journey individually
<a href="#">Vehicle Shape</a>	Occupants panel	Wheelchair and hospital bed are available by default  User-defined vehicles are also available
<a href="#">Requires Assistance to Move</a>	Edit Profiles Dialog	The client cannot move without assistance

Preparing occupants who can become clients can be done as follows:

1. Create a new [Behavior](#) for the clients.

- Add a [Wait for Assistance](#) action to the behavior and specify the team or teams that should assist the clients.
- Add actions that the client should perform while being assisted, such as going to a refuge room or going to an exit. If a **Goto Exits** action is added, the assistant will automatically detach from the client before the client goes through the door, allowing the assistant to continue helping other clients.
- If the client should perform part of the journey on their own, add a [Detach from Assistants](#) action to the behavior and then add the actions that will be performed individually.

2. Create the occupant.

- Assign the client behavior.
- Choose a **Vehicle Shape** for the client (see [Section 5.1.10](#) and [Figure 137](#)) by editing the **Shape** parameter on the occupant properties panel. *All clients must have a vehicle shape* (see [Section 5.2](#)). Either choose a default shape or define a custom shape with user-defined positions for assistants.

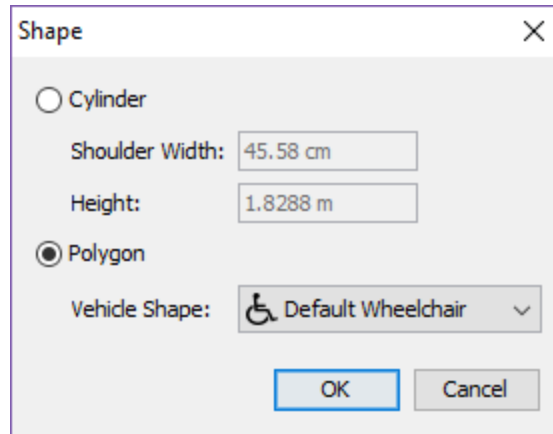


Figure 137. The Shape Dialog where a user may select a vehicle shape for a client

## 7.3. Preparing Assistants

Preparing occupants who can become assistants can be done as follows:

1. Create a new **Behavior** for the assistants.
2. Add an [Assist Occupants](#) action to the behavior and select the **Team** on which the occupant should assist (if no team is selected the default team will be used).
3. Create the occupant or occupant source and assign the **Behavior**.

## 7.4. Preparing Teams

Any number of assisted evacuation teams may be created. Assistants join teams through their behaviors and clients ask for assistance from teams through their behaviors. By default, assistants join the default team.

Assistants may only be a member of one team at a time, but they may be a member of several teams during the course of the simulation. This is accomplished by using an **Assist Occupants** action for each team in the assistant's behavior.

To edit assisted evacuation teams, on the **Model** menu, click **Edit Assisted Evacuation Teams** (see [Figure 138](#))

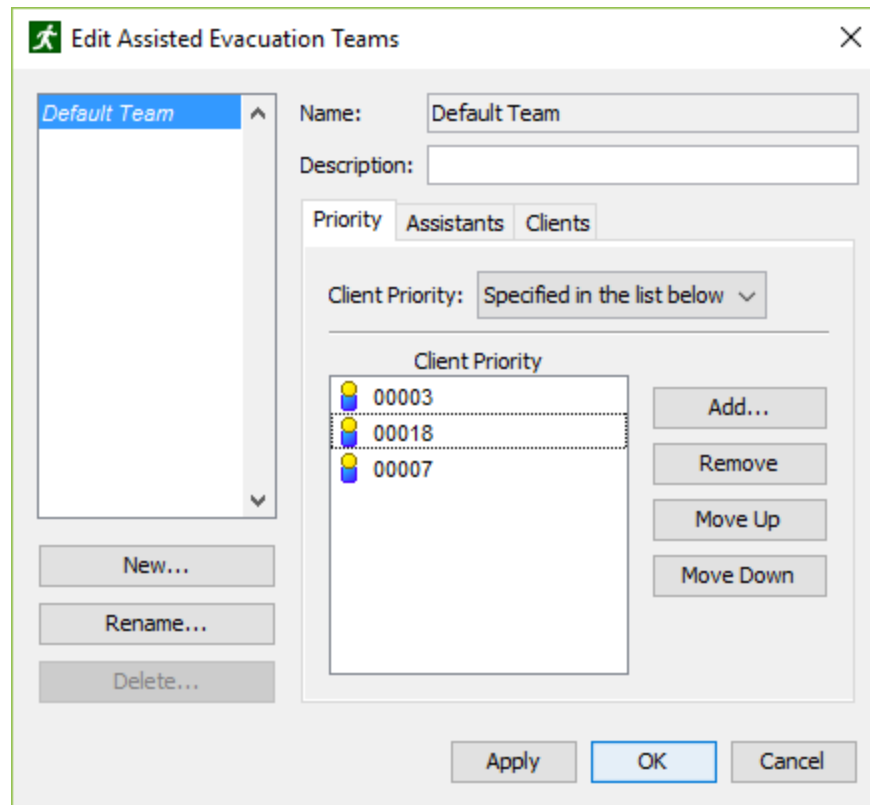


Figure 138. The Edit Assisted Evacuation Teams dialog

On the **Priority** tab, **Client Priority** controls the order in which clients are evacuated by a team.

The priority can be specified as follows:

#### Distance to assistants

Assistants on the team will choose the closest client to them.

#### Specified in the list below

Allows the client ordering to be specified directly by adding clients to a list. Clients in the list are given priority over those absent from the list. The clients in the list are assisted according to the list order. Those absent from the list are assisted based on distance. Once an assistant chooses a client, they are locked into the choice.

For example, say there are three clients listed in the order shown in Figure 138 - "00003", "00018", "00007".

The following behavior may occur:

1. Client "00018" requests help from "Default Team", but no other clients have yet requested assistance.
2. An assistant helping with "Default Team" chooses to assist client "00018".
3. As the assistant is heading toward "00018", client "00003" requests assistance from the

team.

4. The assistant continues on toward "00018", ignoring the request from "00003".
5. Once client "00018" is helped, the assistant heads toward "00003".

**NOTE**

Any occupant can be added to this list, even if their behavior never requests assistance. In this case, they are simply ignored.

The **Assistants** tab is read-only and shows all assistants who currently have a behavior that joins the team with an **Assist Occupants** action at some point in the simulation. You must use an **Assist Occupants** action to add an assistant to this list.

The **Clients** tab is read-only and shows all clients who currently have a behavior that requests assistance from the team with a **Wait for Assistance** action at some point in the simulation. You must use a **Wait for Assistance** action to add a client to this list.

## Chapter 8. Occupant Grouping

Movement groups are used to keep occupants together during the simulation. This feature can be used to explicitly link occupants present at the start of the simulation and can also be used with occupant sources to link occupants inserted during the simulation. The first case might be used in a simulation that begins with families already seated together in a theater and the second might be used to simulate group arrivals via railway car.

Occupants that are part of a group remain grouped during the entire simulation - there is no facility for leaving and joining groups. In addition, to be members of the same group, occupants must use the same behavior.

While occupant movement groups can be used to create simulations that have the appearance of more realistic behavior, specific data about grouping is generally unavailable in validation studies and the relationship between movement times and specific grouping parameters is unknown. Because grouping adds constraints and never increases the individual movement speed of an occupant, it should generally produce more conservative (i.e. slower) simulation times than the ungrouped equivalent. However, if using groups in an evacuation simulation, it is highly recommended to run the simulation both with and without grouped occupants to identify the impact of occupant groupings on evacuation times.

For output associated with grouped movement, see [Section 16.10](#).

### 8.1. Grouped Movement

Grouped movement is controlled by two concepts: connected state and the group leader. If a group is in a "disconnected" state, occupants will walk toward the leader. If a group is in a "connected" state, occupants move toward the goal dictated by their behavior. This framework allows groups that have become broken up to re-form and avoids imposing too harsh movement restrictions related to divided doors and other minor navigational differences.

The details of these behaviors are controlled by the following four parameters:

#### **Follow Leader**

This parameter makes it possible to manually specify the occupant that will be used as the group leader throughout the simulation. If left unset, the simulator automatically selects a leader and the leader may change over time. The group member nearest the current goal, as specified by behavior, will become the leader.

#### **Maximum Distance**

This parameter is used to determine if a group is connected or disconnected. If any part of the group is more than Maximum Distance away from any other part, the group is considered

disconnected. More specifically, the longest edge in the minimum spanning tree connecting all occupants in the group must be less than or equal to Maximum Distance for the group to be considered connected.

### Slowdown Time

This parameter is used by the leader when a group has become disconnected. The leader will gradually reduce speed before coming to a stop. Slowdown Time determines how long the leader will slow down before stopping.

### Enforce Social Distancing Between Group Members

This parameter is used to enforce Social Distancing between group members (see [Section 5.1.7](#)). By default, social distancing is not enforced within the group, but group members will social distance with occupants not in their group.

Smaller values for Maximum Distance and Slowdown Time give more cohesive groups and larger values give looser groups.

Group members share component restrictions (see [Section 5.1.2](#)). If a component is restricted for some group member, other group members will avoid the component as well.

## 8.2. Movement Groups

Movement Groups represent groupings of specific occupants that exist at the start of the simulation. They are listed in the Tree view within the Movement Groups node. They also appear in the property panel when selecting occupants belonging to the same Movement Group. Clicking the name of a Movement Group in the property panel will select it in the Tree view.

To create a Movement Group:

1. Select all occupants that will be members of the group.
2. Right click to activate the pop-up menu, click **New Movement Group from Selection**.

The new Movement Group will appear in the Tree view within the Movement Groups node. The Movement Group parameters (e.g. Maximum Distance) can be edited by selecting the Movement Group in the tree.

Manually creating individual Movement Groups is likely only necessary for special cases. Movement Group Templates offer a way to automatically create Movement Groups.

## 8.3. Movement Group Templates

Rather than describing a specific set of occupants that will move together, Movement Group

Templates describe a *kind* of Movement Group. An example might be "Families, with 1-2 adults and 1-3 children". Pathfinder can use these descriptions to automatically create large numbers of Movement Groups from selections of occupants or during the simulation.

In addition to the three movement parameters described earlier, Movement Group Templates require occupant profile-based creation parameters:

### **Number of Members**

This option can be used when control over the specific profiles used to form the group is not required. It specifies constraints on group size.

### **Numbers of Members per Profile**

This parameter can be used to specify group size constraints along with profile requirements.

### **Leader's Profile**

This option can be used to force the permanent leader to be selected from a specific profile.

This option is only available when the Number of Members per Profile option is used and the Follow Leader option is selected.

To create a Movement Group Template:

1. On the **Model** menu, click **Edit Movement Group Templates**
2. Click **New**
3. Type a name for the template and then click **OK**.

The new template will be added to the Tree view in the Movement Group Templates node. The created template can now be used to automatically create movement groups.

When Pathfinder uses Movement Group Templates to sort occupants into movement groups, the algorithm is controlled by the following parameters:

### **Movement Group Templates**

This is the distribution of templates that should appear in the resulting movement groups.

Click the blue text to edit the distribution. For example, a theater seating area might include 60% groups of 1-4 adults, and 40% groups of families. See [Section 2.6.2](#) for help specifying distributions.

### **Restrict Members to the Same Room**

This parameter is active by default and enforces a rule that all members of a group must share the same starting room.



## Distance Calculation

The group creation algorithm attempts to form groups from clusters of occupants standing near each other. Clusters are identified using either Travel Distance, which takes walls and other obstacles into account, or Geometric Distance. Geometric Distance is fast because it uses a simple distance calculation, but it may have the undesirable consequence of grouping occupants that are nearby, but on different floors. A height multiplier can be used to discourage the algorithm from grouping occupants located up or down a floor. For example, using a multiplier of 10 would make a 3.0 meter difference in Z effectively a 30.0 meter difference for purposes of the distance calculation.

To create Movement Groups using a template:

1. Select all occupants that should be sorted into groups. All selected occupants must share the same behavior.
2. Right-click and on the pop up menu, click **New Movement Group(s) from Template**
3. The **New Movement Group(s) from Template** dialog offers various options that provide control over how Pathfinder will sort the selected occupants into groups.
4. Adjust parameters as required and click **OK**.

The new movement groups will be added to the Navigation view in the Movement Groups node and the parameters of individual groups can be edited if required.

When using the New Movement Group(s) from Template action, it is important to note that occupant profiles will be reset by the operation.

## 8.4. Adding Grouped Occupants from Occupant Sources

Sources insert occupants into the model while the simulation is running. By default, sources do not use groupings when adding occupants. Movement Group Templates can also be used to create movement groups dynamically as occupants are added to the simulation from sources. Sources can create grouped occupants using the following parameter:

### Movement Group Template

The distribution of templates that will be used when creating grouped occupants. The default value is *Ungrouped*, which means that the generated occupants will not be part of any movement group. Since groups usually consist of more than one occupant, the source will add all members of a new group one at a time at the prescribed flow rate until the group has been completely added. Once complete, the distribution of movement group templates will again be used to determine grouped (or ungrouped) status of the new occupant.

To configure an occupant source to add grouped occupants:

1. Select the occupant source.
2. On the property panel, in the **Movement Group Template** field, click to enter a distribution of templates to be selected when inserting occupants.

The source will now insert grouped occupants based on the template settings.

## Chapter 9. Occupant Targets

An **Occupant Target** represents a location on the navigation mesh that can be occupied by a single occupant at a time. Occupants obtain a reservation to a target using a reservation system and can occupy a target for any amount of time. They can even keep a reservation for a target when they are not occupying it, preventing others from using it.

Occupant Targets are useful in a variety of scenarios, including ingress, general movement, and evacuation. For ingress, targets can represent seating, where occupants reserve a seat, move toward it, and then wait for an indefinite amount of time. Similarly, in general movement the targets can also represent seating, but occupants can occupy a seat for a limited time before giving it up for another occupant to fill, such as in a food court or airport terminal waiting area. Targets can also be useful in evacuation, including representing planned refuge locations for assisted evacuation or defining locations at which individuals must perform some action prior to evacuating.

Targets are represented as circular areas on the navigation mesh with a line extending from the center, representing their orientation as shown below.

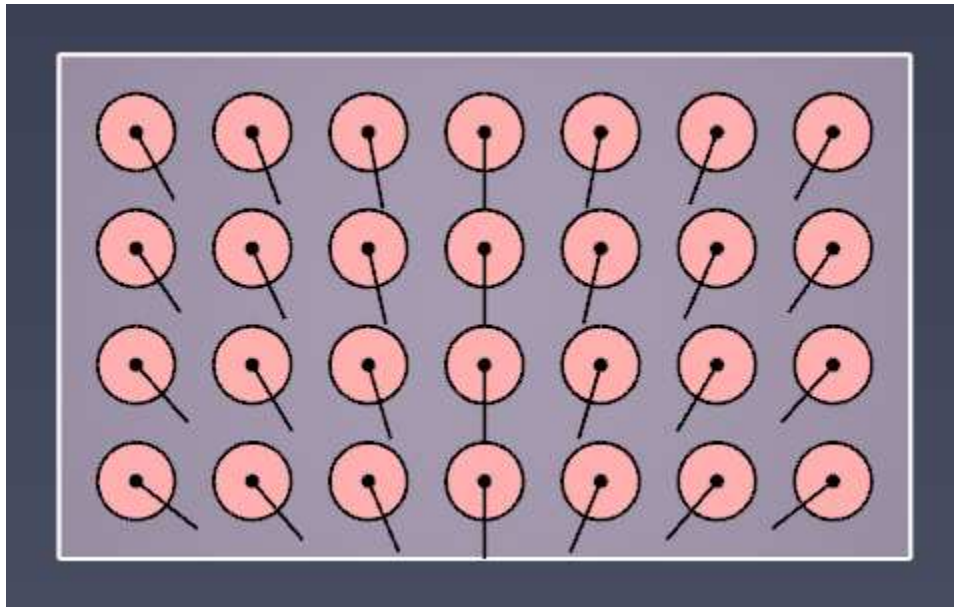



Figure 139. Occupant Target Display

### 9.1. Creating Occupant Targets

There are two main methods for creating Occupant Targets. Individual targets can be placed using the Occupant Target tool. Alternately, multiple targets can be created from existing occupants.

### 9.1.1. Individual Placement

To create an Occupant Target using the Occupant Target tool, perform the following steps:

1. In the 2D or 3D View, click the **Add an Occupant Target** drawing tool .
2. Enter desired properties for the target.
3. Either click on the navigation mesh or enter a location in the property panel and click **Create** to place the target. Alternately, click-drag to set both the location of the target and its orientation before creating it.

### 9.1.2. Creating Multiple Occupant Targets

There are currently no tools to directly add multiple Occupant Targets to the model, but targets can be created from existing occupants. This indirectly allows existing occupant placement tools to also be used to create Occupant Targets.

To create multiple targets in this manner, perform the following:

1. Add occupants who will represent the target locations using one of the existing occupant creation tools (see [Section 5.4](#)).
2. Select the occupants where the targets are to be placed.
3. Right-click the selected occupants and choose **Create Occupant Targets**.

A dialog will appear asking whether to delete the selected occupants. Choosing **Yes** will delete them, effectively converting them into Occupant Targets. Choosing **No** will leave the selected occupants as-is, creating Occupant Targets at their locations.

The resulting Occupant Targets' orientations will match the initial orientations of the selected occupants.

## 9.2. Occupant Target Properties

When an Occupant Target is selected, the following properties can be edited in the property panel:

#### Orientation

The direction an occupant should face when waiting at the target. This value can be edited directly or can be generated as discussed in [Section 9.3](#).

#### Priority

The priority of the target. This is optionally used by an occupant's behavior when making reservations. (see [Section 9.6](#)). This value can also be edited directly or can be generated as discussed in [Section 9.4](#).

## 9.3. Orienting Occupant Targets

The orientation of an Occupant Target can be edited directly by selecting the target and editing the **Orientation** property in the property panel. The orientation can also be adjusted by manipulating the orientation handle in the 2D/3D view (see [Section 12.2.9](#)).

Both of the above methods may be time-consuming for models with many Occupant Targets. A faster approach is to orient the handles of multiple targets at once, which can be accomplished in two different ways. Targets can either be oriented so they all face a reference point or they can be randomized.

To orient the targets toward a reference point, perform the following:

1. Select the Occupant Targets to orient.
2. Right-click the selection and choose **Orient on Point**.
3. Either enter a reference point in the **Choose Target Point** dialog that appears or click a reference point in the 2D/3D view and then click **OK** in the dialog.

All selected targets will now be oriented so that they face the specified reference point. The images below demonstrate some orientations generated using this action.

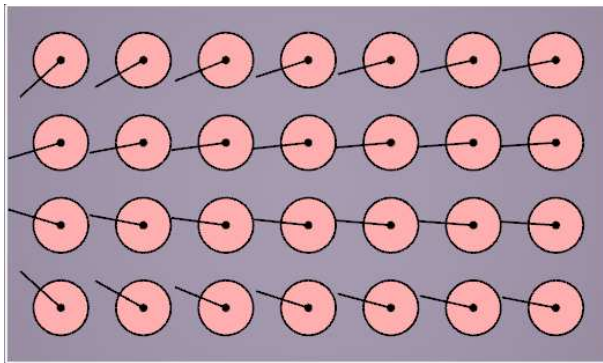


Figure 140. Occupant Targets oriented to the left

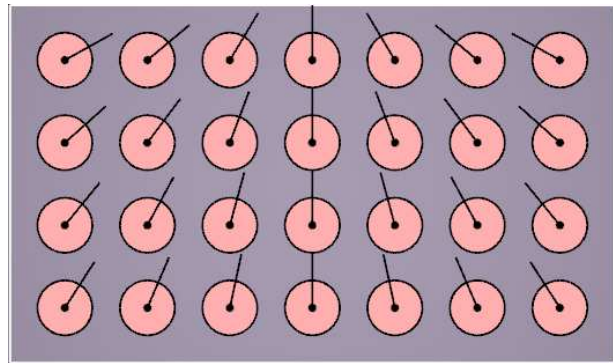


Figure 141. Occupant Targets oriented upward

If a more disorderly arrangement is preferred, Occupant Target orientations can also be randomized as follows:

1. Select the Occupant Targets to orient.
2. Right-click the selection and choose **Randomize Orientation**.

This can be repeated to generate different arrangements.

**NOTE**

Both the **Orient on Point** and **Randomize Orientation** actions can also be used for occupants to set their initial orientation.

## 9.4. Prioritizing Occupant Targets

Each Occupant Target has an associated priority that can optionally be used by an occupant's behavior to help the occupant determine which targets they prefer. The priority of an Occupant Target can be edited directly by selecting the target and editing the **Priority** property in the property panel.

The priorities of multiple Occupant Targets can also be generated based on a distance gradient from a reference point. For instance, with stadium seating this might be used to generate priorities so that occupants fill seats from the middle of a row outward. Alternately, seats might be prioritized such that those closest to a focus point have highest priority.

In order to generate priorities in this manner, perform the following:

1. Select the desired Occupant Targets.
2. Right-click the selection, and click **Prioritize Occ Targets on Distance**.
3. In the **Choose Reference Point** dialog, specify the reference location by either entering **X**, **Y**, and **Z** coordinates or click a reference point in the 2D/3D View.
4. In the **Choose Reference Point** dialog, enter the distribution parameters as described below.
5. Click **OK** to generate the priorities.

The following distribution parameters control how the priorities are generated for the selected targets:

### **Closest priority**

The priority value to assign to the Occupant Target closest to the reference point.

### **Farthest priority**

The priority value to assign to the Occupant Target farthest from the reference point. Targets between the closest the farthest targets are assigned a priority based on the value selected for **Bin Priority Levels**.

### **Bin Priority Levels**

Specifies whether to generate a discrete number of priority levels to assign to targets between the closest and farthest Occupant Targets. In general it is recommended to bin priority levels, especially in large models with many occupants and targets, as binning the priorities can help alleviate reservation conflicts and improve simulation performance. The following values can

be selected:

### Disabled

Binning is disabled. Each target will be assigned a unique priority value between **Closest Priority** and **Farthest Priority** based on the distances to the reference point of the closest and farthest targets and the target in question.

### By Distance

Priorities are binned based on their distance to the reference point, such that each bin spans a unique distance range. For instance, if the **Number of bins** is set to 10, and target distances range from [0-10], then all targets with distance from [0-1] will have the same priority, those with distances on the range [1-2] will have a different priority, etc. A side-effect of this approach is that some priority values might be assigned to more Occupant Targets than others, depending on the distribution of target distances.

### Evenly

Priorities are binned such that each generated priority level is assigned to the same number of Occupant Targets. A side-effect of this approach is that targets that are spatially far apart might be assigned the same priority.

### Number of bins

Specifies the number of discrete priority levels to generate if binning is enabled.

#### NOTE

For the **Prioritize Occ Targets on Distance**, all distances are Euclidean, and *not* travel distance.

## 9.4.1. Visualizing Occupant Target priorities

Occupant Target priorities can optionally be visualized using a colormap. To do so, on the **View** menu, select **Color Occupant Targets → By Priority**. The images below demonstrate coloring by priority with two different priority distributions. Dark red targets are high priority, and dark blue are low priority.

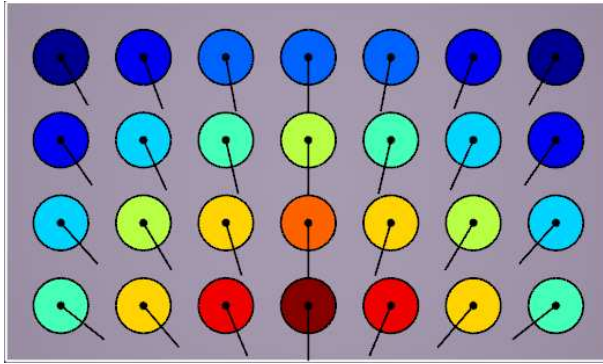


Figure 142. Occupant Targets colored by priority, prioritized toward the front

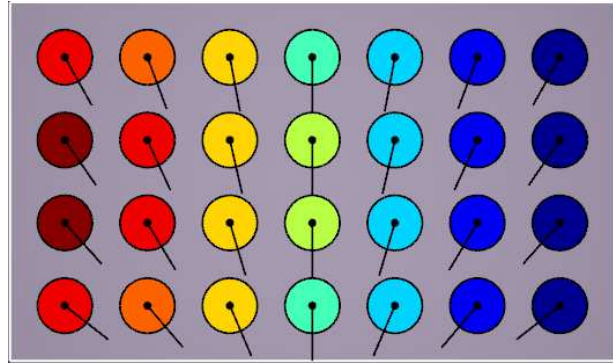


Figure 143. Occupant Targets colored by priority, prioritized toward the left

## 9.5. Using Occupant Targets

In order for an occupant to reserve and navigate toward an Occupant Target, their behavior must have a **Goto Occupant Targets** action, as described in [Section 5.3.3.5](#). When an occupant starts this action, they will reserve a target using the reservation system as described below. If they successfully reserve a target, which requires them to successfully plot a path toward a target and be chosen by the system, they will navigate toward the target.

### NOTE

Before an occupant has successfully reserved a target, the occupant will wait in their current room as described in [Section 5.3.3.9](#).

Once an occupant has reserved a target, the occupant will keep the reservation until the next **Abandon Occupant Targets** action in the behavior list. This prevents other occupants from reserving the target, even if the owning occupant leaves the target to perform a Trigger behavior. When using a Trigger, the occupant will only abandon their reserved target if the Trigger behavior contains an **Abandon Occupant Targets** action.

The **Goto Occupant Targets** action does not by itself cause an occupant to wait at the target. In order to do so, in the occupant's behavior list, add one of the wait actions after the **Goto Occupant Targets** action. The wait action will cause the occupant to wait at their most recently reserved Occupant Target. It also uses the reservation system and the previous **Goto Occupant Targets** action's properties to ensure the occupant maintains a reservation to one of the previously specified targets.

For instance, if an occupant is waiting at an occupant target and is then distracted by a Trigger whose behavior includes an **Abandon Occupant Targets** action, the occupant may no longer have a reserved target when they finish their Trigger behavior. The occupant will then use the reservation system again to reserve a new target from the set of targets chosen in the most recent **Goto Occupant Targets** action.



## 9.6. Occupant Target Reservation System

Whenever an occupant starts a **Goto Occupant Targets** action or is using one of the wait actions that follows a **Goto Occupant Targets** action, the occupant will use the Occupant Target reservation system.

The reservation system operates in multiple steps, possibly over the span of multiple simulation time steps. In general, the reservation system works as follows:

1. In the current timestep, an occupant makes one or more reservation requests to desired targets.
2. The reservation system resolves the requests made by all occupants by either accepting or rejecting reservations. If there are conflicts for a target, the system uses the conflicting occupants' priority preferences and travel distances to the target to determine who wins, preferring the occupant closest to the target who desires it most. Each occupant will only be allowed to keep a maximum of one request. Some occupants might not win any requests.
3. In the next timestep, an occupant will check to see if one of their requests has been accepted. If so, the occupant will navigate toward the reserved target or wait at it, depending on which behavior action is active and whether the occupant has reached the target. If the occupant did not win a reservation, they will repeat the above steps until they do. In the meantime, they will wait in their current room.

