Visual Simulation Models: User Guide
COVID-19 Update

URI VOTES

Engineering for Democracy

Recommendations that will lead to future sustainable practices and tools utilized to overcome new and future obstacles in voting and election processes.

The idea of democracy is simple. Execution is complicated.

Tool: Visual Simulation Modeling - COVID-19 Update

Purpose of Tool:

A visual simulation model is a powerful tool that enables users to explore the relationship between polling place layout design and the flow of voters through the system. For new users, this guide provides step by step instructions to create a basic model of a polling place with electronic poll pads, paper ballots, and an optical scanner. This simulation can be used to identify potential bottleneck locations based on input specifications. A basic visual simulation model and all required inputs can be found in Visual_Sim.zip file. For more experienced users, additional renderings of other voting equipment can be downloaded from https://web.uri.edu/rivotes/ as .skp files and imported into Simio.

In the COVID-19 updated version of this guide, we will demonstrate how to adjust visual simulation models to portray the mandated social distancing requirements for both voters and poll workers. This guide assumes that the mandatory separation distance is at least 6 feet, however, this value can be adjusted for margins of safety or as regulations adjust over time.
.

Required inputs:
- Visual_Sim.zip
- Ballot marking data (i.e., duration) or descriptive statistics for the duration (*Optional*)
- Ballot scanning data (i.e., duration) (*Optional*)

Optional items allow the user to model a polling location using location-specific data

Instructions for Use:
Overview:

To begin the model building process, open the Visual_Sim_Template.spfx file in Simio. At the top left corner, select File, then Save As, to create a new model with a unique name. The Visual_Sim_Template is preprogrammed with all routing logic. By default, the template is configured for 4 check-in stations, 7 voting booths, and 2 ballot scanners. The first section of instructions demonstrates how to import voting equipment symbols and configure them to represent the desired polling place layout design. The second section covers how to arrange the preprogrammed server objects to represent the flow of voters. The third section discusses necessary changes to the programming logic to reflect election day parameters. The fourth and final section provides guidelines and suggestions for experienced users interested in using 'Visual_Sim_Template.spfx' to create more customized simulation models.

1. Creating the Polling Place Layout
   1.1. Adding Empty Floor Space model
      1.1.1. **Select** the *Drawing* tab in the top left corner and **Select** *Place Symbol*
      1.1.2. From the drop-down menu **Select** *Import Symbol*
      1.1.3. Navigate to the *Visual Simulation Model* folder. **Select** the file named *EMPTY Voting Area.skp*. Press **Open**.
      1.1.4. **Select the** *OK* button at the bottom right corner of the Import window.
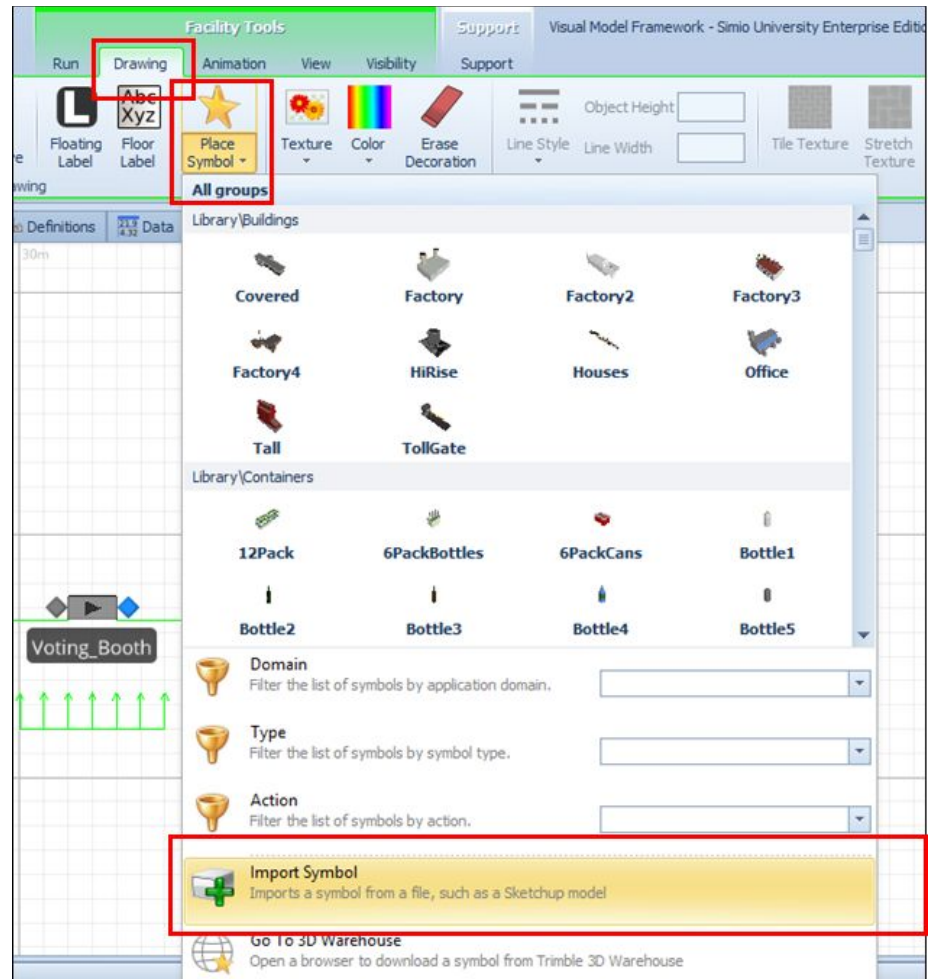
      1.1.5. Use the cursor to **place** the symbol above the default model objects. **Left click** to confirm the placement of the object.
   1.1.6. Green dots will appear around the object (if not, left click on the object again). While pressing the **Control key (CTRL), left click and hold** one of the green dots on any corner.
      1.1.7. **Rotate** the object 90deg **clockwise** using the cursor.
      1.1.8. **Right click** on the center of the object and select *Lock Edits* from the dropdown menu.