### 9.6.1. Choosing an Occupant Target

When choosing which Occupant Targets to request, an occupant will proceed as follows:

1. If the occupant has made a reservation in the past, they check if the most recent one is still reserved for them and is in their current set of allowed targets. If so, and they can still plot a path to this target, they will navigate to this target.
2. If this is not the case, but the occupant has active reservations that are in the current set of allowed targets, they will use their **Priority Preference** and **Distance Preference** to choose among these targets.
3. If the occupant cannot choose a previously-reserved target, they will then use their **Priority Preference** and **Distance Preference** to request a reservation from the allowed and available targets.

When picking from a set of targets, the occupant will use the preferences to further reduce the target set as follows:

1. The occupant starts with the **Priority Preference**. If it is set to **Higher** or **Lower**, the target set will be reduced to those with the same highest or lowest priority, respectively. Otherwise, the

target set is left as is.

2. The occupant uses their **Distance Preference** to pick from the remaining targets in the target set. If the **Distance Preference** is set to **None**, the occupant will pick at random. If the preference is instead set to **Nearest**, the occupant will pick the target nearest to them.

### 9.6.2. Ensuring Timely Resolves

In some models there may be many conflicting reservation requests. For instance, suppose a model is seeded with thousands of occupants who want to go to Occupant Targets. Also suppose there are thousands of Occupant Targets, each with a different priority (see [Section 9.4](#)) and that the occupants all desire the targets with highest priority. In this case, all occupants will immediately request the highest priority target. The reservation system can only pick one winner, while all others have to choose another target. In the next timestep, the occupants who did not win will choose the next highest priority target, repeating until all occupants have a reservation or until there are no more targets left.

If occupants were only allowed to request one target in each timestep, it could take thousands of timesteps to resolve the conflicts. To alleviate this problem, if an occupant sees that there were conflicts in the previous timestep for one of their requested targets, they may make several requests in the current timestep. The number of requests they choose in each time step depends on the number of conflicts in the previous timestep as well as the **Occupant Target Resolve Time** simulation parameter, as discussed in [Section 15.1.5](#). Using this parameter, occupants are guaranteed to reserve all available occupant targets within this time limit plus two additional timesteps from the time they start reserving targets.

For instance, suppose **Occupant Target Resolve Time** is set to the default time of **5 s**. If the occupants start a **Goto Occupant Targets** action at **t=0**, either all targets should be reserved or all occupants should have a reservation by **t=5.05 s**, using the default timestep of **.025 s**.

#### NOTE

**Occupant Target Resolve Time** can be set to **0**, which will guarantee that occupants will reserve their targets in two timesteps, but this may negatively impact simulation performance, as occupants will need to calculate a travel distance to all targets, even though they will only choose one target. Using larger values will limit the number of travel distance calculations that need to be performed per occupant.

## 9.7. Occupant Target Limitations

There are currently some limitations for occupants when using Occupant Targets:

- Occupant Targets are currently unsupported for occupants in movement groups. If an occupant is in a movement group when they start their **Goto Occupant Targets** action, they

will skip the action and immediately begin their next behavior action.

- Including Occupant Targets in tight seating rows for ingress scenarios may lead to excessive congestion in the seating row, as occupants cannot yet be modeled as sitting, which means other occupants may need to try to go around them to get to their seats. A possible workaround would be to follow the **Goto Occupant Targets** action with a **Change Profile** or **Change Profile Property** action that either reduces the occupant's size or their **Reduction Factor**. Another workaround would be to adjust the seating area if possible to give the occupants more room for navigation.

## 9.8. Occupant Target Output

See [Chapter 16](#) for more information on the output available for Occupant Targets.

# Chapter 10. Triggers

Pathfinder **Triggers** are objects that can interrupt an occupant's current behavior and replace it with another behavior, either temporarily or permanently. Triggers can be global or they can have some area of influence, such as a circular area or an entire room. In either case, they can be stationary or can move with an occupant. They can also be placed manually before running the simulation or can be created during the simulation by an occupant as part of their behavior. Occupants can be forced to use triggers or can use them based on a probability distribution.

The following are some common use-cases for triggers:

- A trigger represents a sign that might distract an occupant, such as for a bathroom or food vendor.
- A trigger represents a global alarm that informs occupants to evacuate.
- A trigger moves with a first responder making announcements to evacuate. The trigger behavior (the one that occupants will use when interrupted by the trigger) optionally creates other triggers with the evacuation announcement, causing the message to spread.

Once an occupant uses a trigger, they will not use it again for the duration of the simulation.

## NOTE


Triggers are different from the [Change Behavior](#) action in that triggers interrupt an occupant's current behavior action, whereas the **Change Behavior** action will change an occupant's behavior as part of a sequence of events - the behavior only changes after completing the previous behavior action.

## 10.1. Creating Triggers

As mentioned previously, triggers can either be placed manually prior to starting the simulation or they can be created by an occupant as part of their behavior.

### 10.1.1. Placing Triggers Manually

Placing triggers manually can be helpful for creating global alarms or stationary distractions, such as signs. To add a trigger in this way, perform the following steps:

1. In the 2D or 3D View, click the **Add a trigger** drawing tool .
2. Enter the desired **Awareness** and **Behavior** for the trigger.
3. Enter any other desired properties for the trigger.
4. If the **Awareness** is set to **Line of Sight** or **Same Room**, or the **Behavior** is set to **Wait at Trigger**, enter a location for the trigger. This can be done by either clicking a location in the 2D/3D

View or by typing in a **Location** in the property panel. If the **Awareness** is set to **Line of Sight**, click+drag the left mouse button to set a location and control the **Awareness Radius**.

5. If the trigger location was not set in the 2D/3D view, click **Create** in the property panel to create the trigger manually.

In the **Navigation View**, the created triggers can be found under the **Triggers→Placed Triggers** group.

### 10.1.2. Creating Triggers at Run-time from Trigger Templates

In order for a trigger to be created during the simulation, a **Trigger Template** will first need to be defined prior to running the simulation. Then, an occupant can create a **Trigger Instance** from a **Trigger Template** as part of their behavior with the **Create Trigger** action. Trigger templates share the same **Trigger Properties** as placed triggers except for location. They do not influence occupant behavior until created by a behavior action.

To create a trigger template, perform the following steps:

1. Create a new trigger template using one of these methods:
  - On the **Model** menu, choose **New Trigger Template**.
  - Right-click **Trigger Templates** in the **Navigation View**, and choose **New Trigger Template**.
2. Enter the desired **Awareness** and **Behavior** for the trigger template.
3. Enter any other desired properties for the trigger template.

In the **Navigation View**, trigger templates are found under the **Triggers→Trigger Templates** group.

### 10.1.3. Converting Placed Triggers into Trigger Templates

Trigger Templates can also be created using the properties of an existing placed trigger.

To copy a placed trigger's properties into a trigger template, perform the following steps:

1. Select all the placed triggers that you want to create templates from.
2. Right-click the selection, and from the shortcut menu, click **Copy to Trigger Template**.

A new trigger template will be created based on each trigger selected. After creation, these templates are disconnected from the original triggers and the properties of both can be edited separately.

## 10.2. Trigger Behavior

Triggers can change an occupant's behavior to a simple built-in behavior that waits at the trigger for a limited time, or it can change the behavior to any other defined in the model. The occupant's behavior can be changed permanently if the trigger behavior ends with a terminal action, such as [Goto Exits](#). Alternatively, the trigger can temporarily interrupt the occupant's current action if the trigger behavior ends with a [Resume Prior Behavior](#) action. In this mode, the behavior action that was interrupted by the temporary trigger behavior will be resumed when the trigger behavior is completed. For instance, if the occupant is using a [Goto Rooms](#) action and is interrupted by a trigger before they reach one of the rooms, the occupant will perform the trigger behavior and then resume the **Goto Rooms** action when the trigger behavior is complete.

### 10.2.1. Wait at Trigger Behavior

To use the built-in behavior that waits at the trigger, perform the following:

1. Select the trigger.
2. In the **Property Panel**, for the **Behavior**, select **<Wait at Trigger>**.
3. In the **Property Panel**, enter the desired **Wait Area Radius** and **Wait Time**. These define the waiting area surrounding the trigger's location and how long the occupant will wait once they reach this area.

#### NOTE

In some cases, you may want the occupant to go to their current trigger, but you may want more control over how they track it or what happens once they reach the trigger. In this case, define a custom behavior as described below, and add a [Goto Current Trigger](#) action.

### 10.2.2. Custom Trigger Behavior

To define a custom behavior for the trigger, create the behavior as defined in [Behaviors](#) and select it as the trigger's **Behavior**.

Because a trigger can only define one behavior, all occupants who choose to use the trigger will use the same behavior. If you need the occupants to use different behaviors, however, you can do one of the following:

- Define the trigger behavior such that it only contains a single [Change Behavior Action](#). The **Change Behavior Action** then defines the available behaviors. This method chooses a Behavior at random. If you need more control over the behavior chosen, then use the following method.
- Define separate triggers at the same location but with different behaviors. This gives more

control over the selection of the behavior, as you can control which occupants are affected by each triggers.

### 10.2.3. Interrupted Waiting

The **Wait** and **Wait Until** actions operate in a specific way when interrupted by a temporary trigger. When either of these actions is started by an occupant, an ending time is calculated and never changes, even if interrupted. For instance, if an occupant starts a **Wait** action at  $t=63$  s, and they decide to wait for  $30$  s, the calculated end time is  $t=93$  s. If the occupant is interrupted by a temporary trigger while waiting, they will complete the trigger behavior and then resume the wait action. When they resume the wait action, however, the end wait time remains unchanged at  $t=93$  s. If the current time is already past  $t=93$  s, the occupant will immediately begin their next behavior action. Otherwise, they will wait until  $t=93$  s before beginning the next action as usual.

In addition, whenever an occupant is interrupted by a temporary trigger while waiting, the occupant will return to their previous waiting area when they resume the wait action. For instance, if an occupant is directed to go to a room using a **Goto Rooms** action and then told to wait for some time, their waiting area will be the entire room they reached before waiting. If they are then interrupted by a temporary trigger while waiting, they will return to the waiting room when done with the trigger behavior. Similarly, if they had gone to a waypoint instead, they would return to the waypoint. When the occupant resumes the wait action, the time it takes to get back to the waiting area counts against the wait time. So if the wait time elapses while they are on their way back to the waiting area, the occupant will stop travelling to the wait area and begin their next behavior action.

## 10.3. Occupant Response to a Trigger

There are multiple mechanisms that interact with each other in order for an occupant to become aware of a trigger and then ultimately decide to use it (i.e. interrupt their behavior with the trigger's behavior). This occurs in several stages as detailed below:

1. The occupant filters the triggers in the simulation according to filtering criteria.
2. The occupant tests their awareness of the filtered triggers.
3. When the occupant becomes aware of a trigger, they will schedule a time to decide whether to use the trigger.
4. At the appropriate decision time, the occupant will decide whether to use the trigger.
5. The occupant uses the trigger.

### 10.3.1. Trigger Filtering

Before an occupant can become aware of a trigger, they must first recognize that the trigger exists. This is controlled in two ways:

- In the occupant profile, the occupant's **Allowed Triggers** property defines which triggers can be considered (see [Profile Movement](#)).
- In the trigger itself, the **Allowed Occupants** property separately defines which occupants can use the trigger (see [Section 10.6](#)).

In order for an occupant to consider an trigger, both of the above properties must hold. A trigger must be in the **Allowed Triggers** for the occupant, *and* the occupant must pass the trigger's **Allowed Occupants** test.

### 10.3.2. Trigger Awareness

For each trigger that passes the trigger filtering test as outlined above, the occupant must become *aware* of the trigger before they will decide to use it. This is determined by both the trigger's **Awareness** and **Awareness Requirements** properties. There are several awareness tests and requirements that can be applied to a trigger as outlined in [Section 10.6](#).

### 10.3.3. Trigger Decision Scheduling

Once the occupant becomes aware of a trigger, they will schedule a time at which they will *decide* whether to use the trigger. The trigger's **Decision Time** property defines when the occupant will make a decision. The decision time can be a delay after the occupant becomes aware, can be scheduled at a specific time, or can be calculated automatically (see [Section 10.6](#)).

If a trigger's **Decision Time** is set to **Automatic**, the time is calculated based on the occupant's current movement state.

- If the occupant is **seeking**, they will decide immediately.
- If the occupant is **idling** (e.g. using a wait action), the **Decision Time** will be a random time during their wait period. If they do not know how long they will be waiting, such as when using the [Wait Until Simulation End](#) action, they will use the **Default Trigger Idle Time** property as their wait period. This property is defined in the **Simulation Properties** dialog under the [Miscellaneous Parameters](#) tab.

### 10.3.4. Trigger Decision

Once the **Decision Time** has elapsed, the occupant will decide whether to use the trigger. If they are still within the **Awareness** region or the trigger's **Remain aware** property is checked, the occupant will calculate a probability of using the trigger (they will ignore the trigger, otherwise).



- If the trigger property, **Ignore Occupant Susceptibility**, is **checked**, the probability is exactly the trigger's **Influence**.
- If **Ignore Occupant Susceptibility** is **unchecked**, the probability is the product of the trigger's **Influence** and the occupant's **Trigger Susceptibility** (see [Section 10.6](#) and [Profile Movement](#)).

A probability of **0%** means that the occupant is guaranteed to not use the trigger, whereas a probability of **100%** or greater means that the occupant is guaranteed to use the trigger. Values between are used as follows:

- A random number will be generated for the occupant, such that  $0 \leq \text{random\_number} < 100$ .
- If the random number is less than the probability value, the occupant will use the trigger.

The effect of this can be thought of in two ways:

- With a sufficiently large number of occupants who make a decision about the trigger, and assuming they all calculate the same probability, **P**, approximately **P%** of the occupants will use the trigger.
- If a single occupant becomes aware of the trigger a sufficiently large number of times throughout the course of the simulation, they will use it approximately **P%** of the occurrences.

#### NOTE

Unless the **Decision Time** is set to a **Delay** of **0**, either the occupant's properties or the trigger's influence can change between the time that the occupant becomes aware of the trigger and that the decision time elapses. This can affect the probability of the occupant using the trigger. In addition, there are certain cases where an occupant might not use a trigger. (see [Section 10.7](#)).

## 10.4. Trigger Memory

Once an occupant uses a trigger, they will not use it again for the duration of the simulation. Note that this only applies to a placed trigger or a trigger instance (one created by an occupant from a trigger template). An occupant can still use multiple trigger instances created from a single trigger template (see [Section 10.1](#)).

In addition, if an occupant decides not to use a trigger, the occupant will remember this decision. They will only reconsider the trigger in the future if at least one of the following occurs:

- The probability of using the trigger changes.
- The trigger's **Remain aware** property is **checked** and the occupant leaves the trigger's **Awareness** region.
- The trigger is a **Global** trigger (i.e. its **Awareness** is set to **Global**), the occupant finishes a

behavior action, and the occupant is not currently using any trigger. This allows occupants to periodically consider global triggers, even if they previously decided not to use one.

## 10.5. Trigger Rank

Each trigger has a **Rank**, which is simply an integer value that serves two purposes:

- Prioritizes triggers when an occupant has multiple triggers that it chooses to use at the same time. The occupant can only use one trigger, so the occupant uses the trigger with the highest **Rank**. If there are multiple with the same highest rank, it will choose one of them at random. The unused triggers will be treated as if they were not chosen, and the [Trigger Memory](#) applies.
- Allows a trigger with higher rank to interrupt a lower-rank trigger. For instance, if an occupant is using **TriggerA** with rank 0, and the occupant becomes aware of and decides to use **TriggerB** with rank 1, then **TriggerA's** behavior will be interrupted by **TriggerB's** behavior. If **TriggerB's** behavior ends with a [Resume Prior Behavior](#) action, and **TriggerA's** **Resume if interrupted** property is **checked**, then **TriggerA's** behavior will resume. Otherwise, the remainder of **TriggerA's** behavior will be skipped, and the occupant's original behavior will be resumed instead.

## 10.6. Trigger Properties

The following properties can be set for a trigger in the property panel, either during creation or when a trigger is selected:

### Rank

Used to compare with other triggers if an occupant has multiple choices available or has already started a trigger. Triggers with higher ranks are given priority over those with lower ranks and can interrupt a lower-rank trigger's behavior (see [Trigger Rank](#)).

### Behavior

Defines the behavior that the occupant will use if they decide to use the trigger. The special value, **<Wait at Trigger>**, allows for a simple definition of a behavior, where an occupant will seek to a trigger's location and then wait there for a specified amount of time. If the trigger is moving, an occupant will seek to it once and then wait at that location. The trigger may move away during this wait. If the trigger is unreachable, the occupant will wait until it becomes reachable again. When using this value, you do not have to create a separate behavior (see [Trigger Behavior](#)).

### Resume if interrupted

If the trigger is interrupted by another higher rank trigger with a [Resume Prior Behavior](#)

action, this flag determines what happens when this trigger's behavior is resumed. If checked, this trigger's behavior resumes and is completed normally. If unchecked, the remainder of this trigger's behavior is skipped, replaced immediately with a Resume Prior Behavior action (see [Trigger Rank](#)).

### Wait Area Radius

When the behavior is set to **<Wait at Trigger>**, **Wait Area Radius** defines the waiting area, separate from the awareness area (defined by **Awareness Radius**). This is the same as setting an **Arrival Radius** when specifying a [Goto Waypoint](#) action (see [Trigger Behavior](#)).

### Wait Time

When the behavior is set to **<Wait at Trigger>**, this defines the amount of time an occupant will wait at the trigger before resuming their behavior prior to using the trigger (see [Section 10.2](#)).

### Allowed Occupants

Defines which occupants can use this trigger based on their tags. See [Section 10.3.1](#) for more information on how filtering is used. See [Section 5.8](#) for more information on how tags are applied to occupants. The following values can be specified:

#### Accept All

Any occupant can use this trigger.

#### Reject All

No occupants can use this trigger.

#### From List

An occupant can only use this trigger if they meet the requirements from the list of tags. When using this option, the list can be specified in one of the following ways:

#### Reject

An occupant which has any tags in the list cannot use the trigger.

#### Accept

An occupant must have a tag in the list to use the trigger.

### Decision Time

Specifies the time when an occupant considers the trigger after becoming aware of it (see [Trigger Decision Scheduling](#)). This can be one of the following values:

#### Automatic

The occupant chooses a time to consider the trigger based on their current movement state. If the occupant is currently idling, they consider the trigger at a random time in the future

before the end of their current idle period. If the occupant is currently seeking, they consider the trigger immediately.

**Delay**

The occupant considers the trigger after a specified delay. This delay can be chosen from a continuous or discrete distribution.

**Specific Time**

The occupant considers the trigger at a specified time. This time can be specified as a single time, scheduled list of times, or periodic function of times. These options work similarly to those in the [Wait Until](#) action.

**Remain aware**

Allows the occupant to consider the trigger if they become aware of it and then leave the awareness region before the **Decision Time**. If checked, the occupant will consider the trigger when it's time even if they're no longer in the awareness region. If unchecked, the occupant ignores the trigger if they are no longer in the awareness region (see [Trigger Decision](#)).

**Awareness**

Defines how an occupant becomes aware of the trigger (see [Trigger Awareness](#)). This can be one of the following values:

**Line of Sight**

The occupant must be within the awareness radius of the trigger's location and be able to trace a straight path on the navigation mesh to the trigger location without hitting any room boundaries.

**Awareness Radius**

Defines the area around the trigger's location where an occupant can become aware of the trigger when using the **Line of Sight** awareness option.

**Same Room**

The occupant must be located in the same room as the trigger's location.

**Specified Rooms**

The occupant must be in one of the rooms listed in the **Rooms** property.

**Global**

The occupant always knows about the trigger from anywhere in the model.

**NOTE**

This **Awareness** value may lead to excessive use of the trigger, as occupants clear their decisions about global triggers whenever they start a new behavior action. (see [Trigger Memory](#)).

**Awareness Requirements**

Defines additional requirements that must all be met before an occupant starts the **Decision Time** timer, including the following:

**Min. Awareness Count**

The minimum number of times that an occupant must become aware of the trigger or its template.

**Min. Awareness Time**

The minimum amount of time an occupant must be aware of the trigger or its template. This is similar to the **Min. Awareness Count**, but the occupant accumulates a time counter while it's aware of the trigger.

**NOTE**

Each occupant maintains a count and time accumulator for both a trigger instance and its template. This means that the criteria can be met by either. For example, if the **Min. Awareness Count** is set to **2**, and the occupant sees one trigger instance from a template and then sees another instance from the same template, the count criteria has been met. When the occupant decides to use one of the trigger instances, the occupant will restart the template accumulators so that they must reach the awareness count/time again before using another instance.

**Rooms**

The set of rooms that define where an occupant becomes aware of a trigger when using the **Specified Rooms** awareness option.

**Influence**

Helps define the probability over time that an occupant will decide to use a trigger. If **Ignore Occupant Susceptibility** is **unchecked**, this value is multiplied by the occupant's susceptibility to triggers to define the probability. See the [Profile Movement](#) tab for more information about defining the occupant's susceptibility. While this value is presented as a percentage, the value can actually be greater than 100% to give a strong preference for the trigger. Likewise, it can be set to 0% to effectively disable the trigger (see [Section 10.3.4](#)).

**Influence Timeline**

Defines the timeline on which the trigger's **Influence** values are based.

## Trigger

The trigger's influence values start when it is created.

## Global

The trigger's influence values start at the beginning of the simulation.

## Ignore Occupant Susceptibility

If **checked**, indicates that an occupant's **Trigger Susceptibility** should be ignored when calculating the probability that an occupant will use the trigger. In this case, the trigger **Influence** alone controls the probability. This may be useful, for instance, if you want to ensure that occupants will use the trigger, such as a trigger for a global alarm, as long as the trigger is in the occupant's **Allowed Triggers** list. If **unchecked**, the trigger **Influence** is multiplied by the occupant's **Trigger Susceptibility** to determine the probability of using the trigger. See [Profile Movement](#) and [Trigger Decision](#) for more information.

### NOTE

Even if **Ignore Occupant Susceptibility** is checked, an occupant may still ignore the trigger if it is not in their **Allowed Triggers** list.

## 10.7. Trigger Limitations

While triggers work with most behaviors and for most occupants, there are some limitations, including the following:

- Presently, occupants in [Movement Groups](#) will not use triggers.
- Occupants performing the following behavior actions can only be interrupted by triggers in specific cases:
  - [Assist Occupants](#) - assistants will not be interrupted at any time while performing the assistance behavior action, even if they are not actively assisting a client.
  - [Wait For Assistance](#) - the client occupant cannot be interrupted while waiting for assistance, but once they are being assisted, they can be interrupted by triggers.
  - [Goto Queue](#) - occupants can be interrupted as they move toward a queue or are actively waiting in line. They cannot, however, be interrupted while at a service point. If interrupted while in line, the occupant will lose their place in line, allowing other occupants behind them to move forward. When they resume the action, they effectively start the action over, possibly choosing a new line and going to the back of it. In addition, actual trigger usage of occupants waiting in a queue might not match the expected probability of usage, as an occupant might not be able to accurately predict how much time they will spend in a queue.
  - [Goto Elevators](#) - Occupants *can* be interrupted while going to the elevator, but once they

are inside the elevator, they cannot be interrupted until they reach their destination.


- Any action that completes immediately after starting, such as [Change Behavior](#), will not be interrupted.

## 10.8. Trigger Output

See [Chapter 16](#) for more information on the output available for triggers.

# Chapter 11. Views

At any time, the state of the perspective camera can be saved, including its position, orientation, and zoom. This information is stored in an object called a **View**. A view can be recalled later in either Pathfinder or the 3D Results to view the scene from that perspective.

Views appear in Pathfinder as points and can be hidden either individually or as a whole. To show/hide all views, click the **Show View Objects** button  in the 3D/2D view toolbar.

## 11.1. Creating a View

To create a view, follow these steps:

1. Position the perspective camera using one of the navigation tools as discussed in [Section 2.2.1](#).
2. Create a new view using one of these methods:
  - On the **Model** menu, choose **New View**.
  - Right-click **Views** in the **Navigation View**, and choose **New View**.

A new view will appear in the Navigation View as shown in [Figure 144](#).

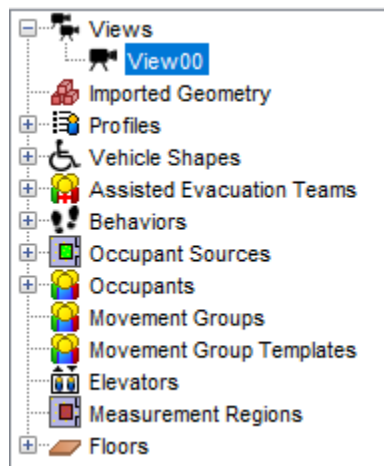


Figure 144. Creating a View

## 11.2. Recalling a View

To recall a view, perform any of the following:

- Double-click the desired view from the Navigation View.
- Right-click a view from either the Navigation View or the 3D/2D View, and choose **Show in 3D View**.



This will show the 3D View and the perspective camera will be initialized with the state of the saved view.

## 11.3. Editing a View

If the orientation of a view needs to be changed, perform the following:

1. Position the perspective camera into the new desired position.
2. Right-click the view and choose **Update View**.

Alternatively, the graphical representation of the camera can be manipulated to move the location of the view as discussed in [Section 12.2](#).

## 11.4. Views in the Results

In order for the views created in Pathfinder to be available in the Results, a special views file is written to the output folder whenever the simulation is run. When the Results application is opened, it then reads the views from this file and makes them available in the Results.

Because this file is only automatically written when the simulation is run, however, a view can be changed in Pathfinder after running the simulation, making it out of sync with the views in the Results. In order to synchronize the views in Pathfinder and the Results without re-running the simulation, the views file must be manually written. To do so, follow these steps:

1. Make the desired changes to the view in Pathfinder.
2. On the **File** menu, choose **Save Views File** from the **Export** submenu.
3. Save over the existing views file.
4. Re-load the results in the 3D Results Viewer.

## 11.5. Camera Tours

In previous versions of Pathfinder, tours could also be created inside Pathfinder. This functionality was later moved to the 3D Results. Tours made in previous versions of Pathfinder, however, will still appear in the **Navigation View** of Pathfinder. While they can no longer be edited within Pathfinder, they can still be deleted. Ones that are not deleted will be written to the Results views file so that they are available in the Results.


# Chapter 12. Editing and Copying Objects

Most objects can be edited in two ways in Pathfinder. One way is to **transform** the object, including rotating, translating (moving), and mirroring it. Another way is to graphically **manipulate** the objects by dragging **handles**. Objects can also be copied either with copy/paste or through the transform tools as discussed in the following section.

## 12.1. Transforming and Copying Objects

All geometric objects can be transformed and/or copied. All transform/copy options are available through tools in the 3D and 2D views. The following transforms are available and are discussed in the following sections: moving, rotating, and mirroring.

### 12.1.1. Moving

To move one or more objects, select the objects and click the move tool  from the 2D or 3D view. The property panel for the move tool is shown in [Figure 145](#).

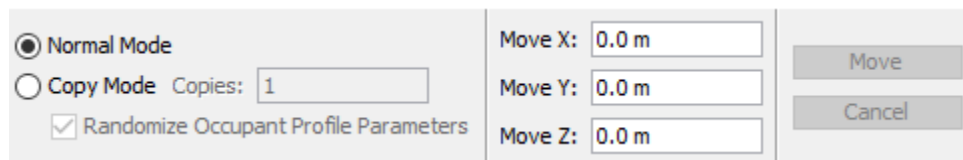


Figure 145. Property panel for the translate tool

The object can be moved either manually or graphically:

#### Manually

Select **Normal Mode** and enter the distance to offset the object in the Move X, Y, and Z boxes. Then click **Move**.

#### Graphically

This is performed most easily in one of the 2D views. To translate graphically click two points on the model. The vector from the first point to the second defines the movement offset. When moving graphically, objects will only be moved parallel to the camera's view plane.

Objects can also be copied using the move tool. To do so, select the move tool, select **Copy Mode** from the property panel, and follow the same steps as above for moving an object. Alternatively, hold **CTRL** on the keyboard while defining the offset. This will create a copy of the object that has been offset by the move distance.

Similarly, an array of objects can be made by specifying a value greater than 1 for the **Copies** field in the property panel. The array is created by offsetting each previous copy by the move distance.

If, when copying rooms, the resulting copies overlap one another the most recent copies take precedence over earlier ones, meaning that earlier ones will have area subtracted from them.

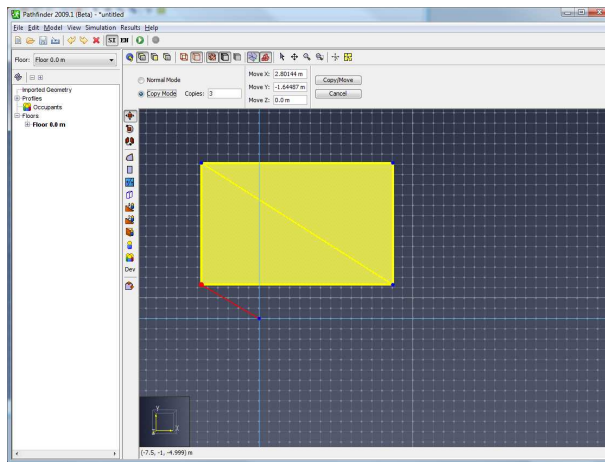


Figure 146. Select object and click first point

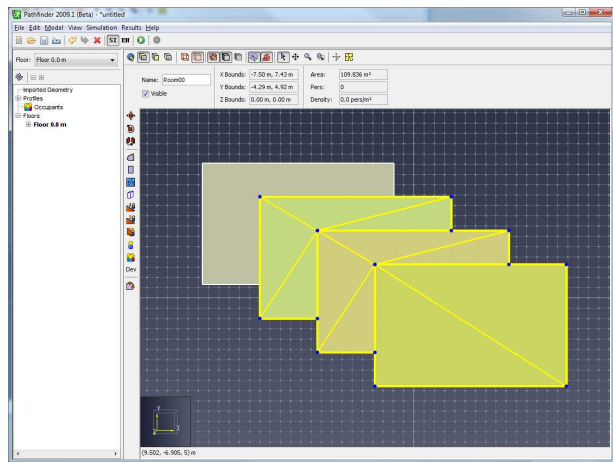



Figure 147. Second click creates array

### 12.1.2. Rotating

To rotate one or more objects, select the objects and click the rotate tool  from the 2D or 3D view. The property panel for the rotate tool is shown in [Figure 148](#).

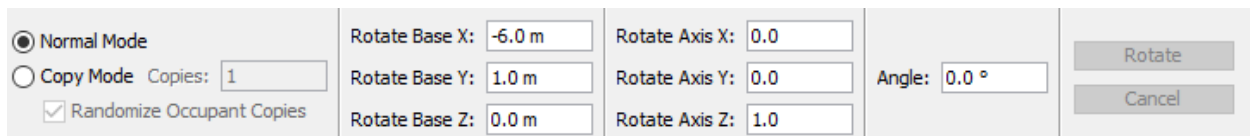


Figure 148. Property panel for the rotate tool

The object can be rotated either manually or graphically:

#### Manually

Select **Normal Mode** and enter the base of rotation, the axis about which to rotate using the right-hand rule, and the angle to rotate. Then click **Rotate**.

#### Graphically

This is performed most easily in one of the 2D views. The rotate axis is automatically set to a vector normal to the camera. Rotating requires three mouse clicks.

- The first specifies the base of rotation ([Figure 149](#)).
- The second defines a reference vector extending from the rotation base ([Figure 150](#)).
- The third defines a second vector extending from the rotation base ([Figure 151](#)). The rotation angle is the angle between these two vectors.

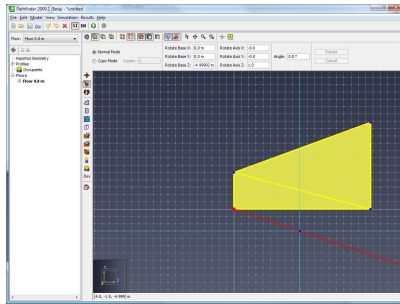


Figure 149. Base point of rotation

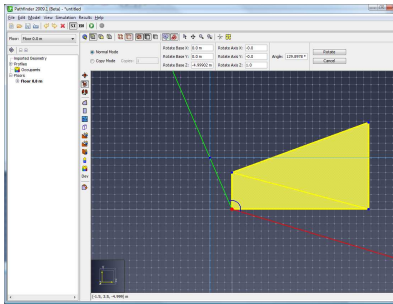


Figure 150. Reference line (red) and rotation angle (green)

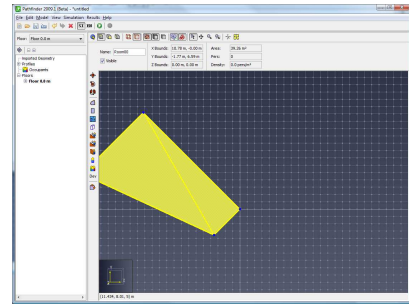


Figure 151. Rotated object

Objects can also be copied using the rotate tool.

1. Select the rotate tool
2. Select **Copy Mode** from the property panel, and follow the same steps as above for rotating an object. Alternatively, hold **CTRL** on the keyboard while defining the rotate properties.

This will create a copy of the object that has been rotated from the original using the rotate parameters. Similarly, an array of objects can be made by specifying a value greater than 1 for the **Copies** field in the property panel. The array is created by rotating each previous copy by the rotate angle.

If copying rooms and resulting copies overlap one another the most recent copies take precedence over earlier ones, meaning that earlier ones will have area subtracted from them. An array is shown in [Figure 153](#).

### Creating an array of objects using the rotate tool

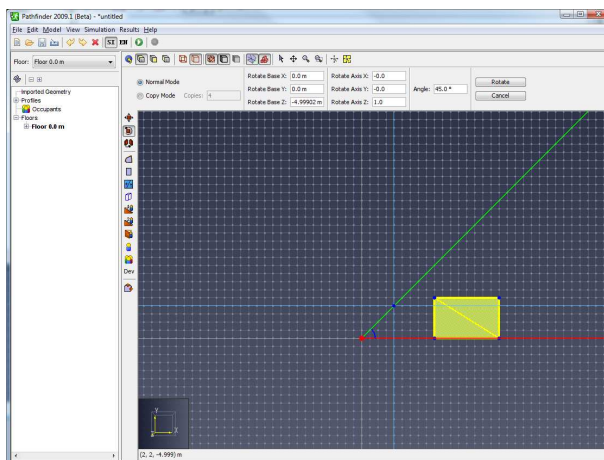


Figure 152. Select object and click and second line

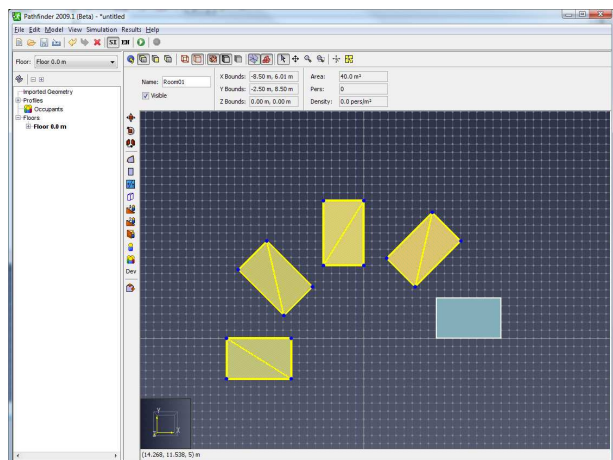



Figure 153. Create array from rotation

### 12.1.3. Mirroring

To mirror one or more objects about a plane, select the objects and click the mirror tool  from the 2D or 3D view. The property panel for the mirror tool is shown in [Figure 154](#).

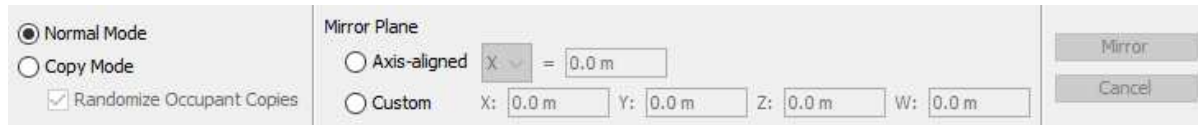


Figure 154. Property panel for the mirror tool

The object can be mirrored either manually or graphically:

#### Manually

Select **Normal Mode** and enter the plane about which to mirror. This can be an axis-aligned plane or a custom plane specified by the plane equation, [stem b8e8825b9f04f4ea473c4e901f2552c5]. Next click **Mirror**.

#### Graphically

This is performed most easily in one of the 2D views. The mirror plane is always perpendicular to the camera's view plane. Defining the plane requires two mouse clicks that define two points in the plane. The steps for mirroring graphically are shown in [Figure 156](#).

#### Mirroring an object

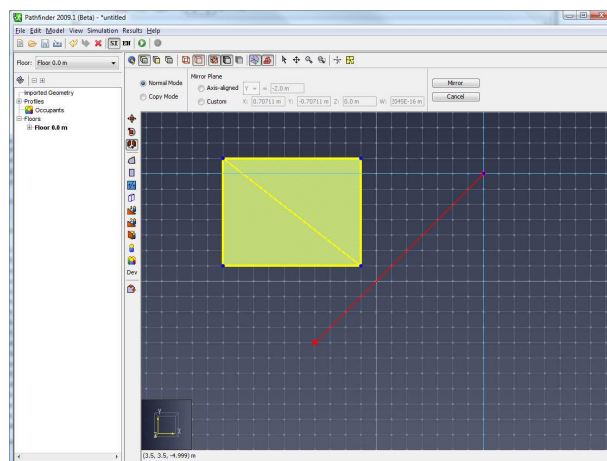


Figure 155. Select object and draw mirror line

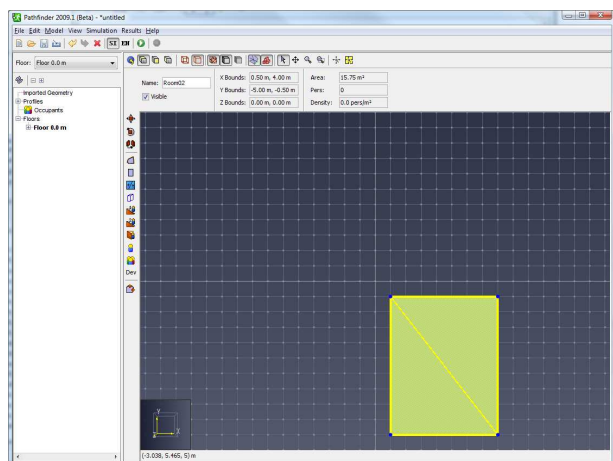


Figure 156. Selected objects mirrored over

Objects can also be *copied* using the mirror tool. To do so, select the mirror tool, select **Copy Mode** from the property panel, and follow the same steps as above for mirroring an object.

Alternatively, hold **CTRL** on the keyboard while defining the mirror plane. This will create a copy of the object that has been mirrored from the original using the mirror plane.

## 12.2. Manipulating Objects with Handles

Some objects, including occupants, rooms, stairs, and doors, can be edited through manipulator **handles**. **Handles** act as points on an object that can either be dragged with the selection tool or edited by the keyboard to edit the attached object. Handles only appear if a single object is selected and show as blue dots as shown in [Figure 157](#).

### NOTE

Imported objects may be very complex with thousands or even millions of potential handles. By default, Pathfinder limits the number of displayed handles to improve application responsiveness. The maximum value can be changed under **File** → **Preferences** → **Maximum manipulation handles**. Lower values will increase responsiveness but may limit the extent to which the object can be manipulated. Enter **max** to disable the limit.

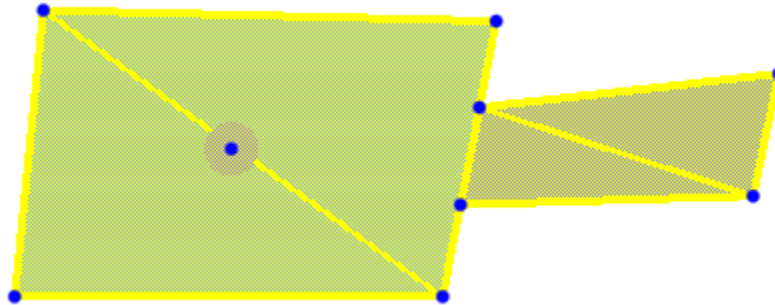



Figure 157. Manipulator handles

### 12.2.1. Selecting and Deselecting a Handle

To select an object's handle, the object itself must first be selected. Once it is selected, the blue handles should appear. Next select the **Select/Edit** tool . Now an individual handle can be selected by clicking it, which will make the handle property panel appear as shown in [Figure 158](#). To deselect the handle, press *ESC* on the keyboard, click anywhere else in the model, or select another object.

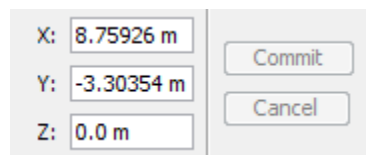


Figure 158. Handle property panel


### 12.2.2. Editing a Handle

A handle can be edited in one of two ways: it can be edited with the keyboard to enter precise values or it can be edited graphically.

## Editing with the keyboard

To edit with the keyboard, a handle must first be selected. Next enter the desired location in the X, Y, and Z fields in the property panel and select the **Commit** button. The handle will attempt to modify the underlying object using the handle's internal constraints that are described in the appropriate section of this guide for the respective handle's object.

## Editing graphically

A handle does not have to be selected before editing graphically. To edit graphically, make sure the **Select/Edit** tool  is selected, and press the left mouse button over the desired handle and drag the handle to the desired location. Release the left mouse button, and the object will be edited. A real-time preview of the object being edited will be shown as the mouse is dragged.

### 12.2.3. Room Handles

When rooms are selected, a handle can be found at every vertex on the boundary of the room. The handles move the underlying vertex to reshape the room. The handles can be moved to any location within the plane of the face the vertex is on. If the vertex is shared between two faces in non-parallel planes, the handle can only be moved along the edge to which it is attached.

### 12.2.4. Thin Door Handles

When a thin door is selected, three handles will be displayed as shown in [Figure 159](#). The handles on the ends of the door allow the door to be moved along the edge to which it is attached. The middle handle allows the door to be made thick by moving the handle to the edge of another room as shown in [Figure 160](#).

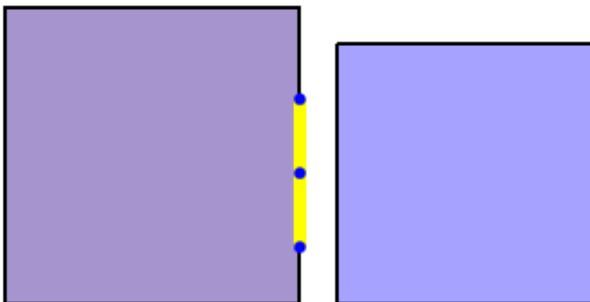


Figure 159. Thin door with three handles

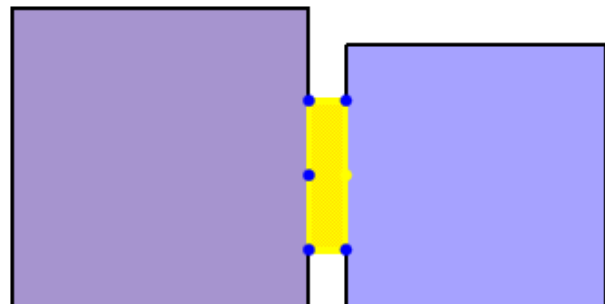


Figure 160. Thick door with six handles

### 12.2.5. Thick Door Handles

When a thick door is selected, six handles will be displayed as shown in [Figure 160](#). The four handles on the corners of the door allow the door to be moved along the edge to which each handle is attached. Each middle handle allows the door to be made into a thin door by dragging to

the other middle handle. The middle handle can also be useful to reattach the door to a room if the door somehow became detached (such as by a modification to the room).

### 12.2.6. Stair and Ramp Handles

When a stair or ramp is selected, six handles will be displayed as shown in [Figure 161](#). The four corner handles each allow the stair/ramp to be moved along the edge to which the handle is attached. The middle handles allow the stair/ramp to be reconnected to another room. The middle handles can also be useful if the geometry of one room attached to a stair/ramp has changed in such a way that the stair/ramp is no longer attached to the room. The middle handle can be used in this case to reconnect to the room.

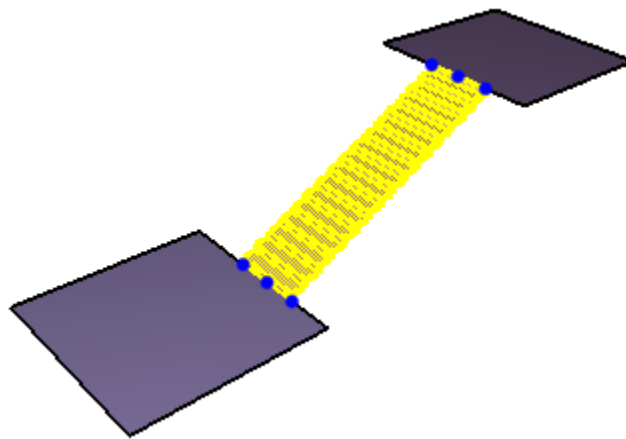


Figure 161. Stair/ramp handles

### 12.2.7. Occupant handle

When an occupant is selected, there is only one handle as shown in [Figure 162](#). The sole purpose of this handle is to move the agent to another location. Moving an agent in this manner has a benefit over the translation tool in that the location automatically snaps to an existing room or stair as when adding an agent using the occupant dropper tool.

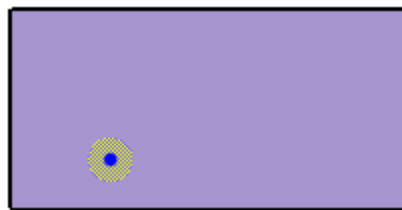


Figure 162. Occupant handle

### 12.2.8. Waypoint Handles

When a waypoint is selected, two handles are displayed as shown in [Figure 163](#). The center



handle allows the waypoint to be moved to another location similarly to the occupant handle. The perimeter of the circle can be used as a handle to change the arrival radius of the waypoint.

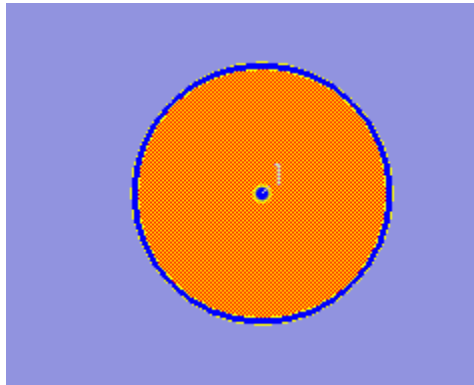


Figure 163. Waypoint handles

### 12.2.9. Occupant Target Handles

When an Occupant Target is selected, two handles are displayed in [Figure 164](#). The center handle allows the target to be moved to another location. The line handle extending from the center can be used to change the orientation of the target. The orientation defines the direction the occupant will face when waiting at the target.

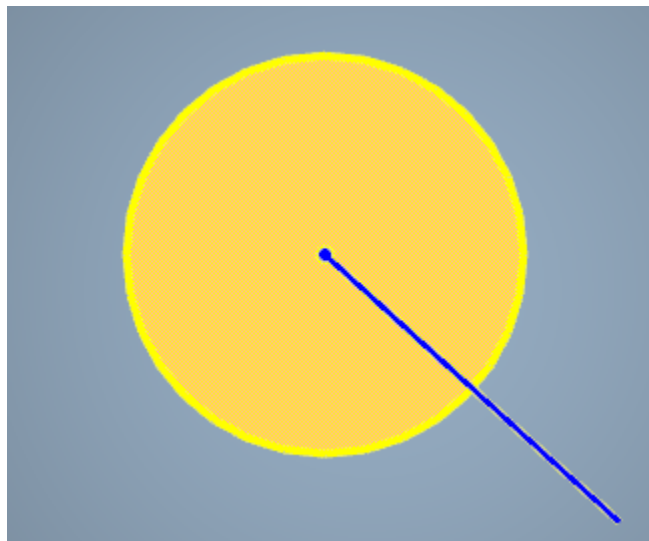


Figure 164. Occupant Target handles

## 12.3. Enabling and Disabling Objects

Several objects in Pathfinder can be enabled or disabled. Disabling an object removes the object from the model. A disabled object does not appear in the simulation. Following objects can be enabled and disabled in Pathfinder:

- Occupants
- Occupant sources
- Rooms: disabling a room will automatically disable its occupants and occupant sources; all connected doors, stairs and ramps will also be disabled.
- Doors: disabling a door will automatically disable its occupant sources.
- Stairs
- Ramps
- Measurement regions

To enable or disable an object, right click on the object in the 3D view or in the navigation tree and select **Enable** or **Disable**. Disabled objects are invisible in the 3D view. In the navigation tree, disabled objects are greyed out and crossed out.

## Chapter 13. Working with Large Models

Large models can contain thousands of occupants and geometry components. Pathfinder provides a variety of utilities to assist in organizing and reorganizing complex models to maintain the proper associations between objects.

### 13.1. Selection

While it is simple to select an individual object from either the Navigation view or the Model view, doing so for every single object to edit can be cumbersome. The following actions are available from the right-click menu to accelerate selecting networks of objects.

#### Select Referencing Objects

Selects other objects in the model referencing the selected objects. Alternatively, use the [Show Referencing Objects](#) action to show these objects in a separate list.

#### Select Connected Components

Selects geometry that is directly connected to the selected objects, or the entire network of selected objects.

#### Select Subitems

Selects all direct sub-items of the selected group node.

#### Select Non-Group Descendants

Selects all non-group descendants that are part of the selected group node.

### 13.2. Show Referencing Objects

Most models can have complex relationships between objects, with some objects referencing others. In these cases, it can often be helpful to start at the referenced object and see which objects are referencing it. To do this, right-click the referenced object and choose **Show Referencing Objects**.

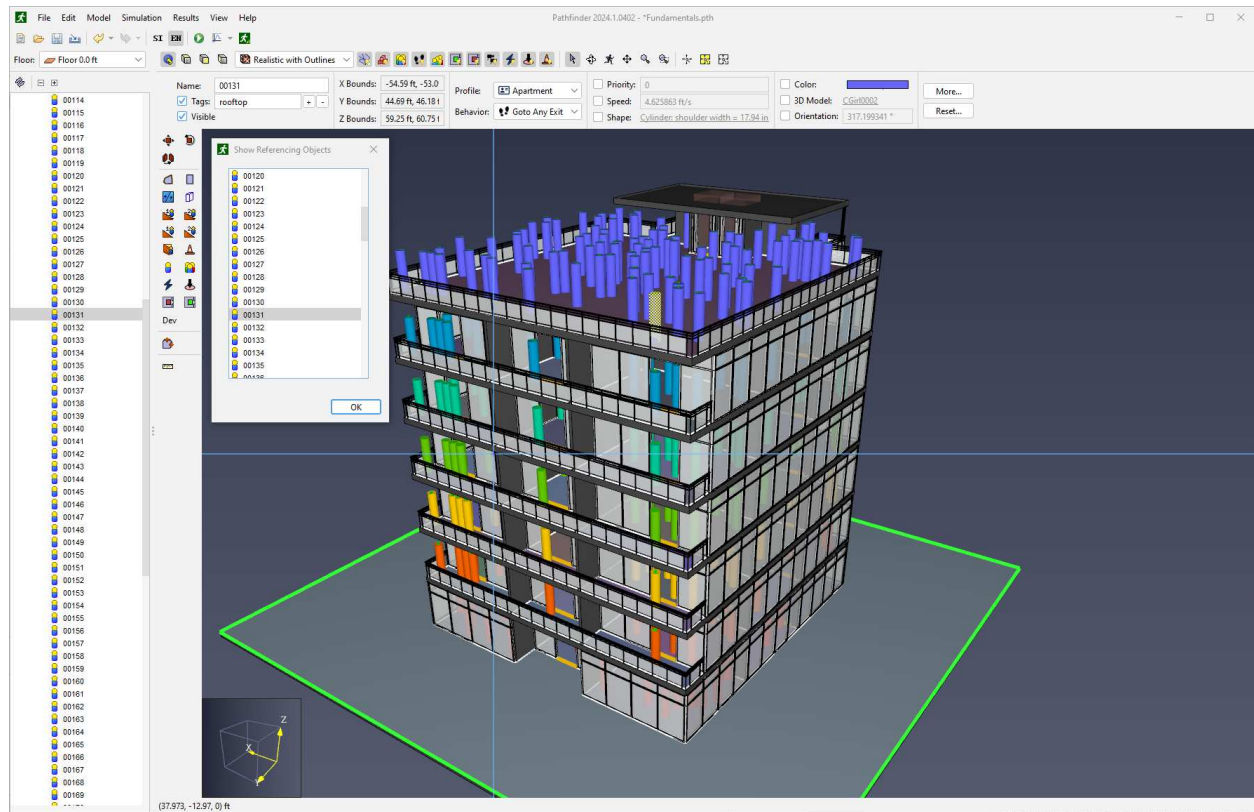


Figure 165. The Show Referencing Objects Dialog

This opens the floating **Show Referencing Objects** dialog, which displays each of the referencing objects in a list. Objects in the list have their selection state synchronized with the selection in the Navigation View and the Model View. This includes the ability to right-click an object in this tool and access the object's context menu.

#### NOTE

The list does *not* dynamically update as references are changed. It only captures the relationships at the time the action was invoked.

## 13.3. Object Tags

Any number of custom Tags may be added to identify and organize objects with common characteristics. Most non-group node objects in the Navigation View support tags. Once added, the given Tag will also appear in its own group in the Navigation View.

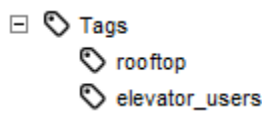


Figure 166. Tags Appearing in the Navigation View

### 13.3.1. Editing Tags

Tags are assigned in the property panel. A Tag will be created for each unique character string separated by spaces or other delimiter characters.

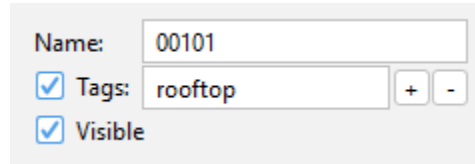


Figure 167. Tag Editor Field

#### 13.3.1.1. Adding and Removing Tags During Multiple Selection

If multiple objects are selected, the Tag editor will display as **<mixed>**. Editing the Tag field in this state can potentially cause unexpected changes.

To work around this, the Tag editor includes custom buttons for adding or removing Tags in a mixed selection state.

The + button will launch an editor where you may enter the Tags you would like applied to each selected object.

The - button will launch an editor where you may enter the Tags you would like to remove from each selected object.

### 13.3.2. Finding Objects with a Shared Tag

One of the main purposes of Tags is to quickly find and identify related objects. Pathfinder provides several ways to quickly find tagged objects.