1.2. Adding Check-in Station

    1.2.1. **Repeat** steps **1.1.1 - 1.1.3**, this time **selecting** the *Table short.skp* file.

    1.2.2. Use the cursor to **place** the symbol on top of the *Empty Floor Space*.

    1.2.3. Green dots will appear around the object (if not, left click on the object again). While pressing the **Control key (CTRL), left click and hold** one of the green dots on any corner.

    1.2.4. **Rotate** the object 90deg **clockwise** using the cursor.

    1.2.5. Using the cursor, **Select** the table object, and **place** it on the right side of the wood floor. The bottom edge of the table should be aligned with the boundary of the gray floorspace.

    1.2.6. **Select** the table object. Then use the keyboard shortcuts to **copy(CTRL C) and paste(CTRL V)** to add a second table to the model

    1.2.7. Use the cursor to **place** the table to the **left** of the first table, at least **6 feet** from its closest edge and **in parallel**.
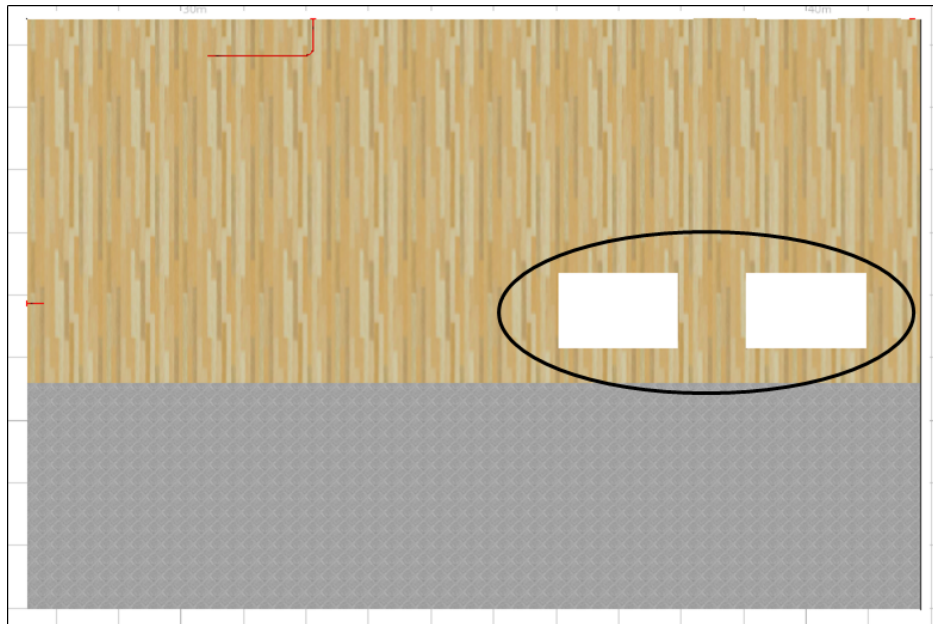