#### Double-click

Double-click Tags in the Navigation View to show the objects that are tagged with any of those in the selection. The objects will be listed in a separate window.

#### Context-menu actions

Right-click Tags in the Navigation View to choose one of the following actions:

##### Show Tagged Objects

Launches a helper dialog to either select the objects tagged with those in the selection or show them in a dialog.

##### Select Referencing Objects

Selects other objects in the model referencing the selected Tag.

**NOTE**

This may return a different set of objects than **Show Tagged Objects**, as it includes objects with any kind of reference to a tag, and not just objects that are tagged with those in the selection.

### Show Referencing Objects

Lists the referencing objects in a floating window.

**NOTE**

This may return a different set of objects than **Show Tagged Objects**, as it includes objects with any kind of reference to a tag, and not just objects that are tagged with those in the selection.

#### 13.3.2.1. Show Tagged Objects

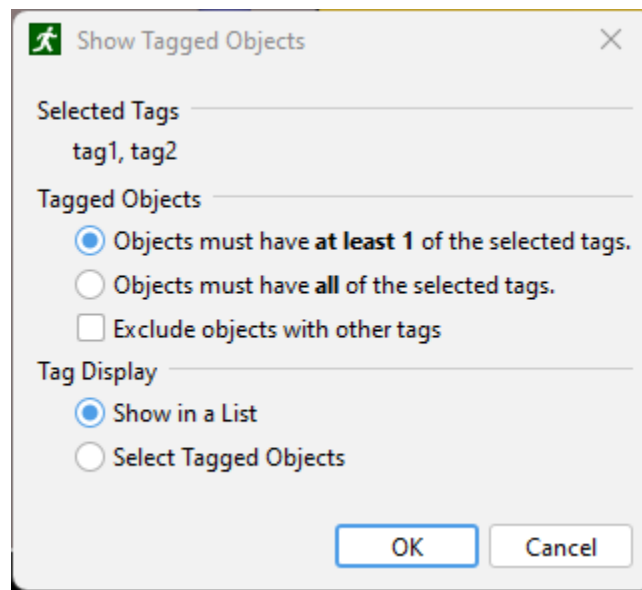


Figure 168. The Show Tagged Objects Dialog

#### Selected Tags

Displays the currently selected tags that will be used for the search.

#### Tagged Objects

Indicates how the tagged objects are chosen. The following options are available:

##### Objects must have at least 1 of the selected tags

This option is available when more than one tag is selected. It indicates that the chosen objects must have at least one of the selected tags. For instance, if the selection contains tags, **tag1** and **tag2**, the chosen objects will have either **tag1** or **tag2** or both **tag1** and **tag2**.

**Objects must have all of the selected tags**

This option is available when more than one tag is selected. It indicates that the chosen objects must have *all* of the selected tags. For instance, if the selection contains tags, **tag1** and **tag2**, the chosen objects will have both of the tags.

**Exclude objects with other tags**

Excludes objects if they have tags that are not in the selection. For instance, if the selection contains tags, **tag1** and **tag2**, and an object contains these tags but also contains **tag3**, it will be excluded from the result. This may be useful, for instance, to find objects that contain exactly the selected tags and no more.

**Tag Display**

Specifies what to do with the resulting tagged objects. The following options are available:

**Show in a List**

Shows the resulting tagged objects in a floating window, similar to the [Show Referencing Objects dialog](#).

**Select Tagged Objects**

Selects all the resulting tagged objects.

## 13.4. Bulk Renaming

As objects are being created via the drawing tools, geometry import, manager dialogs, and Copy / Paste, the names assigned by Pathfinder can become increasingly complex. After some time it may be desirable to change many thousands of object names to make the model more clear.

To open the Rename dialog, right click one of the selected objects in either the 3D view or in the Navigation view and select **Rename....**

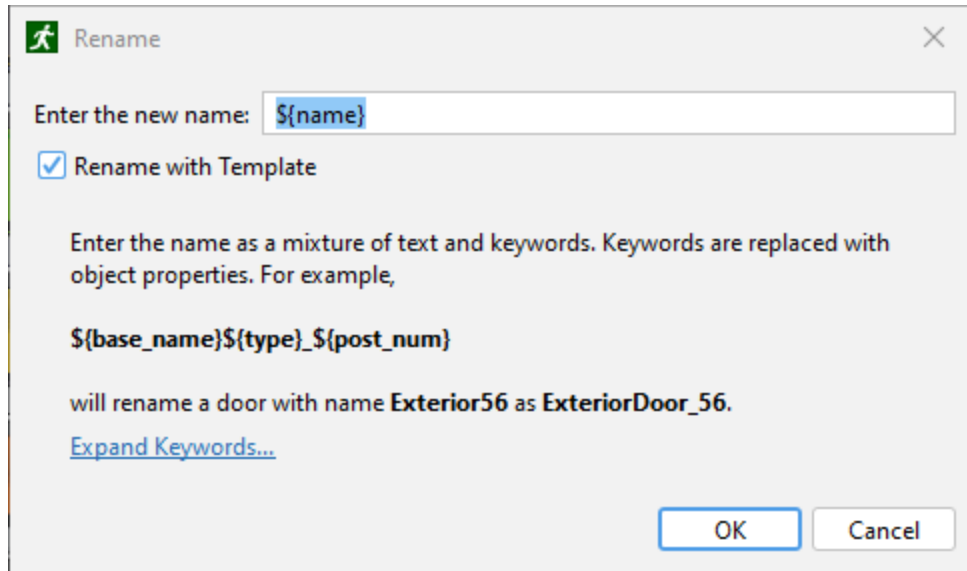


Figure 169. The Rename Dialog

To rename all selected objects with a common name, simply enter the desired new object name in the text field, and click **OK**.

For more complex renaming scenarios, you may check the **Rename with Template** checkbox.

When the **Rename with Template** checkbox is selected, you may create a name as a mixture of both text and keywords. Keywords are uniquely interpreted for each of the selected objects based on the object's properties. Using this approach a large set of mixed-type objects can all be renamed according to a standard syntax.

For example:

```
*${base_name}${type}_${post_num}
```

will rename a door called Exterior56 to ExteriorDoor\_56.

The full set of available keywords are as follows:

#### **name**

The existing name of object.

#### **base\_name**

The existing name of the object without any leading or trailing numbers.

#### **parent\_name**

The name of the parent object.



**parent\_name-x**

Where x is a number. Will go up the hierarchy x times.

**pre\_num**

Any number at the beginning of the existing name.

**post\_num**

Any number at the end of the existing name.

**selection\_index**

The 1-based index of the item in the current selection.

**global\_index**

The 1-based index of the object within its top-level type group.

**local\_index**

The 1-based index of the item within its parent group.

**results\_id**

The globally-unique 64 bit integer ID of the object, if available.

**type**

The object type.

Each of these keywords are detailed in the Rename dialog dropdown.

To make it easier to work with name templates, the keyword names in the dialog work as buttons. Clicking one of the keyword buttons will inject the keyword directly into the editor dialog at the current caret location.

# Chapter 14. Model Analysis

Pathfinder contains some useful tools to analyze various properties of a model.

## 14.1. Measurement Regions

Measurement regions cause the simulator to output time history data for velocity and density within a specific region on the navigation mesh. See [Section 16.5](#) for information about Measurement Region results.

Measurement regions can be created using the **Add a Measurement Region** tool .

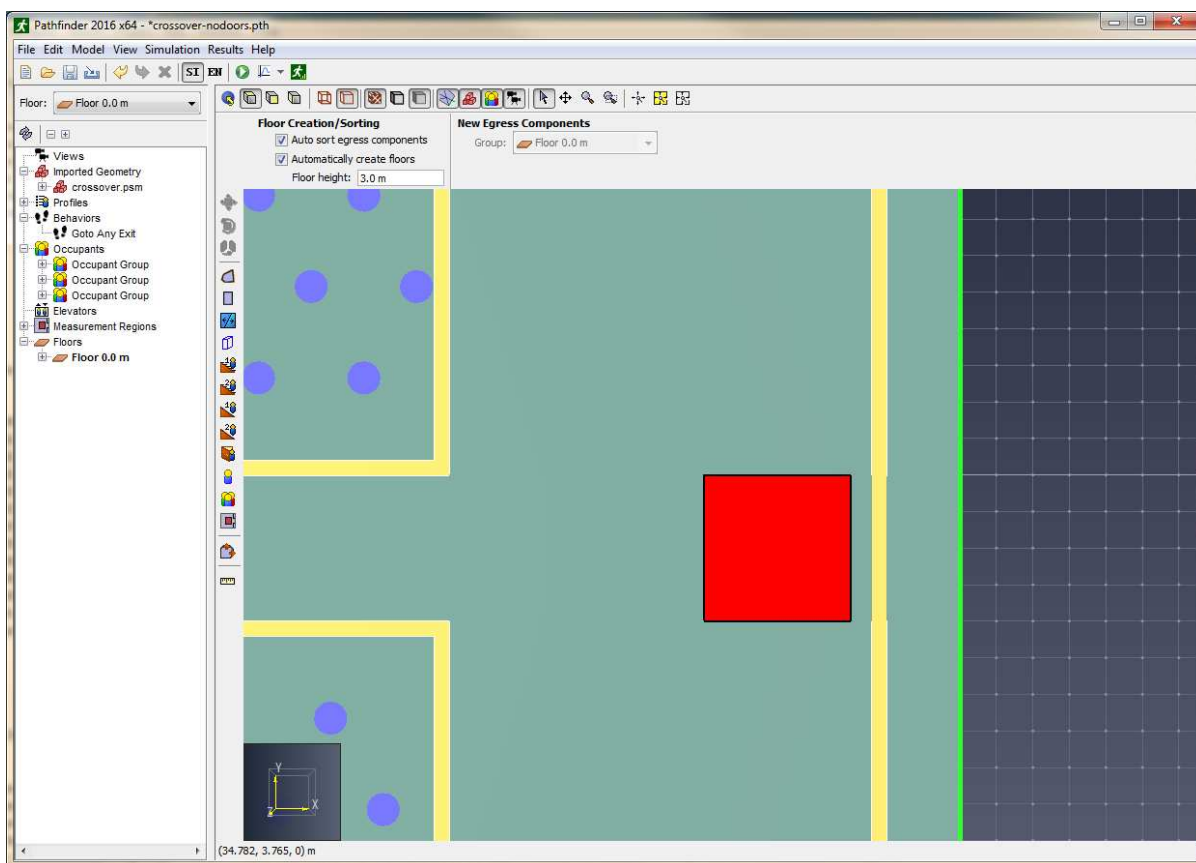


Figure 170. An example of a measurement region placed in front of an exit door.

Adding one or more density regions will cause the simulator to output a CSV file named **filename\_measurement-regions.csv** containing data for each density region. The data in this file can be used to plot fundamental diagrams in the measurement area.

Output frequency for measurement region data is controlled by the **Data Output Freq.** parameter on the Output tab of the Simulation Parameters dialog.

To ensure accurate results, measurement regions must:

- Intersect exactly one room,
- Not extend beyond the boundary of the room, and
- Not intersect any interior walls within the room.

Essentially, measurement regions should be placed on open space that will be used by occupants. Ideally, the measurement region should not be larger than the area where the steady flow under study occurs. If the measurement region is too large, results might indicate a lower value than expected because the quantity is integrated over the entire region.

For additional information about measurement regions, please refer to the Pathfinder Technical Reference.

## 14.2. Measuring Distances

Distances can be measured by using the measuring tool .

To do so, select the measuring tool from the 3D or 2D view. To measure distance along a sequence of points, left-click each point. After each point in the path has been specified, right-click to display the cumulative point-to-point distance in a dialog box.

When measuring distances in the 3D view, the distance is taken as the actual distance between snapped points. When measuring distances in the 2D view, however, the distance is taken by projecting the points onto a plane parallel to the camera view plane, and then taking the distance.

# Chapter 15. Simulating

## 15.1. Parameters

The Simulation Parameters dialog provides a way to control certain features of the simulation, as well as provide some default values.

### 15.1.1. Time Parameters

The **Time** tab (Figure 171) provides the following options:

#### Time Limit

Can be used to automatically stop the simulation after a set simulation time.

#### Time Step Size

Controls the resolution of simulation time steps. Increase the time step size to speed up simulations, reduce the time step size to ensure simulation accuracy.

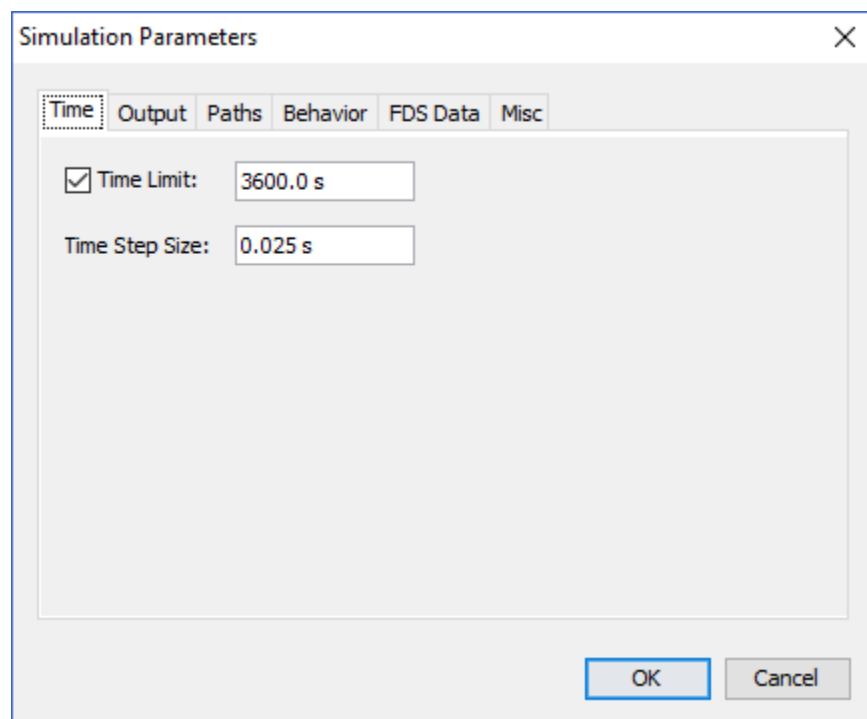


Figure 171. The Time tab of the Simulation Parameters dialog

### 15.1.2. Output Parameters

The **Output** tab (Figure 172) provides the following options:

### 3D Output Freq

controls the time between 3D output file updates. Increasing this value causes data to be written less often, leading to less disk usage and run faster simulations (no file write delay), but can produce a misleading 3D results visualization. In the 3D results visualization, occupants will move in a straight line between two data points - if the two points are far apart in time, an occupant might appear to pass through an obstruction when it actually navigated properly.

### Data Output Freq.

controls the time between data output file updates. Increasing the value causes fewer rows to appear in resulting CSV files and fewer timesteps to be recorded in results JSON files (i.e. lower data resolution). This option has a negligible effect on simulation performance or disk usage, but can affect performance in the 2D results.

### Runtime Output Freq

controls the time between simulation status updates in the Run Simulation dialog. This option has a negligible effect on simulation performance or disk usage.

### Congestion Velocity

controls the velocity threshold for determining whether an occupant is experiencing congestion. An occupant is considered in a state of congestion when their average velocity is below this value.

### Congestion Averaging Time

controls the trailing time period over which to average an occupant's velocity for congestion evaluation.

### Detailed Occupant Data

controls how output data is generated for occupants with **Output Detailed Data** enabled (see [Section 16.9](#)).

#### Merge into one file

All occupants with detailed output data enabled are written to the same file. The file is named `filename_occupants_detailed.[csv, json]`, where `filename` is the name of the saved PTH file without the `.pth` extension.

#### Create one file per occupant

Each occupant with detailed output data enabled is written to a different file. The file is named, `filename_occupant_id_occname.[csv, json]`, where `filename` is the name of the saved PTH file without the `.pth` extension, `id` is the integer id of the occupant assigned by the simulator, and `occname` is the name of the occupant specified in the user interface.

**Include Seek Speed in Measurement Region Files**

controls whether to include an additional column in the measurement region files called "SeekVelocity", which indicates how fast an occupant is moving relative to the direction of their goal. For instance, if an occupant wants to move in the +X direction, and they are moving in the +X direction at 1.2 m/s, this column will show 1.2 m/s; however, if they instead move in the +Y direction, this will become 0 m/s because they are not moving toward their goal. NOTE: This value is always  $\geq 0$ , even if the occupant is moving backward away from their goal.

**Write Occupant Parameter File**

Indicates whether to write the **Occupant Parameters** file (see [Section 16.6](#)).

**Write JSON Output Files**

Controls whether the JSON output files described in [Chapter 16](#) are generated.

<b>NOTE</b>	This parameter does not affect the <code>_views.json</code> or <code>_summary.json</code> files.
-------------	--

**Enable Interpersonal Distance Reporting**

Enabling this feature will cause the simulator to create the additional output files detailed in [Section 16.7](#). The distance calculations performed to create these files are computationally expensive and increase simulation runtime by about 10%.

**Reference Distance**

When interpersonal distance reporting is enabled, this value is used to calculate exposure data and is reported as SD in `filename_sd_accumulated.csv` (see [Section 16.7](#)).

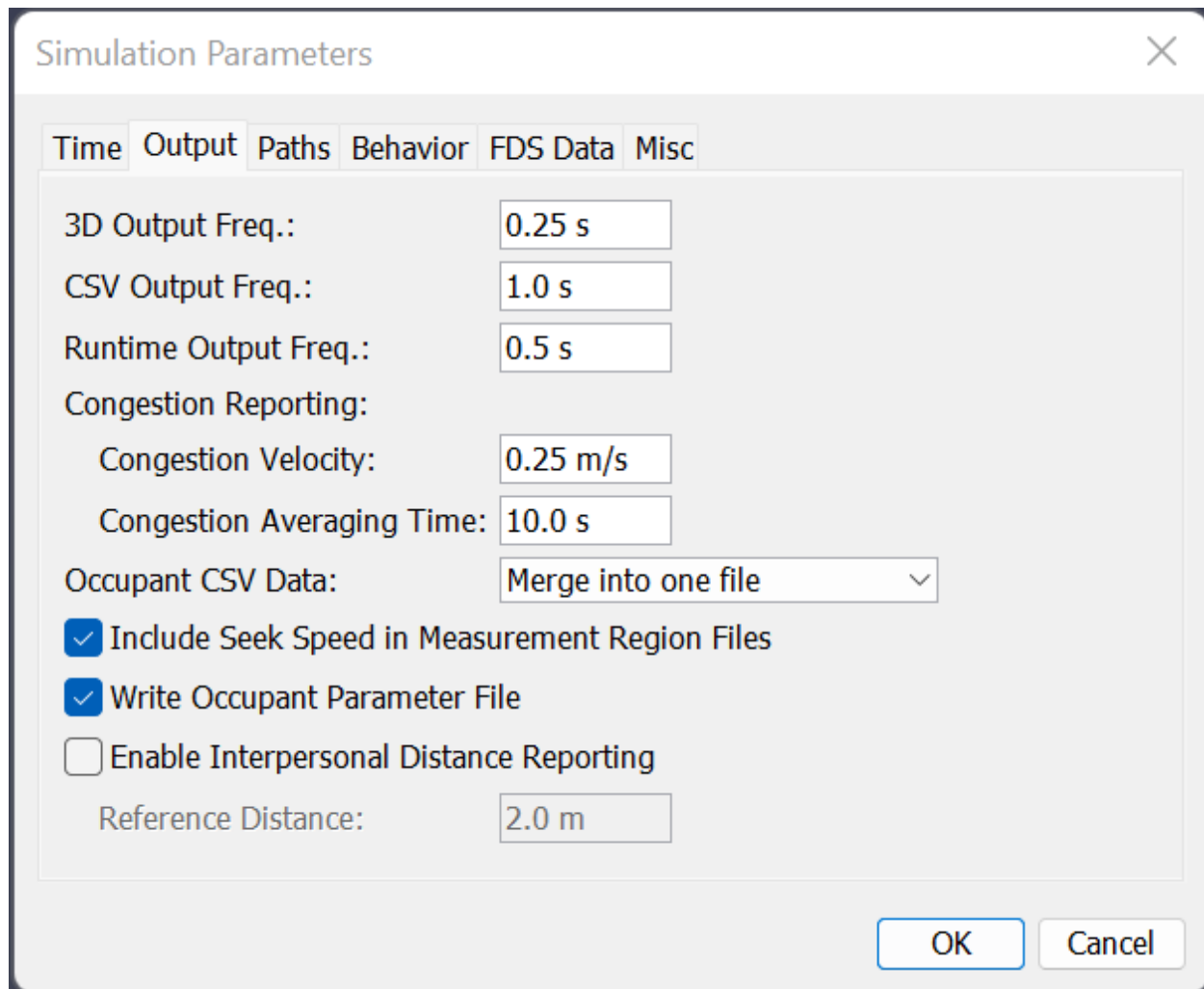


Figure 172. The Output tab of the Simulation Parameters dialog

### 15.1.3. Paths Parameters

The **Paths** tab (Figure 173) contains properties that control how occupant paths are generated and how the navigation mesh is triangulated. Generally, these settings should not be changed.

#### Max Agent Radius Trim Error

This parameter affects how accurately occupants can navigate through tight spaces when the occupants in the simulation have varying sizes. The larger this value is the less likely an occupant is to navigate through a space that has a width close to their body diameter. With larger values, however, the simulation will consume less memory and start faster (sometimes much faster if every occupant has a different size). Each occupant is guaranteed to be able to fit through a space with width equal to the occupant's diameter plus twice this value.

#### Navigation Mesh Refinement

Controls whether and how finely the navigation mesh is refined into smaller triangles. The

following values are allowed:

**None**

Disables mesh refinement. This is the default option and provides the best performance. It allows Pathfinder to generate the fewest and largest possible triangles, which works well in most cases with Pathfinder's search algorithms.

**Constrain triangle area**

Specifies an upper bound on the area of the triangles in the navigation mesh. In some situations, smaller triangles can be useful for more accurately determining shortest distance, which may fix issues where occupants are taking long paths. Smaller values result in more triangles, which can significantly impact simulation performance, increasing the time to simulate with more triangles.

**Constrain Edge Length**

Similar to **Constrain triangle area**, this results in smaller triangles, which may in some cases fix issues where occupants are taking long paths.

**Max Edge Length**

Controls the maximum length of any single edge on a room boundary.

**Min Angle**

Sets a lower bound on the angle between adjacent triangle edges. Larger values increase the quality of the triangles, preventing long, thin triangles, but values greater than 30 degrees can cause Pathfinder to "freeze" when it attempts to generate a simulation input file.



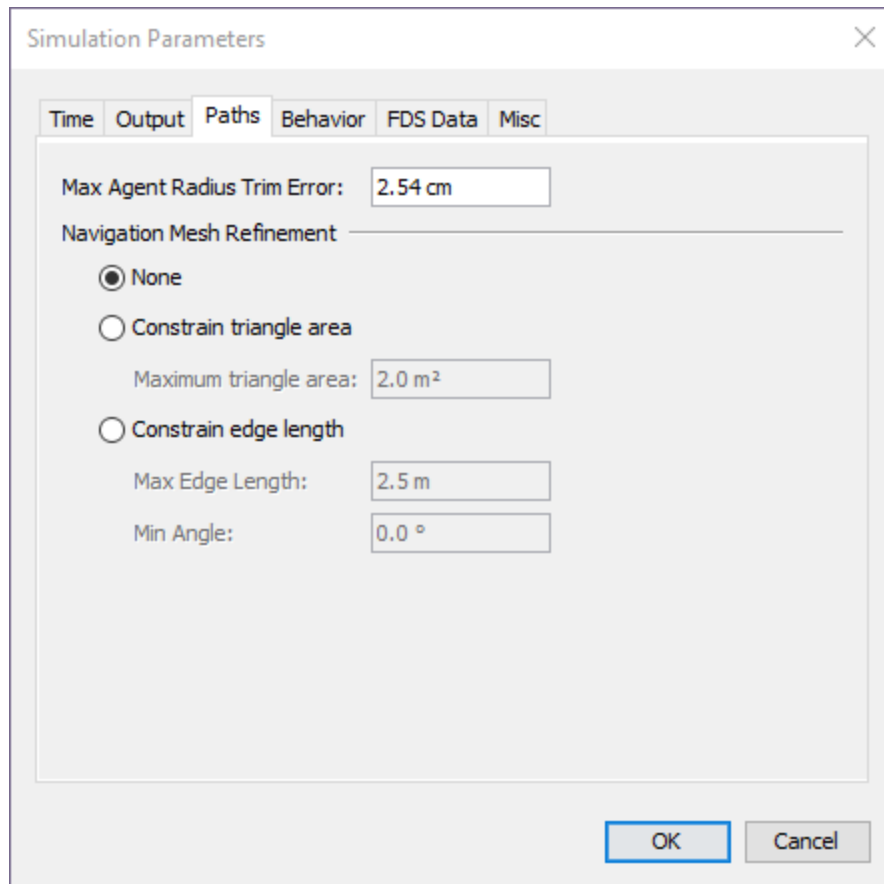


Figure 173. The Paths tab of the Simulation Parameters dialog

### 15.1.4. Behavior Parameters

The **Behavior** tab allows you to set options for Pathfinder's two primary simulation modes: [SFPE](#) and [Steering](#). To select a simulation mode, choose **SFPE** or **Steering** from the **Behavior Mode** drop-down box.

#### 15.1.4.1. SFPE Mode Parameters

The SFPE mode uses the set of assumptions presented in the *Engineering Guide to Human Behavior in Fire* ([SFPE 2019](#)) and can give answers extremely similar to these hand calculations, depending on selected assumptions. In SFPE simulations, the mechanism that controls simulation movement is the door queue. The SFPE mode uses a simple set of assumptions and usually completes much faster than a comparable steering mode simulation in terms of CPU time.

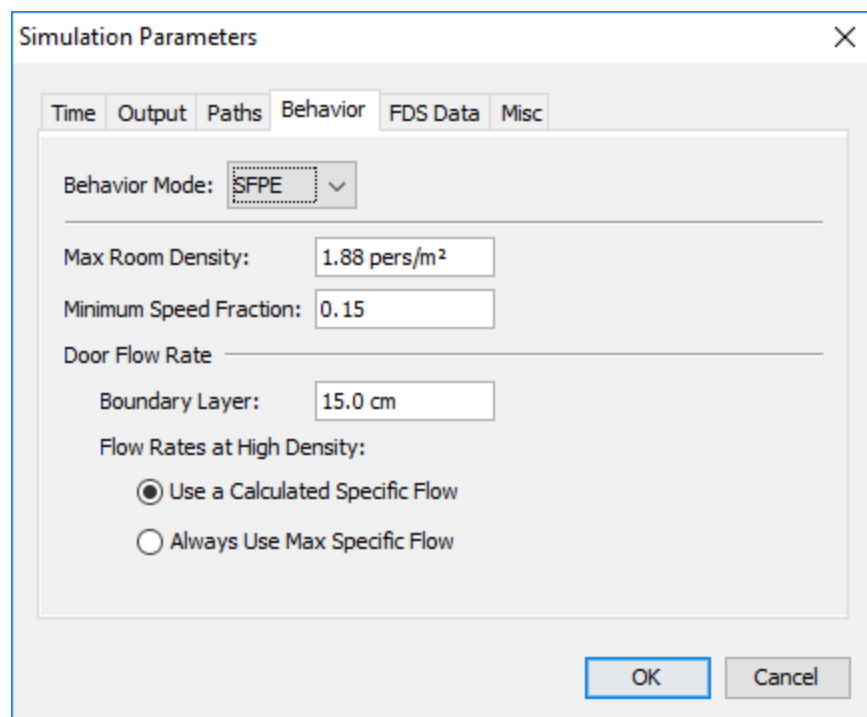


Figure 174. SFPE Behavior tab of the Simulation Parameters dialog

The **SFPE** mode supports the following options:

### Max Room Density

Controls the density at which doors will no longer admit occupants into a room. Using an artificially low value for this number will give faster evacuation times. Using a higher number **3.6 - 3.8** can cause extremely slow evacuation times. Using values above **3.8 pers/m<sup>2</sup>** can cause the simulation to become stuck due to the density dependent velocity calculation.

### Door Flow Rate → Boundary Layer

This value is subtracted from both sides of a door to calculate the effective door width, controlling the flow rate equation. For example, with a **Boundary Layer** setting of **150 mm**, a **1.0 m** door would be reduced to a **0.7 m** opening giving an [stem f335bb7424c5581895ef5a1237366742] of  $(1.32 \text{ pers/s-m} * 0.7 \text{ m}) = 0.924 \text{ pers/s}$ .

#### NOTE

This same **Boundary Layer** is used when calculating the room density as described in the **SFPE Mode Parameters** section of the Pathfinder Technical Reference.

### Door Flow Rate → Flow Rates at High Density

controls how specific flow for doors is calculated. Specific flow is a measure of occupants per unit of time per unit of effective width. For each door, the specific flow is multiplied by the effective door width to calculate the door flow rate in occupants per unit of time.

### Use a Calculated Specific Flow

Specific flow is calculated as a function of room density for a room adjacent to the door. In cases of counterflow, the room with the higher density controls the combined specific flow of the door.

### Always Use Max Specific Flow

Allows the door to flow at the optimum density. Minimum Speed Fraction\* can be used to set a lower limit on occupant speed. Setting this value too low can cause evacuation times to increase significantly in cases of high initial loading of rooms or if the **Max Room Density** is set very high.

#### 15.1.4.2. Steering Mode Parameters

The Steering mode is more dependent on collision avoidance and occupant interaction for the final answer and often gives answers more similar to experimental data than the SFPE mode (i.e. steering mode often reports faster evacuation times). Door queues are not explicitly used in Steering mode, though they do form naturally.

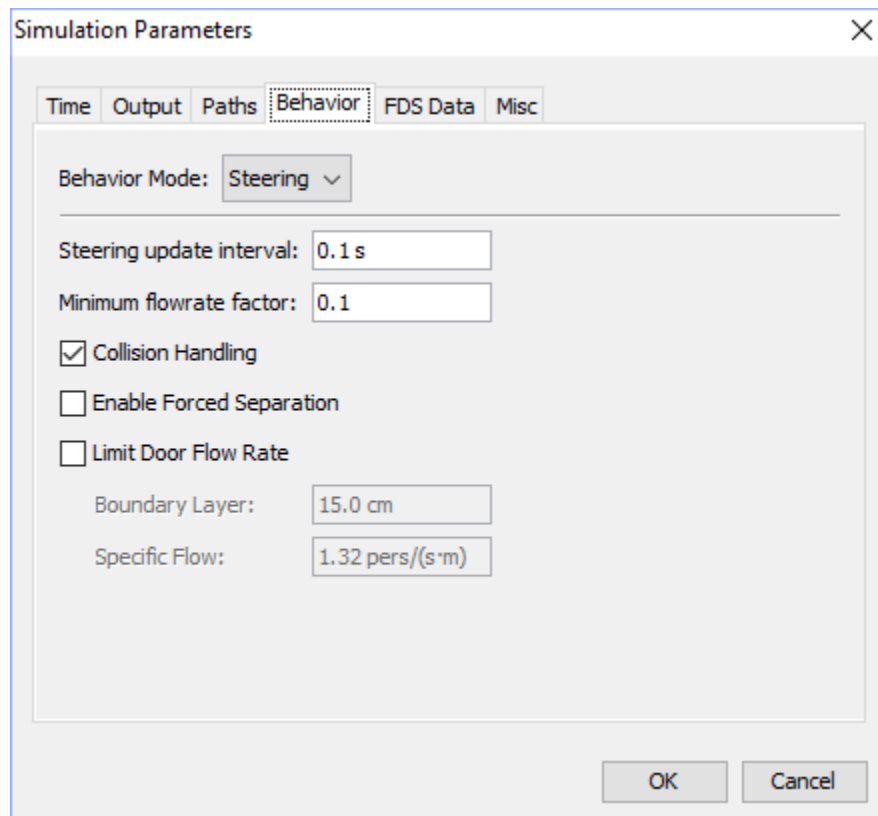


Figure 175. The Steering Behavior tab of the Simulation Parameters dialog

The **Steering** mode supports the following options:

### Steering update interval

Specifies how often (in simulation time) to update the steering calculation. This could also be considered to be the cognitive response time of each occupant. The higher this number, the faster the simulation will run, as long as the simulation time step is less, but the poorer the decision making skills of each occupant will be.

### Minimum flowrate factor

This is used when agents are deciding which doors to use when there are queues at the doors. The factor is multiplied by the door's nominal flowrate to determine a minimum observed flowrate. A non-zero factor will cause a door to always appear to be flowing even if it is not. This helps prevent occupants from excessively switching doors when flowrates are very low; however, it can also cause occupants to stick with the same door even if it cannot flow any more, such as a door into a refuge room that has filled completely near the door. If occupants are refusing to leave a blocked door into a room when other doors are available, try setting this parameter to 0.

### Collision Handling

Controls whether occupants avoid one another and can collide with each other.

### Enable Forced Separation

When enabled, this will cause occupants to attempt to always strictly maintain their **Personal Distance** ([Section 5.1.7](#)). When disabled, Personal Distance will only apply while queueing.

#### NOTE

In Pathfinder 2020.2, this parameter was used experimentally to perform social distancing among occupants. In Pathfinder 2020.3 and beyond, social distancing is performed using the **Social Distance** parameter in occupant profiles. **Enable Forced Separation** may conflict with **Social Distance**, so when loading Pathfinder 2020.2 PTH files that have the **Enable Forced Separation** box checked, you will be asked if you want to convert from the forced separation model to the social distance model. If you choose to do so, **Enable Forced Separation** will be disabled, and all profiles that have a non-default **Personal Distance** of **.08 m** will be converted such that the **Social Distance** is calculated from the **Personal Distance**, and the **Personal Distance** is set to the default. Because **Social Distance** is defined as the center-to-center distance of the occupants and **Personal Distance** is the distance between their shapes, **Social Distance** is estimated as **Personal Distance** plus occupant diameter. If the diameter is specified as a distribution, the distribution average is used as the diameter. Occupants that override the **Personal Distance** setting in their profile will be converted similarly.

### Limit Door Flow Rate

When checked, this imposes a maximum flow rate on doors unless they have it explicitly turned off ([Section 3.5.3](#)). The flow rate for each door is calculated from the **Boundary Layer** and **Specific Flow**, similarly to SFPE Mode. The difference between Steering and SFPE Modes is that Steering does not allow flow rates to be based off room density.

## 15.1.5. Miscellaneous Parameters

The **Miscellaneous** tab ([Figure 176](#)) provides rarely-changed options, including the following:

### Show Runtime Visualization

Whether to enable the debug view when running a simulation. This has the same effect as starting a simulation using **Simulation** → **Debug Simulation**.

### Save Restart Files

Whether to save a restart file while the simulation is running. Enabling this option will cause a snapshot of the simulation to be saved periodically to the restart file while the simulation is running. This is useful for debugging or if the computer loses power while running a simulation, as the simulation can be resumed from any of the saved snapshots by using **Simulation** → **Resume Simulation**.

### Snapshot Interval

Defines how often to save snapshots to the restart file in simulation time (not to be confused with wall-clock time).

### Curve error

Controls how imported curved CAD lines are turned into multi-segmented lines for use in floor extraction and for display purposes. Smaller error values result in more segments.

### Face error

Controls how imported curved CAD faces are turned into triangles for use in floor extraction and for display purposes. Smaller error values result in more triangles.

### Default Trigger Idle Time

When an occupant chooses to use a trigger while waiting an indeterminate amount of time, this property defines the period in which the occupant will schedule the trigger. (see [Section 10.3.3](#)).

### Occupant Target Resolve Time

The time limit in which occupants must reserve an Occupant Target when using the Occupant Target reservation system (see [Section 9.6.2](#)).

### Goto Occupant Search Interval

For the [Goto Occupant](#) behavior action, if a destination occupant is not found, this parameter defines the amount of simulation time until another search is performed. Lower values may increase the time to complete a simulation if there are many potential destination occupants, but will allow occupants to respond more quickly to changes in the model.

### Enable visualized sideways movement of vehicles

Allows vehicles to appear as if they are moving sideways rather than turning as they move. Enabling this option only changes the visualization and not the simulation of how vehicles move. If this option is enabled, the vehicle shape must also have the **Allow sideways movement** option checked.

#### NOTE

The **Allow sideways movement** option will only be visible in the **Edit Vehicle Shapes** dialog if this parameter is enabled.

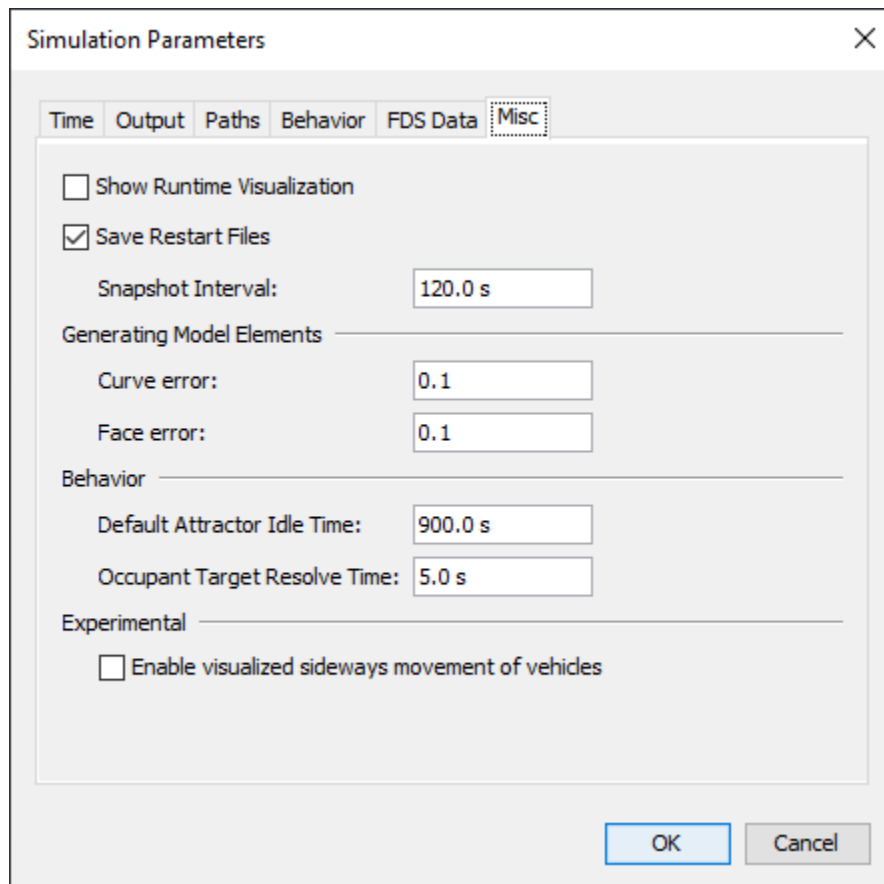


Figure 176. The Miscellaneous tab of the Simulation Parameters dialog

## 15.2. Starting and Managing a Simulation

To run a simulation: on the Model menu, click **Run Simulation** . The simulation will begin and the **Run Simulation** dialog (shown in [Figure 177](#)) will appear.

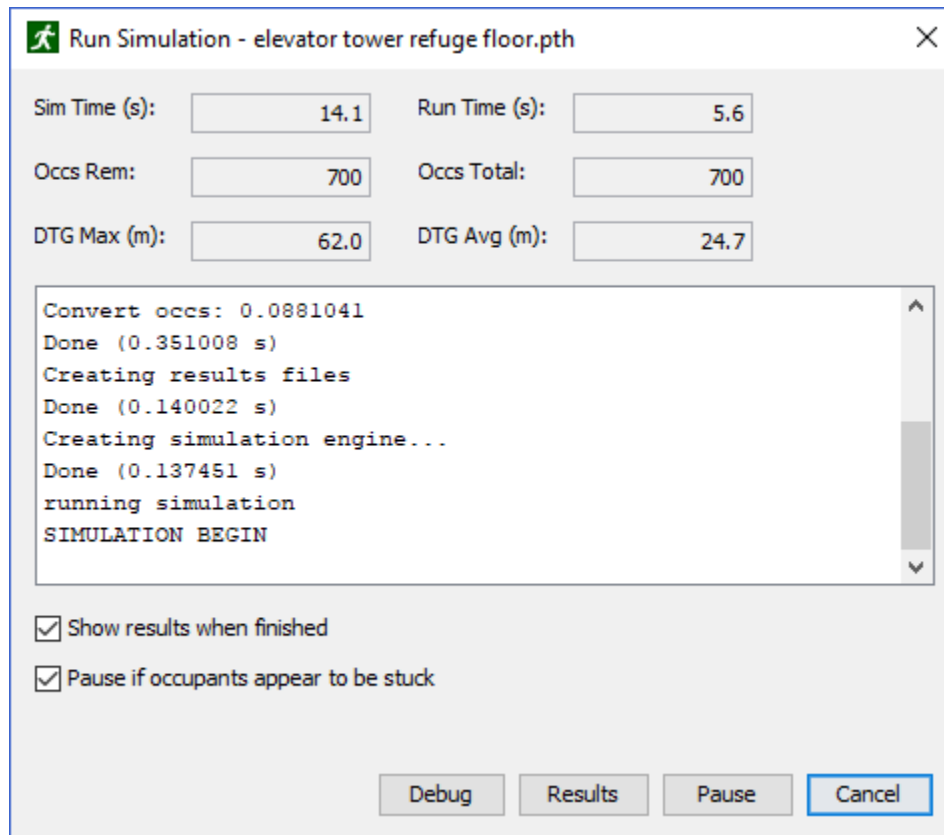


Figure 177. The Run Simulation dialog showing a partially complete simulation.

In this dialog, the abbreviation **DTG** stands for distance to goal. The maximum distance to goal represents distance to goal for the occupant farthest from its goal. The average distance to goal is the average of all occupants' distances to their respective goals.

The **Debug** button launches a runtime visualization that shows the progress of the simulation as it is taking place. This function is different from the **Results** button which launches the 3D visualization view for simulation results.

A simulation can also be paused, resumed, and cancelled at any time.

### 15.2.1. Simulating via Command-line

Simulations can be run from the command line using the `testsim.bat` script located in the Pathfinder installation directory. The following sections detail the two supported use-cases for this script.

**NOTE**

Running a simulation through the command line will not provide a management dialog through which the simulation can be paused and resumed. Additionally, when running a simulation this way it may be desirable to manually create the geometry file for visualization. To do this, select **File › Save Imported Geometry File**.

**15.2.1.1. Running a Single Simulation**

To run a single simulation from the command line:

1. Open a command prompt in the Pathfinder installation directory.
2. Run the following command:

```
testsim.bat "full_model_path"
```

Where *full\_model\_path* is the full path to the Pathfinder model you wish to run.

**NOTE**

This path can point to either the *.pth* model file, or to a *.txt* input file written by Pathfinder. *.txt* input files are automatically created by Pathfinder when a model is run, but can be manually created in the user interface by selecting **File › Save Simulator Input**.

**15.2.1.2. Running all Simulations in a directory**

To run all of the simulations in a directory:

1. Open a command prompt in the Pathfinder installation directory.
2. Run the following command:

```
testsim.bat "path"
```

Where *path* is the path to a directory containing Pathfinder *.pth* model files. This will run all of the model files in the provided directory, but not in its subdirectories.

**NOTE**

This will only run *.pth* model files, and will not run *.txt* inputs files.

## 15.3. Stopping and Resuming a Simulation

When running a simulation, there is the option to pause and resume, but this requires Pathfinder



to be running the entire time. Sometimes the need arises to be able to stop a simulation and resume later between Pathfinder sessions, such as if the computer needs to be restarted after installing a system update.

To stop a simulation:

1. Press **Cancel** during a simulation in the simulation dialog.
2. Pathfinder will ask the user if a snapshot should be made.
3. After clicking **Yes**, the current state of the simulation is written to a snapshot file.

To resume the simulation at a later time:

1. In Pathfinder open the model to resume.
2. Go to **Simulation** › **Resume Simulation**.
3. Select the time that the simulation was cancelled and click **OK**.
4. The simulation will resume from this time.

# Chapter 16. Results

## 16.1. Summary Report

The summary report file contains information about the simulation geometry, simulation performance, statistics, and usage information for each room, stairway, and door.

Figure 178 shows a portion of an example summary report file.

```
***SUMMARY***SUMMARY***SUMMARY***SUMMARY***SUMMARY***

Simulation:      refuge_report
Version:         2021.2.0719
Mode:           Steering
Total Occupants: 50

Completion Times for All Occupants (s):
Min:            0.1 "00015"
Max:            101.4 "00029"
Average:        36.7
StdDev:         23.3

Completion Times by Behavior (s):
  Behavior Count  Min  Min_Name  Max  Max_Name  Avg  StdDev
  Exit           1   101.4 "00029" 101.4 "00029" 101.4 0.0
  Refuge         47   1.2  "00037"  70.8 "00009"  36.9 20.8
  Wait until end  2    0.1  "00015"   0.1 "00015"   0.1  0.0
  *all behaviors* 50    0.1  "00015" 101.4 "00029"  36.7 23.3

Completion Times by Profile (s):
  Profile Count  Min  Min_Name  Max  Max_Name  Avg  StdDev
  [IMO] Females 30y-50y 13  0.1  "00015"  69.7 "00016"  33.2 23.9
  [IMO] Females < 30y  13  4.5  "00007"  70.8 "00009"  41.8 21.5
  [IMO] Males 30y-50y  13  0.1  "00022"  64.2 "00012"  29.0 20.6
  [IMO] Males < 30y   11 19.6  "00047" 101.4 "00029"  44.0 24.2
  *all profiles*      50  0.1  "00015" 101.4 "00029"  36.7 23.3

Travel Distances for All Occupants (m):
Min:            0.8 "00015"
Max:            20.9 "00020"
Average:        12.5
StdDev:         3.8

Movement Distance by Behavior (m):
  Behavior Count  Min Min_Name  Max  Max_Name  Avg  StdDev
  Exit           1   1.0  "00029"   1.0  "00029"   1.0  0.0
  Refuge         47   8.0  "00013"  20.9 "00020"  13.3  2.5
  Wait until end  2    0.8  "00015"   1.7  "00022"   1.3  0.4
  *all behaviors* 50    0.8  "00015"  20.9 "00020"  12.5  3.8

Movement Distance by Profile (m):
  Profile Count  Min Min_Name  Max  Max_Name  Avg  StdDev
  [IMO] Females 30y-50y 13  0.8  "00015"  17.8 "00034"  12.0  3.8
  [IMO] Females < 30y  13  8.0  "00013"  16.8 "00024"  12.9  2.4
  [IMO] Males 30y-50y  13  1.7  "00022"  20.9 "00020"  12.7  4.5
  [IMO] Males < 30y   11  1.0  "00029"  17.6 "00004"  12.5  4.1
  *all profiles*      50  0.8  "00015"  20.9 "00020"  12.5  3.8

[Components] All:  4
[Components] Doors: 2
Triangles:        12
Startup Time:     0.3s
CPU Time:         2.8s
```

Figure 178. Listing for an example summary report file.

This file is saved in the simulation directory and given the name `filename_summary.txt` (where `filename` is the name of your saved PTH file). To view it, under the **Results** menu choose **Show Summary File**. The first section shows the mode the simulation was run in, the total number of occupants, and statistics on the completion times for the occupants. It also shows some information about the mesh, including the number of triangles and the doors. This information can be useful when considering the complexity of a simulation from the standpoint of the

simulator.

Each statistic is generated by sampling quantities from the occupants as data points. Only occupants for whom the quantity is relevant are included in the statistic.

Statistics on the following quantities may be printed to the summary file:

### Completion Time

The amount of time it takes for an occupant to complete all of their behavior actions. Occupants are not included in this statistic if they fail to complete all their behavior actions. This may happen either because they became stuck or they did not finish their actions before the simulation ending time as specified in the **Simulation Parameters** dialog.

#### NOTE

For occupants who go to a refuge room, they are considered complete as soon as they reach the refuge room, even though they remain in the simulation until the end. Similarly, occupants who end with a **Wait Until Simulation End** action are considered complete as soon as they begin the action.

### Travel/Movement Distance

The distance that completed occupants have travelled.

### Trigger Usage Time

The amount of time an occupant spent using triggers. If an occupant did not use any triggers, a special **no triggers** line is added. This statistic is only provided if there are triggers in the model.

### Occupant Target Active Usage Time

The amount of time an occupant spent waiting at an Occupant Target. If an occupant did not use any targets, a special **no occupant targets** line is added.

### Occupant Target Reserved Time

The amount of time an occupant reserved an occupant target. This will include the time before they have arrived at the target. If an occupant did not reserve any targets, a special **no occupant targets** line is added.

Some statistics are further broken down by behavior and by profile. For occupants who used multiple behaviors and/or profiles due to **Change Behavior** or **Change Profile** behavior actions, they are reported according to the behavior or profile in use when they completed their actions.

The table at the end of the summary file gives a listing of each component (doors, rooms, and stairs) in the simulation. For each component, the **FIRST IN** column shows the simulation time when the first occupant entered that component. **LAST OUT** shows the simulation time when the

last occupant exited that component. The TOTAL USE column shows how many times a component was entered by occupants. For doors that served more than 1 occupant, the **FLOW AVG** column shows the result of dividing the total use by the amount of time the room was in use (**LAST OUT - FIRST IN**). Each component is annotated with its group names. For example, "Floor 0.0 m → Rooms → Room00" indicates that "Room00" is in "Floor 0.0 m" in a group called "Rooms".

## 16.2. Cumulative Output

The **filename\_out.json** file (where **filename** is the name of your saved PTH file) combines all of the data provided by JSON files described in [Chapter 16](#) into a single JSON file upon completion of the simulation. This file is ideal for accessing results data programmatically to do custom post processing in external applications, but is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

The Cumulative output file uses the file format shown below.

```
{
  "triggers": {},
  "doors": {},
  "groups": {},
  "measurementRegions": {},
  "occupants": {
    ...,
    "socialDistancing": {
      ...,
      "transient": {}
    },
    "initialParams": {},
    "detailed": {}
  },
  "targets": {},
  "rooms": {}
}
```

The file provides the following keys:

### triggers

The contents of the *Triggers History JSON file* ([Section 16.11.2](#)).

### doors

The contents of the *Door History JSON File* ([Section 16.3.2](#)).

**groups**

The contents of the *Groups Output JSON File* ([Section 16.10.2](#)).

**measurementRegions**

The contents of the *Measurement Regions JSON File* ([Section 16.5.2](#)).

**targets**

The contents of the *Occupant Target JSON File* ([Section 16.12.2](#)).

**rooms**

The contents of the *Room History JSON File* ([Section 16.4.2](#)).

**occupants**

The **occupants** key provides quick access to all data relating to individual Occupants. This key combines data from the *Occupant Summary JSON File* ([Section 16.8.2](#)), *Occupant Parameters JSON File* ([Section 16.6.2](#)), *Occupant History JSON File* ([Section 16.9.2](#)), and *Interpersonal Distance JSON Files* ([Section 16.7.2](#)) in to a single JSON object, keyed by Occupant ID. Accessing the data for a given Occupant ID will yield an object containing all data related to that Occupant. The object under the Occupant ID uses the format of the *Occupant Summary JSON File* ([Section 16.8.2](#)), however it injects the following 3 new keys:

**socialDistancing**

This key provides the data of the *Interpersonal Distance JSON Files* ([Section 16.7.2](#)) in a single object, filtered down to match the Occupant ID key. The object under this key uses the format of the *Accumulated JSON File* ([Section 16.7.2.2](#)), however it injects a single new key:

**transient**

This key provides the data of the *Transient JSON File* ([Section 16.7.2.1](#)), filtered down to match the Occupant ID key.

**initialParams**

This key provides the data of the *Occupant Parameters JSON File* ([Section 16.6.2](#)), filtered down to match the Occupant ID key.

**detailed**

This key provides the data of the *Occupant History JSON File* ([Section 16.9.2](#)), filtered down to match the Occupant ID key.

## 16.3. Door History

The door history files provide output data for the doors in your model.

### 16.3.1. Door History CSV File

The **filename\_doors.csv** file (where **filename** is the name of your saved PTH file) includes a five-row header, including the column descriptions, names of associated doors, a numeric ID used by Results, quantities, and units. Each row in the represents a single time step, and includes the following columns:

#### **Time**

The output time for this data row, in seconds. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog.

#### **Remaining (Total)**

The number of occupants remaining in the simulation.

#### **Exited (Total)**

The number of occupants that have successfully passed through an exit door (leaving the simulation).

#### **Time Step Usage [{+/-}{X/Y}]**

The number of occupants who have passed through the specified door since the previous output in the specified direction. For columns with no direction specification, this is the total number of occupants to pass through the door in both directions since the previous output.

#### **Width**

The total width of the specified door, in meters.

#### **Total Boundary**

The total boundary layer of the specified door, in meters.

#### **Queued Occupants**

The number of occupants who are waiting in the queue to pass through the specified door at the current time. This only includes occupants who have actually reached the door and are waiting to enter. Occupants that are stacked up waiting to reach a door will not be counted. This value is only meaningful in SFPE mode.

This file is used to display door flow rate, specific flow, and usage history in Pathfinder. The values in this file can also be plotted by right clicking a door in the 3D Results.

### 16.3.2. Door History JSON File

The **filename\_doors.json** file provides the same data as the **filename\_doors.csv** file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section](#)

15.1.2).

```
{
  "exited": {
    "time": value,
    ...
  },
  "remaining": {
    "time": value,
    ...
  },
  "doorname [{+/-}{X/Y}": {
    "usage": {
      "time": value,
      ...
    },
    "queueingDoorUsage": {
      "time": value,
      ...
    },
    "width": {
      "time": value,
      ...
    },
    "boundary": {
      "time": value,
      ...
    }
  }
},
  ...
}
```

The JSON file contains the following JSON objects:

#### **exited**

An object of time/value entries representing the data in the *\_Exited(total)* CSV column.

#### **remaining**

An object of time/value entries representing the data in the *\_Remaining (total)* CSV column.

#### **doorname [{+/-}{X/Y}]**

Objects representing data for individual doors, in the specified direction. If the object's key does not specify a direction, the data presented is cumulative for both directions of occupant travel.

Each door object contains the following objects:

**usage**

An object of time/value entries representing the data in the *doorname* [{+/-}{X/Y}] CSV column.

**queuingDoorUsage**

(Optional) An object of time/value entries representing the *Queued Occupants* CSV column.

**width**

(Optional) An object of time/value entries representing the *doorname width(m)* CSV column.

**boundary**

(Optional) An object of time/value entries representing the *doorname total boundary(m)* CSV column.

### 16.3.3. Plots

#### 16.3.3.1. Door Flow Rate and Specific Flow

To view door flow rate or specific flow, click **View Door Flow Rates** on the **Results** menu. This opens a time history plot for doors as in [Figure 179](#). This plot shows data from the door history file.

In the left portion of the window is a list of the doors, and on the right is a graph of the data. For each door, the list on the left shows three rows: one for each of the two directions and one for total. The directional data can be hidden by unchecking **Include Directional Data** under the **View** menu. Each row in the list shows group names for each door (if the **Include Group Names in Output** option under **File** → **Preferences** was enabled in Pathfinder when the simulation was run). The group names can be hidden by unchecking **Show Group Names** under the **View** menu of the history plot dialog.



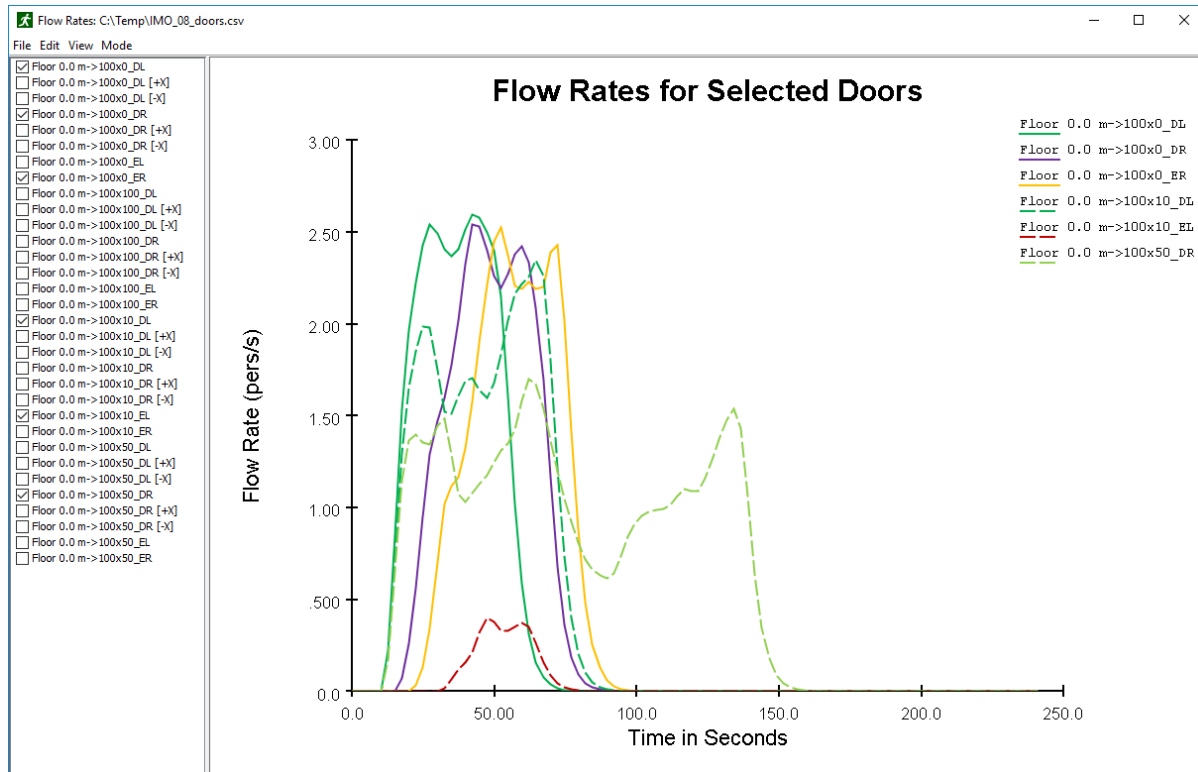


Figure 179. A time history plot for door flow rates

By default, door flow rates are shown. Alternatively, on the **Mode** menu, choose **Specific Flow** to view door specific flow.

There are three filtering modes for presenting the flow rate that are selectable through the **View** menu:

### Raw

this provides raw flow rate, which is simply [stem d5e020fd41765ccb883057193c044ae9], where [stem 3b221d7a5c7d17ebec518e6041269f5d] is the number of occupants to pass through the door in an output time step, and [stem 5a8af6f173febd968ef4c52695efcf85] is the output time step.

### Low-pass Filter

the raw flow rate is filtered with a bi-quad low-pass filter with a user-specified cutoff frequency. This is the default filter, and the default cut-off frequency is .05 Hz. Lower cut-off frequencies produce smoother graphs.

### Moving-average Filter

the raw flow rate is averaged over a user-specified period.

### 16.3.3.2. Door Usage

To view door usage, open the **Door Flow Rate Rates** dialog, and on the **Mode** menu, select **Occupant Counts**. This shows the number of occupants that use a particular door in each output time step.

Alternatively, cumulative totals can be viewed by selecting **Cumulative Occupant Counts**. This shows the total number of occupants to use the door up until that time.

## 16.4. Room History

The room history files provide data on occupant usage of the rooms in your model

### 16.4.1. Room History CSV File

The `filename_rooms.csv` file (where `filename` is the name of your saved PTH file) includes a five-row header, including the column descriptions, names of associated rooms, a numeric ID used by Results, quantities, and units. Each row after the header represents a different time step. The columns are as follows:

#### Time

The output time for this data row, in seconds. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog.

#### Remaining (Total)

The number of occupants still in the simulation.

#### Exited (Total)

The number of occupants that have successfully passed through an exit door (leaving the simulation).

#### Occupant Count

The number of occupants present in the associated room (or stairway) at the current time.

### 16.4.2. Room History JSON File

The `filename_rooms.json` file provides the same data as the `filename_rooms.csv` file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```
{
  "remaining": {
    "time": value,
    ...
  },
  "exited": {
    "time": value,
    ...
  },
  "roomname": {
    "time": value,
    ...
  },
  ...
}
```

The JSON file contains the following JSON objects:

#### **exited**

An object of time/value entries representing the data in the *\_Exited (total)* CSV column.

#### **remaining**

An object of time/value entries representing the data in the *\_Remaining (total)* CSV column.

#### **roomname** [{+/-}{X/Y}]

Objects of time/value entries representing the data in the *\_Occupant Count* CSV column.

### 16.4.3. Plots

#### 16.4.3.1. Room Usage

To view the room history plot, click **View Room Usage** on the results menu. Similarly to door history, the list on the left shows group names for each room (if the **Include Group Names in Output** option under **File** → **Preferences** was enabled in Pathfinder when the simulation was run). The group names can be hidden by unchecking **Show Group Names** under the **View** menu of the history plot dialog.

The data in this file can also be plotted by right clicking a room in the 3D Results.

## 16.5. Measurement Regions Data

The measurement region files provide data for each measurement region in your model.

### 16.5.1. Measurement Regions CSV File

Each row in the `filename_measurement-regions.csv` file (where `filename` is the name of your saved PTH file) represents a single time step, and provides data for each of the following columns:

**Time(s)**

The output time for this data row. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog.

**Density(pers/m<sup>2</sup>)**

The density in the observation region at a specified output time.

**Velocity(m/s)**

The speed in the observation region at a specific output time.

**SeekVelocity(m/s)**

The velocity in the direction of pedestrians' desired path.

**Count(pers)**

The counting heads passing within a time interval.

To display this data as a time history and a speed vs density plot, click **View Measurement Regions** on the **Results** menu.

### 16.5.2. Measurement Regions JSON File

The `filename_measurement-regions.json` file provides the same data as the `filename_measurement-regions.csv` file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```

{
  "regionName": {
    "density": {
      "time": value,
      ...
    },
    "velocity": {
      "time": value,
      ...
    },
    "seekVelocity": {
      "time": value,
      ...
    },
    "count": {
      "time": value,
      ...
    }
  },
  ...
}

```

The JSON file contains the following JSON objects:

### **regionName**

Objects representing data for individual measurement regions.

Each measurement region object contains the following objects:

### **density**

An object of time/value entries representing the data in the *\_Density(pers/m^2)* CSV column.

### **velocity**

An object of time/value entries representing the data in the *\_Velocity(m/s)* CSV column.

### **seekVelocity**

An object of time/value entries representing the data in the *\_SeekVelocity(m/s)* CSV column.

### **count**

An object of time/value entries representing the data in the *\_Count(pers)* CSV column.

## 16.5.3. Plots

### 16.5.3.1. Velocity Mode

To view a speed vs time history graph, click **View Measurement Regions** on the **Results** menu, this is the default mode. To view this mode from other alternative modes, on the **Mode** menu, select **Velocity**. This shows the speed in the observation region(s) in each output time step.

### 16.5.3.2. Density Mode

To view a density vs time history graph, click **View Measurement Regions** on the **Results** menu, and on the **Mode** menu, select **Density**. This shows the density in the observation region(s) in each output time step.

### 16.5.3.3. Speed vs Density Mode

To view a speed vs density graph, click **View Measurement Regions** on the **Results** menu, and on the **Mode** menu, select **Speed vs Density**. This shows how a change of density in the observation region(s) affects the speed of movement.

The three filtering modes explained earlier could also be applied to measurement region graphs.

## 16.6. Occupant Parameters

The occupant parameters files provide a summary of the initial states of almost all parameters assigned to occupants in your model.

These files are useful for verifying that real parameter distributions match the distributions specified in the occupant profiles (see [Section 5.1](#) and [Section 5.1.8](#) for more information). These files are written by default but can be turned off in the Simulation Properties dialog (see [Section 15.1.2](#)).

#### NOTE

It may take a significant number of occupants (> 1000) for the resulting distributions to match to input distributions. The greater the number of occupants using a particular distribution, the greater the likelihood that the distributions will match.

In addition, any occupants that have customized parameters ([Section 5.1.10](#)) will invalidate the distributions. These customized occupants must be manually excluded from the analysis to determine resulting distributions.

### 16.6.1. Occupant Parameters CSV File

Each row in the `filename_occupant_params.csv` (where `filename` is the name of your saved PTH file) represents an individual occupant, and provides the initial parameters for each of the following columns:

**id**

The unique ID of the Occupant.

**name**

The user-specified name of the Occupant.

**start time**

The time when this Occupant was created.

**profile**

The name of the Profile assigned to this Occupant. (See [Section 5.1](#))

**behavior**

The name of the Behavior assigned to this Occupant. (See [Section 5.3](#))

**maxVel**

The maximum velocity allowed for this Occupant.

**shape**

The name of the shape assigned to this Occupant. (See [Section 5.1.1](#) > Shape)

**occupantRadius**

The minimum radius of the Occupant's shape. (See [Section 5.1.1](#) > Shape)

**geometryRadius**

The minimum radius of the Occupant's geometry.

**height**

The height of the Occupant. (See [Section 5.1.1](#) > Height)

**room**

The name of the room where the Occupant was created.

**x**

The X component of the Occupant's initial position.

## **y**

The Y component of the Occupant's initial position.

## **z**

The Z component of the Occupant's initial position.

## **speedInSmoke(m/s)**

Data regarding the Occupant's reaction to Smoke when running an FDS coupled simulation.  
(See [Section 5.1.7](#))

## **ACCEL\_TIME(s)**

The acceleration time assigned to this Occupant. (See [Section 5.1.7](#))

## **MIN\_SQUEEZE\_FACTOR\_CONST**

The maximum reduction factor assigned to this Occupant. (See [Section 5.1.1](#) > Reduction Factor)

## **OCCMODEL**

The name of the 3D model used to represent the Occupant. (See [Section 5.1](#) > 3D Model)

## **PRIORITY\_LEVEL**

The priority level assigned to this Occupant. (See [Section 5.1.1](#) > Priority Level)

## **PERSIST\_TIME(s)**

The persist time assigned to this Occupant. (See [Section 5.1.7](#) > Persist Time)

## **COLLISION\_RESPONSE\_TIME(s)**

The collision response time assigned to this Occupant. (See [Section 5.1.7](#) > Collision Response Time)

## **PROP\_SPACING(m)**

The personal distance assigned to this Occupant. (See [Section 5.1.7](#) > Personal Distance)

## **PROP\_SOCIAL\_DIST(m)**

The social distance value assigned to this Occupant. (See [Section 5.1.7](#)) This value will be **null** if social distancing is disabled for this Occupant.

## **SLOW\_FACTOR**

The slow factor assigned to this Occupant. (See [Section 5.1.7](#) > Slow Factor)



**BOUNDARY\_LAYER(cm)**

The boundary layer assigned to this Occupant. (See [Section 5.1.7](#))

**OBEY\_ONEWAY\_DOORS**

Whether this occupant will obey one-way doors. **1** if the Occupant will obey one-way restrictions, **0** if they will ignore them. (See [Section 5.1.2](#) > Ignore One-way Door Restrictions)

**ESCALATOR\_PREF**

This Occupant's selected escalator preference. (See [Section 5.1.2](#) > Escalator Preference)

**LOCAL\_QUEUE\_TIME\_FACTOR**

The Current Room Queue Time factor applied to this Occupant. (See [Section 5.1.4](#) > Current Room Queue Time)

**LOCAL\_TRAVEL\_TIME\_FACTOR**

The Current Room Travel Time factor applied to this Occupant. (See [Section 5.1.4](#) > Current Room Travel Time)

**TAIL\_TIME\_FACTOR**

The Global Travel Time factor applied to this Occupant. (See [Section 5.1.4](#) > Global Travel Time)

**CURR\_DOOR\_PREF(%)**

The Current Door Preference applied to this Occupant. (See [Section 5.1.4](#) > Current Door Preference)

**DIST\_TRAVELLED\_DOUBLE\_DIST(m)**

The Current Room Distance Penalty applied to this Occupant. (See [Section 5.1.4](#) > Current Room Distance Penalty)

**REQUIRES\_ASSISTANCE**

Whether this Occupant requires assistance to move. **0** if Occupant can move on their own, **1** if they cannot. ([Section 5.1.2](#) > Require Assistance to Move)

**INIT\_ORIENT(deg)**

The Occupant's initial orientation when created. (See [Section 5.1.2](#) > Initial Orientation)

**ELEVATOR\_WAIT\_TIME(s)**

The Elevator Wait Time applied to this Occupant. (See [Section 5.1.4](#) > Elevator Wait Time)

**ATTRACTOR\_SUSCEPTIBILITY**

The Attractor Susceptibility applied to this Occupant while the Occupant is moving. (See

[Section 5.1.2](#) > Attractor Susceptibility (Seeking))**ATTRACTOR\_SUSCEPTIBILITY\_IDLE**

The Attractor Susceptibility applied to this Occupant while the Occupant is idle. (See [Section 5.1.2](#) > Attractor Susceptibility (Idle))

**16.6.2. Occupant Parameters JSON File**

The `filename_occupant_params.json` file provides the same data as the `filename_occupant_params.csv` file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```
{
  "occupantId": {
    "id": value,
    "name": value,
    "startTime": value,
    "profile": value,
    "behavior": value,
    "maxVelocity": value,
    "shape": {
      "type": value,
      "occId": value,
      "radius": value,
      "geometryRadius": value,
      "height": value,
      "center": {
        "x": value,
        "y": value,
        "z": value
      },
      "dir": {
        "x": value,
        "y": value,
        "z": value
      }
    },
    "occupantRadius": {
      "unit": value,
      "data": value
    },
    "geometryRadius": {
      "unit": value,
      "data": value
    },
    "height": {
```

```

    "unit": value,
    "data": value
  },
  "room": value,
  "position": {
    "unit": value,
    "x": value,
    "y": value,
    "z": value
  },
  "profileProps": {
    "speedInSmoke": {
      "mode": value,
      "customFuncVisThreshold": {
        "d_value": value,
        "d_unit": {
          "_symbol": value
        }
      }
    },
    "customFuncVisType": value,
    "customFuncVis": {
      "x": [value, ...],
      "y": [value, ...],
      "extrapolate": value
    }
  },
  "accelTime": {
    "unit": value,
    "data": value
  },
  "minSqueezeFactorConst": value,
  "occModel": value,
  "priorityLevel": value,
  "persistTime": {
    "unit": value,
    "data": value
  },
  "collisionResponseTime": {
    "unit": value,
    "data": value
  },
  "propSpacing": {
    "unit": value,
    "data": value
  },
  "slowFactor": value,
  "boundaryLayer": {
    "unit": value,

```

```

    "data": value
  },
  "obeyOnewayDoors": value,
  "escalatorPref": value,
  "localQueueTimeFactor": value,
  "localTravelTimeFactor": value,
  "tailTimeFactor": value,
  "currDoorPref": {
    "unit": value,
    "data": value
  },
  "distTravelledDoubleDist": {
    "unit": value,
    "data": value
  },
  "requiresAssistance": value,
  "initOrient": {
    "unit": value,
    "data": value
  },
  "elevatorWaitTime": {
    "unit": value,
    "data": value
  },
  "attractorSusceptibility": {
    "val": value
  },
  "attractorSusceptibilityIdle": {
    "val": value
  },
  "attractorRestrictions": {},
  "tags": [
    {
      "name": value,
      "desc": value,
      "options": [
        value,
        ...
      ],
      "d_occCounts": [
        value,
        ...
      ],
      "d_stats": {
        id: {
          "tLastTagged": value,
          "tLastUntagged": value,
          "tAccum": value
        }
      }
    }
  ]
}

```

```

    },
    {
      "id": 1,
      "name": "John",
      "startTime": 1000,
      "profile": "A",
      "behavior": "B",
      "maxVelocity": 10,
      "shape": {
        "type": "circle",
        "occId": 1
      }
    }
  ]
}

```

The JSON file contains the following JSON objects:

### **occupantId**

An object, keyed by Occupant ID, representing a single Occupant's initial Parameters.

Each Occupant entry contains the following key/value entries:

#### **id**

The Occupant's ID.

#### **name**

The Occupant's user-defined name.

#### **startTime**

See the *start time* CSV column.

#### **profile**

See the *profile* CSV column.

#### **behavior**

See the *behavior* CSV column.

#### **maxVelocity**

See the *maxVel* CSV column.

#### **shape**

An Object containing data about this Occupant's shape. This Object contains the following keys:

##### **type**

See the *shape* CSV column.

##### **occId**

The Occupant's ID.

**radius**

The radius of the shape.

**geometryRadius**

The radius of the shape's geometry.

**height**

The height of the shape.

**center**

The relative center of the shape.

**dir**

The relative forward direction of the shape.

**room**

See the *room* CSV column.

**position**

An object representing the starting position of the Occupant. See the *X*, *Y*, and *Z* CSV columns.

**speedInSmoke**

An object representing the Occupant's reaction to smoke. See the *speedInSmoke(m/s)* CSV column. The JSON format provides a little easier access to this data. This object contains the following keys:

**mode**

The selected Speed in Smoke mode. (See [Section 5.1.7](#) > Speed in Smoke)

**customFuncVisThreshold**

The maximum visibility value for visibility to affect the Occupant's speed.

**customFuncVisType**

Whether the function defined for speed in smoke is to be used as a speed function, or as a speed modifier function.

**customFuncVis**

The X and Y values of the given speed in smoke function.

**extrapolate**

Whether this PieceWise function should be extrapolated (NOTE: This is always false for

speed in smoke functions.)

**accelTime**

See the *ACCEL\_TIME(S)* CSV column.

**minSqueezeFactorConst**

See the *MIN\_SQUEEZE\_FACTOR\_CONST* CSV column.

**occModel**

See the *OCCMODEL* CSV column.

**priorityLevel**

See the *PRIORITY\_LEVEL* CSV column.

**persistTime**

See the *PERSIST\_TIME* CSV column.

**collisionResponseTime**

See the *PERSIST\_TIME(s)* CSV column.

**propSpacing**

See the *PROP\_SPACING(m)* CSV column.

**propSocialDist**

See the *PROP\_SOCIAL\_DIST(m)* CSV column.

**slowFactor**

See the *SLOW\_FACTOR* CSV column.

**boundaryLayer**

See the *BOUNDARY\_LAYER(cm)* CSV column.

**obeyOnewayDoors**

See the *OBEY\_ONEWAY\_DOORS* CSV column.

**escalatorPref**

See the *ESCALATOR\_PREF* CSV column.

**localQueueTimeFactor**

See the *LOCAL\_QUEUE\_TIME\_FACTOR* CSV column.

**localTravelTimeFactor**

See the *LOCAL\_TRAVEL\_TIME\_FACTOR* CSV column.

**tailTimeFactor**

See the *TAIL\_TIME\_FACTOR* CSV column.

**currDoorPref**

See the *CURR\_DOOR\_PREF(%)* CSV column.

**distTravelledDoubleDist**

See the *DIST\_TRAVELLED\_DOUBLE\_DIST(m)* CSV column.

**requiresAssistance**

See the *REQUIRES\_ASSISTANCE* CSV column.

**initOrient**

See the *INIT\_ORIENT(deg)* CSV column.

**elevatorWaitTime**

See the *ELEVATOR\_WAIT\_TIME(s)* CSV column.

**attractorSusceptibility**

See the *ATTRACTOR\_SUSCEPTIBILITY* CSV column.

**attractorSusceptibilityIdle**

See the *ATTRACTOR\_SUSCEPTIBILITY\_IDLE* CSV column.

## 16.7. Interpersonal Distance

The interpersonal distance output files provide data related to how closely Occupants are spaced throughout the simulation. Reported distances in these files are measured center-to-center.

**NOTE**

Pathfinder reports reference distance using a shortest path calculation very similar to that used for agent navigation. Boundary edges in the navigation mesh such as walls and separators block the reference distance calculation, partial blockages report the reference distance after routing around the blockage, and all doors are non-blocking. To model furniture such as a desk or service counter, rooms with all doors one-way exit only can be used.



## 16.7.1. Interpersonal Distance CSV Files

### 16.7.1.1. Transient CSV File

Each row in the `filename_sd_transient.csv` (where **filename** is the name of your saved PTH file) contains transient data about a single Occupant's separation from other Occupants at a single CSV output timestep. Each row lists the closest other occupant and their separation distance. It also lists the number and ID of Occupants within 1m, 2m, and 3m.

**Table 15. Transient Interpersonal Distance CSV File Structure**

Column Name	Type	Optional	Description
Time (s)	float		Simulation time
ID	int		Unique ID for occupant
Name	text	x	Occupant name from nav view in UI
Group ID	int	x	Unique ID for occupant group, if present
Closest Occupant ID	int	x	ID of closest occupant, if present within 3m
Closest Occupant Name	text	x	Occupant name from nav view in UI, if present within 3m
Closest Occupant Distance (m)	float	x	Distance to closest occupant, if present within 3m
Occupants Within 1m (count)	int		Number of occupants within 1 meter
Occupants Within 1m IDs	text	x	Space-separated list of occupant IDs within 1m
Occupants Within 2m (count)	int		Number of occupants within 2 meter
Occupants Within 2m IDs	text	x	Space-separated list of occupant IDs within 2m
Occupants Within 3m (count)	int		Number of occupants within 3 meter
Occupants Within 3m IDs	text	x	Space-separated list of occupant IDs within 3m

### 16.7.1.2. Accumulated CSV File

Each row in the `filename_sd_accumulated.csv` (where **filename** is the name of your saved PTH file) contains accumulated exposure data for a single Occupant. The data includes the ID and total exposure time of the Occupant who spent the longest time within reference distance of the row's Occupant, and a count and IDs of all Occupants who were within reference distance of the row's

Occupant for more than 1 and 5 minutes.

**Table 16. Accumulated Interpersonal Distance CSV File Structure**

Column Name	Type	Optional	Description
ID	int		Unique ID for occupant
Name	text	x	Occupant name from nav view in UI
Group ID	int	x	Unique ID for occupant group, if present
SD (m)	float		Distance used to calculate reference distance exposure.
Longest Occupant ID	int	x	ID of occupant of greatest exposure (longest within distance SD)
Longest Occupant Name	text	x	ID of occupant of greatest exposure (longest within distance SD)
Longest Occupant Time (s)	float	x	Time within distance SD of occupant of greatest exposure
Occupants > 60s (count)	int		Number of occupants within distance SD for >60 seconds
Occupants > 60s IDs	text	x	Space-separated list of occupant IDs within distance SD for >60 seconds
Occupants > 300s (count)	int		Number of occupants within distance SD for >300 seconds
Occupants > 300s IDs	text	x	Space-separated list of occupant IDs within distance SD for >300 seconds

## 16.7.2. Interpersonal Distance JSON Files

### 16.7.2.1. Transient JSON File

The `filename_sd_transient.json` file provides the same data as the `filename_sd_transient.csv` file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```

{
  "occupantID": {
    "time": {
      "name": value,
      "groupId": value,
      "closest": {
        "id": value,
        "name": value,
        "distance": value
      },
      "occsWithin": {
        "distance": {
          "count": value,
          "ids": [value, ...]
        },
        ...
      }
    },
    ...
  },
  ...
}

```

The JSON file is structured as an object that stores the interpersonal distance data for each occupant, keyed by the Occupant's ID. Each interpersonal distance object has data stored at each reporting time step. Each of these time step objects contains the following keys:

**name**

See the *Name* CSV Column.

**groupId**

See the *Group ID* CSV Column.

**closest**

An object containing data about the Occupant nearest to this Occupant at the given timestep.

**id**

See the *Closest Occupant ID* CSV Column.

**name**

See the *Closest Occupant Name* CSV Column.

**distance**

See the *Closest Occupant Distance (m)* CSV Column.

**occsWithin**

An object keyed by the filtered distance. Currently, this object reports 1m, 2m, and 3m distances, and uses **1.0**, **2.0**, and **3.0** as keys for the filtered data.

**count**

See the *Occupants Within **distance** (count)* CSV Columns.

**ids**

See the *Occupants Within **distance** IDs* CSV Columns.

**16.7.2.2. Accumulated JSON File**

The **filename\_sd\_accumulated.json** file provides the same data as the **filename\_sd\_accumulated.csv** file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```

{
  "occupantID": {
    "id": value,
    "name": value,
    "detectionRadius": {
      "unit": value,
      "data": value
    },
    "groupId": value,
    "longestExposure": {
      "id": value,
      "name": value,
      "time": value
    },
    "exposureOver60": {
      "exposed": value,
      "count": value,
      "ids": [value, ...]
    },
    "exposureOver300": {
      "exposed": value,
      "count": value,
      "ids": [value, ...]
    }
  },
  ...
}

```

The file is structured as an object keyed by Occupant ID. Each value represents a single Occupant's accumulated interpersonal distance data. The values contain the following keys:

#### **id**

See the *ID* CSV Column.

#### **name**

See the *Name* CSV Column.

#### **detectionRadius**

An object containing the radius used for detection.

#### **unit**

The unit of the radius.

**data**

The scalar value of the detection radius. See the *SD (m)* CSV Column.

**groupId**

See the *Group ID* CSV Column.

**longestExposure**

An object containing data about the Occupant that this Occupant was exposed to the longest during the simulation.

**id**

See the *Longest Occupant ID* CSV Column.

**name**

See the *Longest Occupant Name* CSV Column.

**time**

See the *Longest Occupant Time (s)* CSV Column.

**exposureOver60**

An object containing data about Occupants that this Occupant was exposed to for more than 60 seconds during the simulation.

**exposed**

True if this Occupant was exposed to any other Occupants for more than 60 seconds. False otherwise.

**count**

See the *Occupants > 60s (count)* CSV Column.

**ids**

See the *Occupants > 60s IDs* CSV Column.

**exposureOver300**

An object containing data about Occupants that this Occupant was exposed to for more than 300 seconds during the simulation.

**exposed**

True if this Occupant was exposed to any other Occupants for more than 300 seconds. False otherwise.

**count**

See the *Occupants > 300s (count)* CSV Column.

**ids**

See the *Occupants > 300s IDs* CSV Column.

## 16.8. Occupant Summary

### 16.8.1. Occupant Summary CSV File

Each row in the `filename_occupants.csv` file (where `filename` is the name of your saved PTH file) represents data for a single Occupant and provides the following columns:

**id**

A unique integer identifier for the occupant assigned by the simulator.

**name**

The name given to the occupant in the user interface.

**exit time**

The time at which the occupant went through an exit door.

**active time**

The amount of time the occupant was actively seeking a location in the model, such as a waypoint or exit.

**congestion time total**

Total time the occupant spent moving at an average velocity less than the **Congestion Velocity** as specified in the simulation parameters (see [Section 15.1.2](#)).

**congestion time max continuous**

The maximum continuous amount of time the occupant spent at an average velocity less than the **Congestion Velocity**.

**level congestion time**

Time spent by the occupant moving at an average velocity less than **Congestion Velocity** on level surfaces.

**stair congestion time**

Time spent by the occupant moving at an average velocity less than **Congestion Velocity** on stairs.

**ramp congestion time**

Time spent by the occupant moving at an average velocity less than **Congestion Velocity** on ramps.

**start time**

The time at which the occupant was generated. For occupants pre-seeded in the model, this is always 0. For occupants generated by an occupant source, this could be any time.

**finish time**

The time at which the occupant left the simulation for any reason.

**distance**

The total distance the occupant travelled during the simulation.

**num triggers used**

The total number of triggers used by the occupant during the simulation.

**trigger time**

The amount of time spent by the occupant using triggers.

**num occ targets used**

The total number of unique reserved Occupant Targets where the occupant waited during the simulation.

**occ target time**

The total amount of time the occupant waited at a reserved Occupant Target.

**last\_goal\_started time**

The time at which the occupant started their last behavior action.

**safe time total**

The total amount of time the occupant spent in a refuge room.

**tag:safe time last**

The last time the occupant entered a refuge room.

**untag:safe time last**

The last time the occupant exited a refuge room.

The remaining columns are dedicated to tags that might have been applied to occupants during the course of the simulation. For each tag, there are three columns in this order, where `tag_name` is the name of the tag (see [Section 5.8](#)).



**tag\_name total**

The total amount of time the occupant had this tag.

**tag:tag\_name time last**

The last time the occupant had this tag applied to them.

**untag:tag\_name time last**

The last time the occupant had this tag removed from them.

## 16.8.2. Occupant Summary JSON File

The `filename_occupants.json` file (where `filename` is the name of your saved PTH file) provides the same data as the `filename_occupants.csv` file in a different format, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```
{
  "occupantID": {
    "id": value,
    "name": value,
    "exitTime": value,
    "activeTime": value,
    "congestionTimeTotal": value,
    "congestionTimeMaxContinuous": value,
    "congestionTimeLevel": value,
    "congestionTimeStair": value,
    "congestionTimeRamp": value,
    "startTime": value,
    "finishTime": value,
    "distance": value,
    "numTriggersUsed": value,
    "triggerTime": value,
    "numTargetsUsed": value,
    "targetTime": value,
    "tags": {
      "tagName": {
        "usageTime": value,
        "lastAdded": value,
        "lastRemoved": value
      },
      ...
    }
  },
  ...
}
```

The file is structured as an object keyed by Occupant ID. Each value represents a single Occupant's accumulated interpersonal distance data. The values contain the following keys:

**id**

See the *ID* CSV Column.

**name**

See the *Name* CSV Column.

**exitTime**

See the *exit time* CSV Column.

**activeTime**

See the *active time* CSV Column.

**congestionTimeTotal**

See the *congestion time total* CSV Column.

**congestionTimeMaxContinuous**

See the *congestion time max continuous* CSV Column.

**congestionTimeLevel**

See the *level congestion time* CSV Column.

**congestionTimeStair**

See the *stair congestion time* CSV Column.

**congestionTimeRamp**

See the *ramp congestion time* CSV Column.

**startTime**

See the *start time* CSV Column.

**finishTime**

See the *finish time* CSV Column.

**distance**

See the *distance* CSV Column.

**numTriggersUsed**

See the *num attractors used* CSV Column.

**triggerTime**

See the *attractor time* CSV Column.

**numTargetsUsed**

See the *num occ targets used* CSV Column.

**targetTime**

See the *occ target time* CSV Column.

**tags**

An object containing information about all tags that were applied to this Occupant throughout the simulation

**tagName**

The defined name of the tag.

**usageTime::**

See the <Occupant Summary CSV File> > *tag\_name total* column.

**lastAdded::**

See the <Occupant Summary CSV File> > *tag:safe time last* column.

**lastRemoved::**

See the <Occupant Summary CSV File> > *untag:tag\_name time last* column.

## 16.9. Occupant History

The Occupants History files provide additional time history data for each Occupant whose profile has detailed data output enabled ([Section 5.1](#)). These files may optionally be written out to one file per Occupant, or one cumulative file by setting the **Detailed Occupant Data** option described in [Section 15.1.2](#).

### 16.9.1. Occupant History CSV File

Each row of the **filenameoccupants\_detailed.csv** and **filenameoccId\_occName.csv** files (where **filename** is the name of your saved PTH file, **occId** is the Occupant's unique ID, and **occName** is the Occupant's name) provides data for a single Occupant at a single time step, and contains the following columns of data:

**t(s)**

The output time for this data row. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog.

**id**

The integer identifier of the occupant assigned by the simulator.

**name**

The name assigned to the occupant in the user interface.

**active**

Whether the occupant is actively seeking an exit (1 if they are seeking an exit and 0 if not).

**x(m), y(m), z(m)**

The 3D location of the occupant.

**v(m/s)**

The velocity of the occupant.

**distance(m)**

The total distance the occupant has travelled.

**location**

The occupant's current room.

**terrain type**

The terrain type of the occupant's current room.

**safe**

Whether the occupant is in a refuge room.

**trigger**

The name of the trigger in use by the occupant or blank if the occupant is not using a trigger.

**occupant\_target**

If the occupant is waiting at one of their reserved Occupant Targets, this is the name of the target; otherwise, this is blank.

**last\_goal\_started**

Whether the occupant has started their last behavior action.

**report\_refuge\_reached**

Whether an occupant with a **Goto Refuge Room** action has reached their refuge room (see [Section 5.3.3](#)).

### **report\_wait\_until\_end**

Whether an occupant with a **Wait Until Simulation End** action has started waiting (see [Section 5.3.3](#)).

In addition to the previous columns, there is also one for each tag that might have been applied to the occupants. **0** indicates that the tag is not applied to the occupant, and **1** indicates that it is (see [Section 5.8](#)).

For occupants with detailed data output enabled, this data may optionally be written to one output file or one file per occupant. This preference is controlled by the **Detailed Occupant Data** option as described in [Section 15.1.2](#).

## **16.9.2. Occupant History JSON File**

The **filenameoccupants\_detailed.json** and **filenameoccId\_occName.json** files (where **filename** is the name of your saved PTH file, **occId** is the Occupant's unique ID, and **occName** is the Occupant's name) provide the same data as the respective CSV file, just in a different format, shown below. These files are only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```

{
  "occupantID": {
    "timeStep": {
      "name": value,
      "isActive": value,
      "position": {
        "x": value,
        "y": value,
        "z": value
      },
      "velocity": {
        "x": value,
        "y": value,
        "z": value,
        "magnitude": value
      },
      "distance": value,
      "location": value,
      "terrainType": value,
      "trigger": value,
      "target": value,
      "tagsApplied": [ value, ... ]
    },
    ...
  },
  ...
}

```

The file is structured as an object keyed by Occupant ID. The object under an Occupant's ID contains all of the detailed output data for that Occupant over the course of the simulation. This data is presented as a series of objects, keyed by the time step value for that data. Each time step object contains data under the following keys:

**name**

See the *name* CSV Column.

**isActive**

See the *active* CSV Column.

**position**

See the *x(m)*, *y(m)*, *z(m)* CSV Column.

**x**

The x component of the Occupant's position.

**y**

The y component of the Occupant's position.

**z**

The z component of the Occupant's position.

**velocity**

See the *v(m/s)* CSV Column.

**x**

The x component of the Occupant's velocity.

**y**

The y component of the Occupant's velocity.

**z**

The z component of the Occupant's velocity.

**magnitude**

The magnitude of the Occupant's velocity.

**distance**

See the *distance(m)* CSV Column.

**location**

See the *location* CSV Column.

**terrainType**

See the *terrain type* CSV Column.

**trigger**

See the *trigger* CSV Column.

**target**

See the *occupant\_target* CSV Column.

**tags**

A string array of the tag names that were active on the Occupant during this time step.

## 16.10. Groups Output

The groups output files provide information about groups created before and during the simulation.

### 16.10.1. Groups Output CSV File

Each row in the `filename_groups.csv` file (where `filename` is the name of your saved PTH file) provides the following data columns for a single group:

**Table 17. Groups Output CSV File Structure**

Column Name	Type	Optional	Description
Group ID	int		Unique ID for group
Member IDs	text		Space separated list of numeric group member IDs
Group Name	text	x	For groups created prior to running the simulation, this column lists the group name from the tree view
Template	text	x	For groups created at simulation runtime, this column lists the name of the template (group type) used to create the group

### 16.10.2. Groups Output JSON File

The `filename_groups.json` file (where `filename` is the name of your saved PTH file) provides the same data as the CSV file, with the addition of a little bit more detailed output on the behavior of groups during the simulation. The main benefit of this file is the addition of detailed membership data for each group, shown in the format description below. This file is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).



```

{
  "groupId": {
    "id": value,
    "name": value,
    "template": value,
    "members": {
      "timeStep": [
        occupantId,
        ...
      ],
      ...
    }
  },
  ...
}

```

The file is structured as an object keyed by group ID. Each object under a group ID contains the following data:

#### **id**

See the *id* CSV Column.

#### **name**

See the *name* CSV Column.

#### **template**

See the *template* CSV Column.

#### **members**

This data differs slightly from the CSV data. This key provides an object keyed by time step value. For each time step, a list of occupant IDs is provided. This list represents all of the members that were a part of this group at the give time step.

## 16.11. Triggers History

The Trigger History files provide time history data for each Trigger in the simulation.

### 16.11.1. Triggers History CSV File

Each row of the `filename_triggers.csv` file (where **filename** is the name of your saved PTH file) represents a single time step during the simulation. The first column is the time for the data row. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog. The following columns are provided for each Trigger in the simulation:

### Trigger Usage

The number of occupants using *Trigger* at the time step.

## 16.11.2. Triggers History JSON File

The `filename_triggers.json` file (where `filename` is the name of your saved PTH file) provides the same usage data as the CSV file, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```
{
  "triggerName": {
    "timeStep": value,
    ...
  },
  ...
}
```

This file is structured as an object keyed by Trigger name. Under the key for a given Trigger's name, another object is provided that is keyed by time step. For each time step, a value is provided that matches the corresponding usage value in the CSV file.

## 16.12. Occupant Target History

The Occupant Target History files provide time history data for each Occupant Target in the model.

### 16.12.1. Occupant Target CSV File

Each row of the `filename_occtargets.csv` (where `filename` is the name of your saved PTH file) represents a single time step. The first column is the time for the data row. The frequency of output is controlled by the **Data Output Freq.** box in the **Simulation Parameters** dialog. The following columns are provided for each Occupant Target in the simulation:

#### ***OccTarget reserved\_by***

The name of the occupant who has reserved *OccTarget* and is actively waiting at it.

#### ***OccTarget reserved\_by\_id***

The ID of the occupant who has reserved *OccTarget* and is actively waiting at it.

#### ***OccTarget in\_use***

Whether the occupant who has reserved *OccTarget* is currently waiting at it. The value is **1** if the occupant is waiting or **0** if not.

### 16.12.2. Occupant Target JSON File

The `filename_occtargets.json` (where `filename` is the name of your saved PTH file) provides the same data as the CSV file, shown below, and is only written if **Write Json Output Files** is enabled (see [Section 15.1.2](#)).

```
{
  "targetName": {
    "reservedBy": {
      "timeStep": value,
      ...
    },
    "reservedById": {
      "timeStep": value,
      ...
    },
    "using": {
      "timeStep": value,
      ...
    }
  },
  ...
}
```

This file is structured as an object keyed by Occupant Target name. An object is provided for each Occupant Target that provides time history data about that target. The data objects are structured as objects keyed by the time step. Each object contains the following keys:

#### **reservedBy**

See the *OccTarget reserved\_by* CSV column.

#### **reservedById**


See the *OccTarget reserved\_by\_id* CSV column.

#### **using**

See the *OccTarget in\_use* CSV column. The JSON value will be a boolean value, rather than an integer.

## 16.13. 3D Results

Included with Pathfinder is the Pathfinder Results Viewer for visualizing 3D results. This software can be used to visualize results from both Pathfinder and the Fire Dynamics Simulator (FDS) from the National Institute of Science and Technology (NIST).

By default, the Pathfinder Results Viewer starts automatically when a Pathfinder simulation has finished. To start the results viewer manually either click **View 3D Results**  in main toolbar or in the **Results** menu click **View Results**.

For more information on using the *Pathfinder Results Viewer*, refer to the ([Pathfinder Results User Manual, n.d.](#)).

# Chapter 17. Troubleshooting

Errors, warnings, mesh connectivity, and other issues can prevent the simulation from running or cause occupants to get stuck. Errors and warnings often provide a tip on fixing the issue. The following sections offer solutions to common problems you might face while using Pathfinder.

## 17.1. Occupants cannot reach their goals, or they are getting stuck

Check the connectivity of the mesh. Check how various components are connected in the model to debug simulation errors or to ensure model validity.

To determine why a specific occupant cannot reach its next goal:

1. Right-click the room containing the occupant, and click **Select Connected Components** from the right-click menu.
2. From the drop-down box in the **Select Connected Components** dialog, we choose **Entire graph** and click **OK**. (This traces model connectivity to help determine where a door or other connection is missing.)


Pathfinder will highlight the entire graph of components touching the initial selection. We can then inspect the selected components to find out where they are disconnected from the occupant's goal.

For finer inspection in a highly connected model, select only immediately adjacent components in the **Select Connected Components** dialog.

Sometimes occupants become "stuck" even on a properly connected mesh, preventing a proper simulation run. There can be many causes of this problem and we do everything we can to prevent it, but it does happen. If occupants are becoming stuck in tight spaces, consider the following steps to resolve the problem:

1. Alter the navigation mesh in the area where occupants are becoming stuck. This can be especially useful if the area was originally extracted from imported geometry. Sometimes simply re-drawing the area using the drawing tools can fix the issue.
2. Slightly increase the occupants' **Interpersonal Distance** in their profile.
3. Slightly reduce the occupants' **Reduction Factor** in their profile.
4. Switch to the SFPE simulation mode. This mode uses the simplest set of assumptions and is not as susceptible to the geometric and movement irregularities that can cause occupants to become stuck.

## 17.2. Warnings and errors in the navigation tree

Pathfinder shows warnings and errors in the Navigation Tree to help debug potential errors in the simulation. These warnings appear as a small exclamation mark imposed over an occupant or record, . By hovering the mouse over the warning, a more detailed description of the problem can be determined. For convenience, if an object has problems, the warning marker is propagated up through the navigation tree to its ancestor groups in order to quickly identify problems in the model.

Some common warnings include:

- Warning if stairs are not connected to rooms at both ends.
- Warning if stairs or doors overlap along an edge.
- Warning if stairs or doors create a non-manifold topology (e.g. a stair connects to the inside of a room rather than an outer wall or connects to two rooms on one of its ends).

Problematic objects can be quickly selected by right clicking a group in the tree that has a warning or error icon, and selecting either **Select Errors** or **Select Warnings** from the right-click menu. In addition, if the warning on a component indicates that it interferes or overlaps with other components, the objects with which it interferes can be quickly selecting by right-clicking the object with the warning and selecting **Select conflicting components** from the right-click menu.

As a general note, if the warning "Edge is adjacent to more than 2 triangles" appears when simulating, an option to click **Cancel** appears to highlight the navigation components causing the warning.

## 17.3. Find objects related to a specific object

In Pathfinder, there are several objects that can refer to other objects in the model. For instance, an occupant can refer to both a profile and a behavior, a behavior can refer to exits, the goto elevators action can refer to elevators, etc. There are times when it is useful to know what objects are referring to another. For instance, it may be important to know which occupants are using a particular behavior. To do this, right click the behavior (or other in-use object), and from the right-click menu, click **Select Referencing Objects**. This will highlight all objects currently using that behavior.

## 17.4. System memory issues

Some models may require significant system memory to run, either because the navigational model is complex with many doors/connections, etc, or simply because there are tens of thousands of occupants. By default, Pathfinder will use up to half of system memory to run the

user interface and the simulation. In some cases, this is not enough, which may result in Pathfinder crashing or the operating system becoming unresponsive while the simulation is running. In this case, it may be possible to allow Pathfinder to use more system memory. There are two ways to do this.

The first way is to run Pathfinder from the command line and specify an additional flag that indicates more memory should be allowed for Pathfinder. To do so:

1. In Windows, click the Start Menu, and then type `cmd` into the search bar. *Enter* on the keyboard to start the command prompt.
2. In the command prompt, type `cd "C:\Program Files\Pathfinder 2020"` and press *Enter*. If you installed Pathfinder to a different location, enter that path instead.
3. In the command prompt, type `pathfinder -J-DXmx16000m`, where `16000` is the amount of system memory to allow in megabytes. In most cases this amount should not exceed 95% of actual system memory.

**NOTE**

If you specify an amount that is too high, there may not be enough resources for the graphical features of Pathfinder, which may lead to crashes or other issues.

For a more permanent way to specify a higher memory limit you can perform the following:

1. If Pathfinder is not already pinned to the taskbar, in Windows click the Start Menu and type `Pathfinder` into the search bar. Then right-click the correct Pathfinder executable and select **Pin to taskbar**.
2. In the Windows task bar, find the pinned Pathfinder application and right-click it.
3. In the sub-menu that appears, right-click the Pathfinder executable and select **Properties**.
4. At the end of the **Target** field, type a space on the keyboard and then `-J-DXmx16000m`. The shortcut dialog will look similar to [Figure 180](#).
5. Press *OK* and then start Pathfinder using the pinned shortcut.

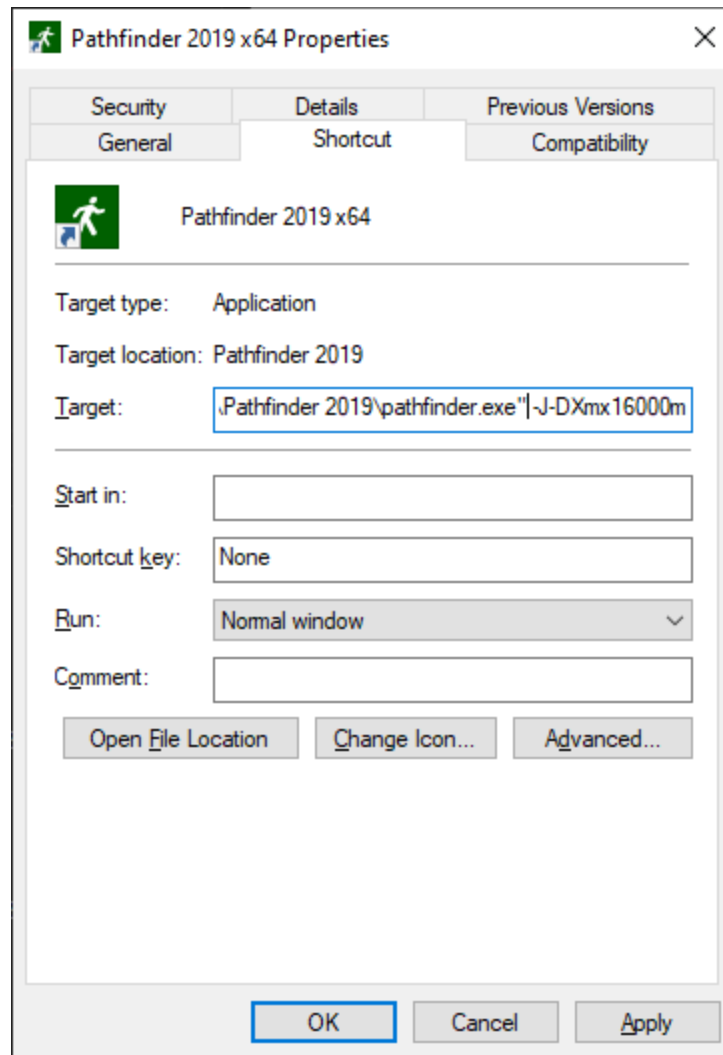


Figure 180. Example Shortcut Dialog

## 17.5. Issues with custom avatars

If you experience issues displaying a custom occupant or vehicle avatar, see [Section 4.5.5](#) for help troubleshooting.

## 17.6. Video display problems and crashes on startup

Pathfinder utilizes many advanced graphics card features in order to provide accelerated and enhanced display of models.

Sometimes these graphics features combined with certain display drivers can cause display issues or crashes on startup. The first step in these cases is to ensure you have the latest operating system updates and graphics drivers installed. If you do and you still have display problems or crashing on startup, you can start Pathfinder in Safe Mode, which disables several graphics



features. To start in Safe Mode:

1. Open a command prompt by opening the start menu and typing `cmd`. Then press *Enter*.
2. In the command prompt, navigate to the Pathfinder install folder by typing `cd "C:\Program Files\Pathfinder 2024"`.
3. Start Pathfinder in Safe Mode by typing `pathfinder -DSafeMode`.
4. If Pathfinder starts up successfully, you can see which display properties were changed. Under the **File** menu, click **Preferences**, and then click on the **Rendering** tab.
5. You can enable each item one at a time to see which ones caused the issue.

**NOTE**

Disabled safe-mode properties will remain disabled the next time you run Pathfinder, even if not in Safe Mode. If you've re-enabled some of those options, running in Safe Mode again will disable the options again.

If you encounter display problems or crashes, please let us know the make/model of your video card and what video driver you are using, even if Safe Mode fixes your issues. That will help us improve the software in future updates.

## Appendix A: Pre-defined Speed Profiles

Pre-defined speed profiles are included in Pathfinder as a convenient starting point [Table 18](#). All Pathfinder models are initialized with a "Default" profile. Additional bundles of profiles are included in the lib/profiles subdirectory of the Pathfinder install location.

**Table 18. Summary of the profiles included with Pathfinder.**

Profile Source	Speed	Speed-Density Relation	Speed Factor Stairs
Default	Constant (1.19 m/s)	SFPE (built-in)	SFPE (built-in)
imo_speed_profiles.pli b	Uniform Distribution	From Table	Constant
fruin_speed_profiles.pl ib	Normal Distribution	SFPE (built-in)	From Table

The "Default" profile present in new Pathfinder models is based on data in the SFPE's *Engineering Guide to Human Behavior in Fire* ([SFPE 2019](#)). This profile uses a constant speed of **1.19 m/s** and relies on Pathfinder's defaults for speed-density relation and slope-based speed factor on stairs.

### A.1. Profiles from the IMO's Revised Guidelines for Passenger Ships

The profile bundle `imo_speed_profiles.plib` is based on data from the IMO's guidelines ([IMO 2007](#)).

This source gives ranges of walking speeds on flat terrain for each of 12 different population groups. This is implemented directly in the Pathfinder profiles as per-profile uniform distributions. The bundle of profiles in Pathfinder includes only the ten population groups that do not correspond to crew members.

A table is given for speed-density relation on flat terrain in terms of "initial specific flow" and a reference "initial speed". These two values were used to create a speed-density factor table shared by all profiles in this set.

#### NOTE

This speed-density relation differs from the built-in speed-density relation primarily in very dense situations, clamping the max density based slowdown at **0.1 m/s** rather than **0.15 m/s**.

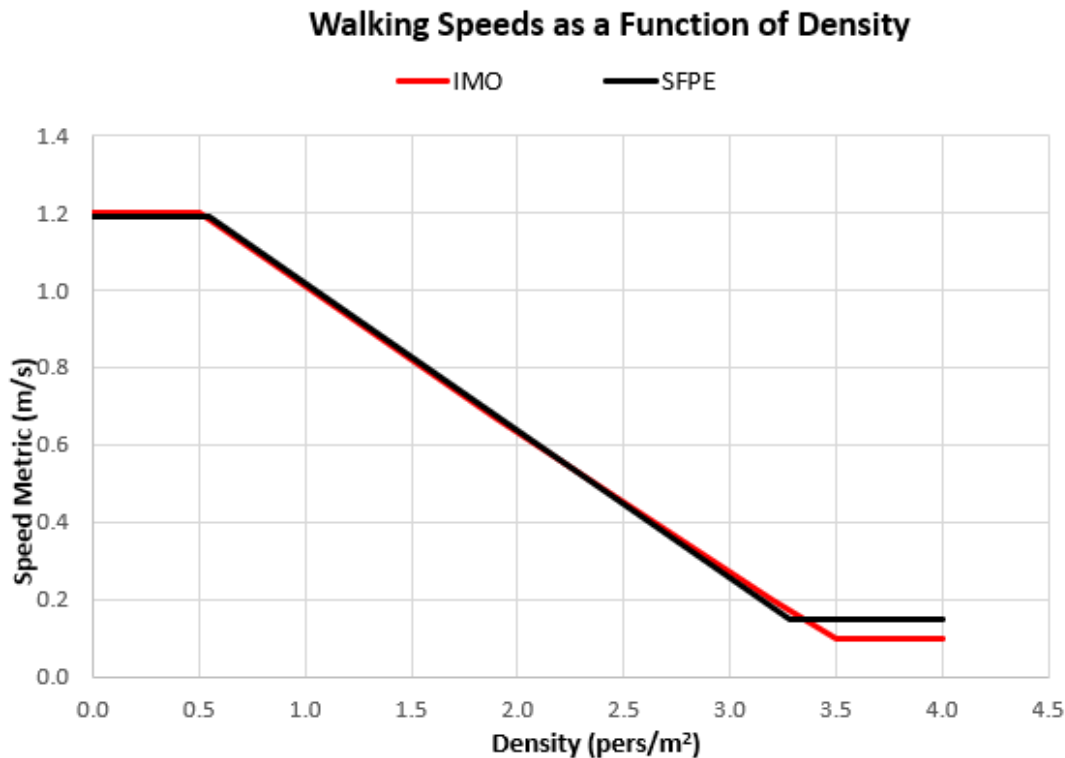


Figure 181. Graph of Walking Speeds as a Function of Density

The revised guidelines also define ranges of walking speeds up and down stairs for all population groups. Since Pathfinder requires a speed factor for stairs rather than a range, these values were approximated by calculating a constant scaling factor based on the average of the flat terrain range and the average of the range of speed on stairs.

## A.2. Profiles from Fruin's Pedestrian Planning and Design

The profile bundle `fruin_speed_profiles.plib` is based on data from John Fruin's *Pedestrian Planning and Design* (Fruin and Strakosch 1987).

This source describes six population groups based on age and gender, and only gives a single normal distribution of walking speeds that is used for all population groups. This is implemented in Pathfinder by re-using the same normal distribution for the six population groups and an "Average All" aggregate profile.

The built-in SFPE-based speed-density relation is used for all profiles in this set. The built-in SFPE-based speed-density relation differs from the original data primarily in the locations where the min and max walking speeds are clamped. The original data was in terms of an absolute

measurement of walking speed across multiple people rather than a value that could be normalized to a specific speed (1.2 m/s) and it went to zero at high densities which is undesirable for simulation because it could cause issues with occupants becoming permanently stuck. The following chart compares the original data to the relation used for this profile in Pathfinder.

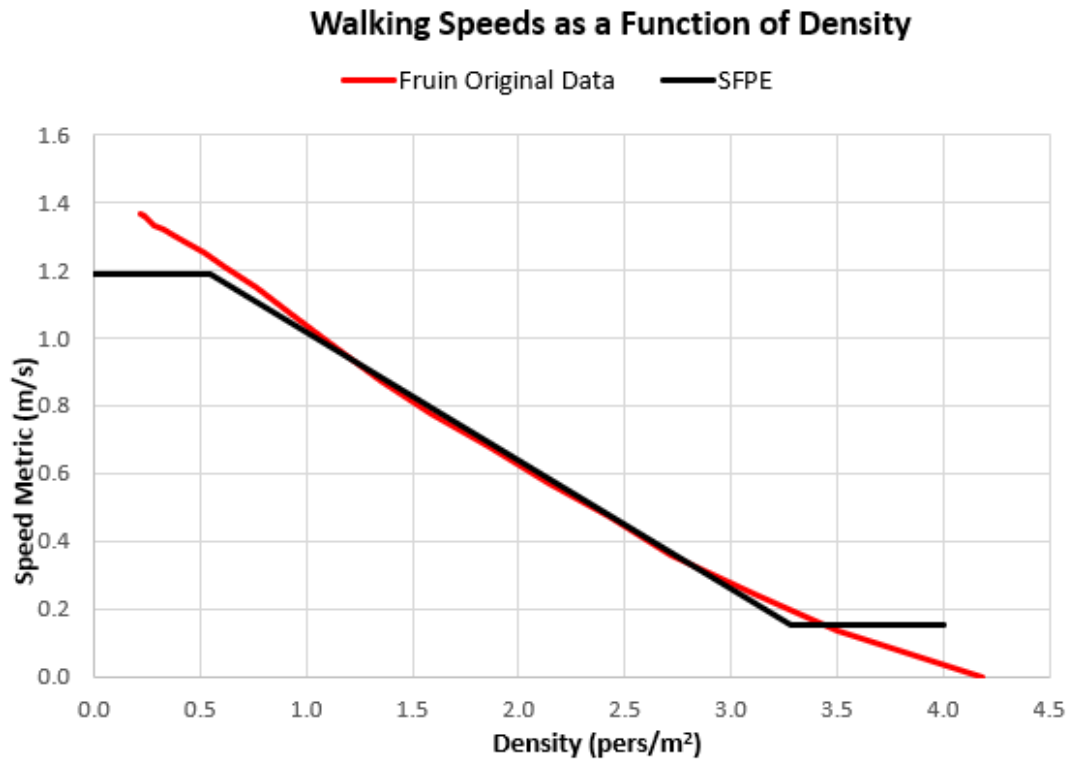


Figure 182. Graph of Walking Speeds as a Function of Density from Fruin

*Pedestrian Planning and Design* (Fruin and Strakosch 1987) gives stairway speeds for two slopes, identified as an "indoor" 32 degree stair (7" / 11.25") and an "outdoor" 27 degree stair (6" / 12"). Speeds are given for the up direction and the down direction and they are individualized for the six population groups and the aggregate group. These speeds were converted to tables of speed modifier fractions. For each slope and direction, the speed on stairs was converted to a fraction of max speed by dividing by the average of the flat walking speed (1.2 m/s) used by all population groups. This made it possible to create tables of speed modifiers unique to each population group and travel direction that contained two data points. For stairs with slopes more or less than the given 32 and 27 degree stairs, the factor is constant (i.e. not extrapolated).

# Appendix B: Avatar and Animation File Format

When Pathfinder imports a custom avatar or animation (see [Section 4.5](#) and [Section 4.6](#)), a JSON file is also created for each avatar and animation. Avatar JSON files are placed in `%APPDATA%/Pathfinder/models/md5` and animation JSON files are placed in `%APPDATA%/Pathfinder/models/anim`s. A unified JSON file format is used for both avatars and animations and includes information such as a transformations (e.g. scaling, rotation, and offset), natural speeds for move animations, etc. One of these files can contain information about the avatar, animations, or both.

## B.1. Avatar JSON File

An avatar JSON file contains information about a single avatar, including the base avatar file, transformation information, custom animation information, etc. When Pathfinder imports a custom avatar, it will automatically create this file and place it in `%APPDATA%/Pathfinder/models/md5`, along with other supporting files, such as the imported FBX or GLB file. This file can be manually modified in a text editor to fix avatar issues, such as incorrect animations, scaling, orientation, etc. (see [Section 4.5.5](#)).

In order to be recognized by Pathfinder and 3D Results, an avatar file must have the following traits.

- The avatar file must be located in a sub-directory of either of the following:
  - `%PROGRAMDATA%/Pathfinder/models/md5` - shared with all users on the computer.
  - `%APPDATA%/Pathfinder/models/md5` - only available to the current user on the computer. This is the location for imported avatars.
- The avatar file must contain a [mesh](#) section.

An avatar file can optionally contain an [anim](#)s section. If it does, the listed animations can either be ones that are only available to the file's avatar or animations that override those in the animation database. For instance, if an avatar appears to be skating when using one of the move animations (see [Section 4.6.3.2](#)), the move animation can be overridden in the avatar's JSON file to specify a different natural speed than that in the animation database.

The following example illustrates the JSON avatar file format. In this example, the idle animation specifies a UUID that is not used in any animation database (shipped or custom). This means that this idle animation would only be available to this avatar. The move animation, however, shares a UUID with one of the move animations in the shipped animation database (`pathfinder_install_folder/lib/models/md5/anim/anim_database.json`), indicating that the

database move animation is being overridden with this one for this avatar.

### Example Avatar JSON File

```
{
  "mesh":
  {
    "path": "Ch23_nonPBR.fbx"
  },
  "anim":
  [
    {
      "type": "idle",
      "uuid": "B32615B3-DD16-4D7B-ABDC-7CA79505E626",
      "tags": ["sit", "hands-in-lap"],
      "path": "clips/Sitting Idle.fbx",
      "retargetTo": "other",
      "retargetSource": "meshes/Ch36.fbx"
    },
    {
      "type": "move",
      "uuid": "C15E0D93-BD70-43EF-9C47-E29CBD0691CB",
      "tags": ["upright", "default"],
      "dir": ["forward"],
      "clips":
      [
        {
          "path": "walk1.md5anim",
          "uuid": "98844734-D888-40CB-A9EF-258458498963",
          "frameOffset": 42,
          "naturalSpeed": 0.6,
          "maxSpeed": 1.1,
          "retargetTo": "other",
          "retargetSource": "base_mesh.md5mesh"
        },
        {
          "path": "walk3.md5anim",
          "uuid": "8A069997-851C-4585-AEB9-C76553AAF6E0",
          "frameOffset": 20,
          "naturalSpeed": 1.5,
          "maxSpeed": 1.7,
          "retargetTo": "other",
          "retargetSource": "base_mesh.md5mesh"
        },
        {
          "path": "maleRun.md5anim",
          "uuid": "9FAD15C0-34F5-45A3-876E-F10E9FF2B8C5",
          "frameOffset": 14,
```

```

        "naturalSpeed": 2.5,
        "retargetTo": "other",
        "retargetSource": "base_mesh.md5mesh"
    }
  ]
}
]
}

```

## B.2. Example Animation JSON File

The animation **JSON** file represents a single database animation that can be used with any avatar and contains information about the animation, such as the base animation file, transformation information, movement speeds and directions, etc. When Pathfinder imports a custom animation, it will automatically create this file and place it in `%APPDATA%/Pathfinder/models/anim`s, along with other supporting files, such as the imported FBX file. In most cases, these files should not be edited directly; the Pathfinder user interface should be used instead. However, in some cases, it may be necessary to manually edit in a text editor to fix certain issues, such as unrecognized joint names (see [Section 4.5.5](#)).

While nearly identical to the avatar **JSON** file format, the animation file format has the following key differences:

- The animation file must be located in one of the following:
  - `%PROGRAMDATA%/Pathfinder/models/anim`s - shared with all users on the computer.
  - `%APPDATA%/Pathfinder/models/anim`s - only available to the current user on the computer. This is the location for imported avatars.
- The **mesh** section is ignored.
- Only a single animation should be specified in the **anim**s section.

**NOTE** Multiple animations can be specified, and 3D Results will recognize all of them, but the Pathfinder user interface only lists the first animation in the file.

The following examples illustrate the animation **JSON** file format.

The first example shows an **idle** animation that has a unique UUID across all database animations (those shipped with Pathfinder and custom animations imported using Pathfinder). As such, it will exist as a unique animation in the database.

**Animation JSON File: Example 1**

```
{
  "anim": [
    {
      "type": "idle",
      "uuid": "B32615B3-DD16-4D7B-ABDC-7CA79505E626",
      "tags": ["sit", "hands-in-lap"],
      "path": "clips/Sitting Idle.fbx",
      "retargetTo": "other",
      "retargetSource": "meshes/Ch36.fbx"
    }
  ]
}
```

The second example represents a move animation composed of multiple clips. Unlike in the first example, however, the animation's UUID (**C15E0D93-BD70-43EF-9C47-E29CBD0691CB**) matches one of the move animations in the shipped animation database (`pathfinder_install_folder/lib/models/md5/anim/anim_database.json`). This indicates that all avatars whose move animation matches the tags **default**, **upright** should use this animation instead of the one shipped with Pathfinder.



**Animation JSON File: Example 2**

```

{
  "anim": [
    {
      "type": "move",
      "uuid": "C15E0D93-BD70-43EF-9C47-E29CBD0691CB",
      "tags": ["upright", "default"],
      "dir": ["forward"],
      "clips": [
        {
          "path": "walk1.md5anim",
          "uuid": "98844734-D888-40CB-A9EF-258458498963",
          "frameOffset": 42,
          "naturalSpeed": 0.6,
          "maxSpeed": 1.1,
          "retargetTo": "other",
          "retargetSource": "base_mesh.md5mesh"
        },
        {
          "path": "walk3.md5anim",
          "uuid": "8A069997-851C-4585-AEB9-C76553AAF6E0",
          "frameOffset": 20,
          "naturalSpeed": 1.5,
          "maxSpeed": 1.7,
          "retargetTo": "other",
          "retargetSource": "base_mesh.md5mesh"
        },
        {
          "path": "maleRun.md5anim",
          "uuid": "9FAD15C0-34F5-45A3-876E-F10E9FF2B8C5",
          "frameOffset": 14,
          "naturalSpeed": 2.5,
          "retargetTo": "other",
          "retargetSource": "base_mesh.md5mesh"
        }
      ]
    }
  ]
}

```

**B.3. Avatar/Animation JSON Structure**

The following tables list the properties available in the JSON avatar and animation files.

Table 19. root Properties

Prop erty	Type	Defa  ult Value	Requ ired?	Description	Example
mesh	<a href="#">mesh</a>	{}	Yes, if the file represents an avata r.	Links to the avatar file and describes any scaling and other transformation properties.	
anim s	list of <a href="#">anim ation</a>	[]	Yes, if the file represents a datab ase anim ation. Optional if the file represents an avata r.	Lists the animations in the file.	
joint Name MapF ile	string	""	no	References an external file containing a joint name map (see <a href="#">Section B.3.3</a> ) that should be used for the referenced mesh and all animations in the file. Ignored if <a href="#">jointNameMap</a> is defined.	<a href="#">"joint_name_map.props"</a> The joint name map is in a file called <a href="#">joint_name_map.props</a> .

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
joint Name Map	map	{}	no	Defines joint name mapping (see <a href="#">Section B.3.3</a> ) that should be used for the referenced mesh and all animations in the file.	<pre>{   "pelvis.5_5": "pelvis",   "L5.6_6": "Spine" }</pre> <p>Maps the file's unrecognized joint names, <b>pelvis.5_5</b> and <b>L5.6_6</b> to known joint names, <b>pelvis</b> and <b>Spine</b>, respectively.</p>

Table 20. **mesh** Properties

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
path	string	""	yes	The relative or absolute path to the avatar file. This can be an FBX, MD5MESH, GLTF, or GLB file.	<p><b>"Ch23_nonPBR.fbx"</b></p> <p>Links to the file, <b>Ch23_nonPBR.fbx</b>, found in the same folder as this JSON file.</p>
uuid	string	""	No, but recommended	A UUID (universally-unique identifier) that uniquely identifies the mesh. This should be unique across all other avatars. While providing a UUID is recommended, if one is not provided, it will be generated automatically from other mesh properties.	<p><b>"A477B469-49B9-4BB2-ACF2-3DC807642DBF"</b></p> <p>Assigns the UUID <b>A477B469-49B9-4BB2-ACF2-3DC807642DBF</b> to the avatar.</p>

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
trans form	list of <a href="#">trans form</a>	[]	no	<p>A list of transform objects to apply to the mesh. The transform objects are combined to form a single transformation as follows:</p> <ol style="list-style-type: none"> <li>1. All scales are multiplied and then applied.</li> <li>2. All rotates are multiplied and then applied (this has the effect of applying rotations in the reverse order that they are listed).</li> <li>3. All translations are added and then applied.</li> </ol>	<pre>[{"rotate":[0,0,1,-45]},{"translate":[0.3,0.8,0]},{ "rotate":[1,0,0,90]},{"scale":1.5}]</pre> <p>Applies transforms in this order:</p> <ol style="list-style-type: none"> <li>1. Scales by <b>1.5</b>.</li> <li>2. Rotates about <code>X</code> axis <code>90&amp;deg;</code>, <code>CCW</code>.</li> <li>3. Rotates about the <code>Z</code> axis <code>-45&amp;deg;</code>, <code>CCW</code>.</li> <li>4. Translates along <b>X</b> axis <b>.3 m</b> and <b>Y</b> axis <b>.8 m</b>.</li> </ol>
impo rtOpt ions	numb er	0	no	Options that can be used to import the mesh file. The available options are specific to the type of mesh file.	0
joint Name MapF ile	string	""	no	References an external file containing a joint name map for the mesh. This overrides any joint name map specified in the <a href="#">root</a> . Ignored if <a href="#">jointNameMap</a> is defined.	See <a href="#">Table 19</a> .
joint Name Map	map	{}	no	Defines joint name mapping for the mesh. This overrides any joint name map specified in the <a href="#">root</a> .	See <a href="#">Table 19</a> .

**Table 21. transform Properties**

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
transl ate	array of 3 numb ers	[0,0,0 ]	no	Defines an offset in meters.	<code>[1, 0.5, 0]</code>  Moves along the x axis <code>1 m</code> and along y axis <code>.5 m</code> .
rotat e	array of 4 numb ers	[0,0,1, 0]	no	Defines a rotation. The first three values describe the axis (should be normalized), and the fourth value describes the angle in degrees. Rotations are always performed CCW about the axis.	<code>[0,0,1,-90]</code>  Rotates about the <code>Z</code> axis <code>-90deg</code> .
scale	numb er	1	no	Defines scaling. All scaling is performed uniformly about all axes.	<code>1.5</code>  Scales <code>1.5</code> times.

**Table 22. animation Properties**

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
type	string	"idle"	yes	<p>Describes the type of animation. Must be one of the following values:</p> <p><b>"idle"</b> Will be used when an occupant is idling.</p> <p><b>"move"</b> Will be used when the occupant is actively moving to a destination.</p> <p><b>"pivot"</b> Currently unused.</p>	<p><b>"idle"</b></p> <p>This is an idle animation (used when an occupant is stationary).</p>
tags	list of string	[]	yes	A list of tags that can be used in Pathfinder to identify the animation (see <a href="#">Section 5.1.5</a> ).	<p><b>["default", "upright"]</b></p> <p>Associates the animation with all combinations of <b>default</b> and <b>upright</b>. I.e:</p> <ul style="list-style-type: none"> <li>• default</li> <li>• upright</li> <li>• default, upright</li> </ul>

### B.3.1. Idle Animations

The following **animation** properties are also available if **animation:type** is **"idle"**.

**Table 23. animation Properties (idle-only)**

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
path	string	""	yes, if <b>type</b> = "idle"	The path to the animation file. This may be an <b>FBX</b> or <b>MD5ANIM</b> .	"Sitting Idle.fbx"  Links to the file, <b>Sitting Idle.fbx</b> , found in the same folder as this JSON file.
uuid	string	""	No, but reco mme nded	A UUID (universally-unique identifier) that identifies the animation. If the UUID is unique across all other animations, the animation exists on its own. However, if the UUID matches one in the animation database that is provided with Pathfinder, this animation overrides that in the database. While providing a UUID is recommended, if one is not provided, it will be generated automatically from other animation properties.	"A477B469-49B9-4BB2-ACF2-3DC807642DBF"  Assigns the UUID <b>A477B469-49B9-4BB2-ACF2-3DC807642DBF</b> to the animation.
fram eOffs et	numb er	0	no	Remaps the specified frame as frame <b>0</b> of the animation.	<b>5</b>  Assigns the 5th frame in the animation file as the first frame of the animation.
trans form	list of <b>trans form</b>	[]	no	Uses the same format as in the <b>mesh</b> section.	

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
retar getTo	string	""	no	<p>Indicates whether the animation needs to be retargeted. The following values are accepted:</p> <p><b>""   "none"</b></p> <p>No retargeting necessary - the animation was created exactly for the <b>mesh</b> avatar.</p> <p><b>"same"</b></p> <p>The animation was created for a different avatar mesh that has the same joint/skeletal structure as the <b>mesh</b> avatar.</p> <p><b>"self"</b></p> <p>The animation was created for a different avatar with a different joint/skeletal structure than the <b>mesh</b> avatar. The animation file itself contains the avatar for which the animation was made - this only works if the animation file is an FBX file and contains the avatar as well as the animation.</p> <p><b>"other"</b></p> <p>The animation was created for a different avatar with a different joint/skeletal structure than the <b>mesh</b> avatar. <b>retargetSource</b> specifies the file relative to the JSON file that contains the avatar for which the animation was made.</p>	<b>"self"</b>



Property	Type	Default Value	Required?	Description	Example
retargetSource	string	""	Yes, if <b>retargetTo</b> is "other"	Specifies the path to the mesh for which the animation was created if <b>retargetTo</b> was "other". This can be any of the files support by <b>mesh:path</b> .	<b>"anim/base_mesh.md5mesh"</b>  The animation was created for <b>base_mesh.md5mesh</b> .
idlePlayback	string	"randomize"	no	Defines how an idle animation will be played. Can be one of the following values:  <b>"randomize"</b> The animation will start at a random frame and repeat indefinitely.  <b>"once"</b> The animation will play once from the beginning and then hold the last frame.  <b>"repeat"</b> The animation will play from the beginning and repeat indefinitely.	<b>"repeat"</b>
jointNameMapFile	string	""	no	References an external file containing a joint name map for the animation. This overrides any joint name map specified in the <b>root</b> . Ignored if <b>jointNameMap</b> is defined.	See <a href="#">Table 19</a> .
jointNameMap	map	{}	no	Defines joint name mapping for the animation. This overrides any joint name map specified in the <b>root</b> .	See <a href="#">Table 19</a> .

### B.3.2. Move Animations

The following **animation** properties are also available if **animation:type** is "move" or "pivot".

**Table 24. animation Properties (move-only)**

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
uuid	string	""	No, but reco mme nded	A UUID (universally-unique identifier) that identifies the move animation. If the UUID is unique across all other animations, the animation exists on its own. However, if the UUID matches one in the animation database that is provided with Pathfinder, this animation overrides that in the database. While providing a UUID is recommended, if one is not provided, it will be generated automatically from other animation properties.	"A477B469-49B9-4BB2-ACF2-3DC807642DBF"
dir	list of string	[]	yes (at least 1 must be defin ed)	Defines the natural movement direction of the animation, relative to the avatar's facing direction. This can be a combination of the following: <ul style="list-style-type: none"> <li>• "forward"   "backward"</li> <li>• "left"   "right"</li> <li>• "up"   "down"</li> </ul>	"forward up"  Defines an animation that moves the occupant <b>forward</b> and <b>up</b> (e.g. up stairs).

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
clips	list of <a href="#">move_clip</a>	[]	yes (at least 1 must be defin ed)	Defines a list of individual animation clips that make up a composite animation in the specified direction. Each <a href="#">move_clip</a> in the list defines the movement in a specific speed range. All the <a href="#">move_clip</a> together define the full speed range.	See the JSON examples above (see <a href="#">Section B.2</a> ).

Table 25. [move\\_clip](#) Properties

Prop erty	Type	Defa ult Value	Requ ired?	Description	Example
path uuid fram eOffs et retar getTo retar getSo urce joint Name MapF ile joint Name Map				See the <a href="#">animation</a> properties for <a href="#">idle</a> animations.	

Property	Type	Default Value	Required?	Description	Example
naturalSpeed	number	0	no	The natural movement speed in m/s of the animation. When the occupant is moving this at this speed, the animation will be played back at 1x speed. If the occupant is moving at twice this speed, the animation will be played back at 2x speed, etc.	1.3  The animation will be played back at 1x when the occupant's speed is 1.3 m/s.
maxSpeed	number	+infinity	Yes, if this is not the last move animation clip.	The maximum occupant speed in m/s at which this animation should be used. If the occupant is moving faster than this speed, the next animation of this type is used unless this is the last of its type.	2.0  The animation will only be used if the occupant's speed is $\leq 2.0$ m/s and greater than the previous move animation's maxSpeed.

### B.3.3. Joint Name Map

If some or all of an occupant avatar's body parts do not move during animations, this may be because either the avatar file or animation file uses unknown joint names for the skinning hierarchy. To determine if this is the case, perform the following:

1. Start Pathfinder from a command prompt.
2. Create a model with an occupant and change their **3D Model** to the imported avatar.
3. In the **View** menu, select **Occupant Display** → **Show as People**.

The command prompt will display lines such as the following if there are unknown joint names:

```
[avatar.fbx] Could not determine joint type for joint, "L5.6_6"
[avatar.fbx] Could not determine joint type for joint, "l_hip_n.87_87"
[avatar.fbx] Could not determine joint type for joint, "l_knee_n.88_88"
```

If this is the case, these errors can be fixed by supplying a joint name map.

There are two ways this can be accomplished:

1. In the avatar or animation **JSON** file (depending on which file caused the errors), specify the joint name mapping using the **jointNameMap** property. See [Table 19](#). This may be useful if the name map only needs to be defined once.
2. In the avatar or animation **JSON** file, specify a joint name map file using the **jointNameMapFile** property. This may be useful if the name map needs to be used in several places. It will require creating a name map file as follows:

- a. Create a new text file in the same directory as the avatar's JSON file as described above.
- b. For each of the above errors, add a line to the file in the format, **unknown\_name=known\_name**, where **unknown\_name** is the name printed to the command prompt above, and **known\_name** is one of the known joint names as listed in [Known Joint Names](#).

If an unknown joint has no equivalent in the known list (e.g. the avatar has an extra spine joint), the mapping can be omitted from the joint name map file. In this case, the unknown joint will rotate to match its parent joint.

- c. In the JSON file, add the **jointNameMapFile** property where appropriate.

The following example is a joint name map file that would fix the errors in the above example:

```
L5.6_6=Spine
l_hip_n.87_87=L Thigh
l_knee_n.88_88=L Calf
```

In this example, **L5.6\_6** is the unknown name of the joint in the referenced file, and **Spine** is the known joint name.

### B.3.3.1. Known Joint Names

The following is one set of known joint names that can be used as **known\_name** in the joint name map file. Unless indicated with **#** or a blank line, each joint is a direct parent of the joint on the next line in the list. Either **Bip** or **Pelvis** can be the root node.

```
Bip
Pelvis
Spine
# Spine splits to 'Spine1', 'R Thigh', and 'L Thigh'

Spine1
Spine2
Spine3
```

```
Neck
# Neck splits to `Head`, `R Clavicle`, and `L Clavicle`

Head
HeadNub

R Thigh
R Calf
R Foot
R Toe0
R Toe0Nub

R Clavicle
R UpperArm
R Forearm
R Hand
# R Hand connects to `R Finger4`, `R Finger3`, `R Finger2`, `R Finger1`, and `R Finger0`

R Finger4
R Finger41
R Finger42
R Finger4Nub

R Finger3
R Finger31
R Finger32
R Finger3Nub

R Finger2
R Finger21
R Finger22
R Finger2Nub

R Finger1
R Finger11
R Finger12
R Finger1Nub

R Finger0
R Finger01
R Finger02
R Finger0Nub

L Thigh
L Calf
L Foot
L Toe0
L Toe0Nub
```

```
L Clavicle
L UpperArm
L Forearm
L Hand
# L Hand connects to `L Finger4`, `L Finger3`, `L Finger2`, `L Finger1`, and `L Finger0`

L Finger4
L Finger41
L Finger42
L Finger4Nub

L Finger3
L Finger31
L Finger32
L Finger3Nub

L Finger2
L Finger21
L Finger22
L Finger2Nub

L Finger1
L Finger11
L Finger12
L Finger1Nub

L Finger0
L Finger01
L Finger02
L Finger0Nub
```

# Bibliography

*Pathfinder Technical Reference*. n.d. 403 Poyntz Avenue, Suite B, Manhattan, KS 66502, USA: Thunderhead Engineering. <https://support.thunderheadeng.com/docs/pathfinder/latest/technical-reference-manual/>.

*Pathfinder Verification and Validation*. n.d. 403 Poyntz Avenue, Suite B, Manhattan, KS 66502, USA: Thunderhead Engineering. <https://support.thunderheadeng.com/docs/pathfinder/latest/verification-validation/>.

*Pathfinder Results User Manual*. n.d. 403 Poyntz Avenue, Suite B, Manhattan, KS 66502, USA: Thunderhead Engineering. <https://support.thunderheadeng.com/docs/pathfinder/latest/results-user-manual/>.

Bukowski, Richard W., and Fang Li. 2010. "Using Elevator In Fires." *Consulting - Specifying Engineer*, July.

Fridolf, Karl, Enrico Ronchi, Daniel Nilsson, and Håkan Frantzich. 2019. "The Representation of Evacuation Movement in Smoke- Filled Underground Transportation Systems." *Tunnelling and Underground Space Technology* 90 (April): 28–41. <https://doi.org/10.1016/j.tust.2019.04.016>.

Fruin, J.J., and G.R. Strakosch. 1987. *Pedestrian Planning and Design*. Elevator World. <https://books.google.com/books?id=vrckAQAAMAAJ>.

IBM. n.d. "Default Swing Key Bindings." Accessed November 18, 2019. <https://ibm.co/2O58h5A>.

IMO. 2007. "IMO Guidelines for Evacuation Analysis for New and Existing Passenger Ships." 4 Albert Embankment, London, Great Britain: International Maritime Organization.

Pheasant, Stephen, and Christine M. Haslegrave. 2005. *BodySpace: Anthropometry, Ergonomics and the Design of Work*. 3rd ed. CRC Press. <https://books.google.com/books?id=vrckAQAAMAAJ>.

SFPE. 2019. *SFPE Guide to Human Behavior in Fire*. 2nd ed. Springer International Publishing. <https://www.springer.com/gp/book/9783319946962>.

Still, G. Keith. 2000. "Crowd Dynamics." PhD thesis, University of Warwick. [http://gkstill.com/Support/Links/Documents/2000\\_still.pdf](http://gkstill.com/Support/Links/Documents/2000_still.pdf).