1.3. Adding Voting Booths

    1.3.1. **Repeat steps 1.1.1 - 1.1.3,** this time **selecting** the *Voting Booth.skp* file.

    1.3.2. Use the cursor to **place** the voting booth object along the **back wall**.

    1.3.3. **Select** the voting booth object. Then use the keyboard shortcuts to **copy(CTRL C) and paste(CTRL V)** to add a second voting booth to the model.

    1.3.4. Use the cursor to **place** the voting to the **left** or **right** of the first voting booth **in parallel,** allowing for at least **6 feet** of open space between their closest edges.

1.3.5. **Repeat steps 1.3.3 and 1.3.4** until there are a total of **7** voting booths in the model.
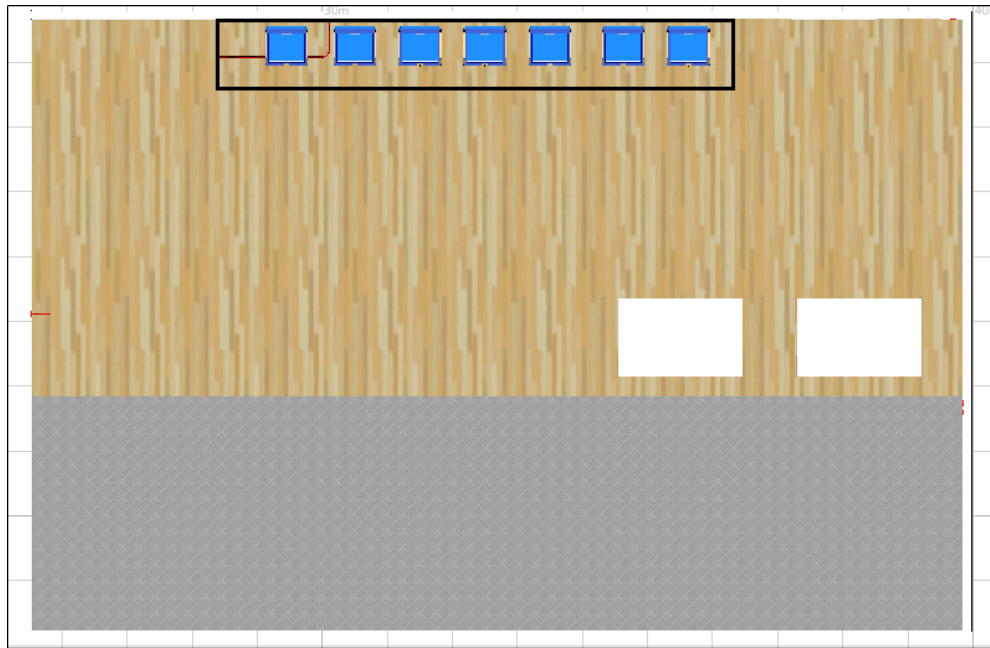
1.4. Adding DS200 Scanners

    1.4.1. **Repeat steps 1.1.1 - 1.1.3,** this time **selecting** the *DS200 Scanner.skp* file.

    1.4.2. Use the cursor to **place** the DS200 object along the **left edge** of the floor space in the wood section along the boundary line of the gray floor section.

    1.4.3. **Select** the DS200 scanner object. Then use the keyboard shortcuts to **copy(CTRL C) and paste(CTRL V)** to add a second DS200 scanner to the model.

    1.4.4. Use the cursor to **place** the DS200 Scanner above the first DS200 Scanner, **in parallel,** allowing for **6 feet** of space between the devices.

    1.4.5. After all model items have been placed, **press and hold the CTRL key** and **select** all items with the cursor.

    1.4.6. **Right click** and **Select** *Lock Edits* to temporarily lock the placement of all model items.
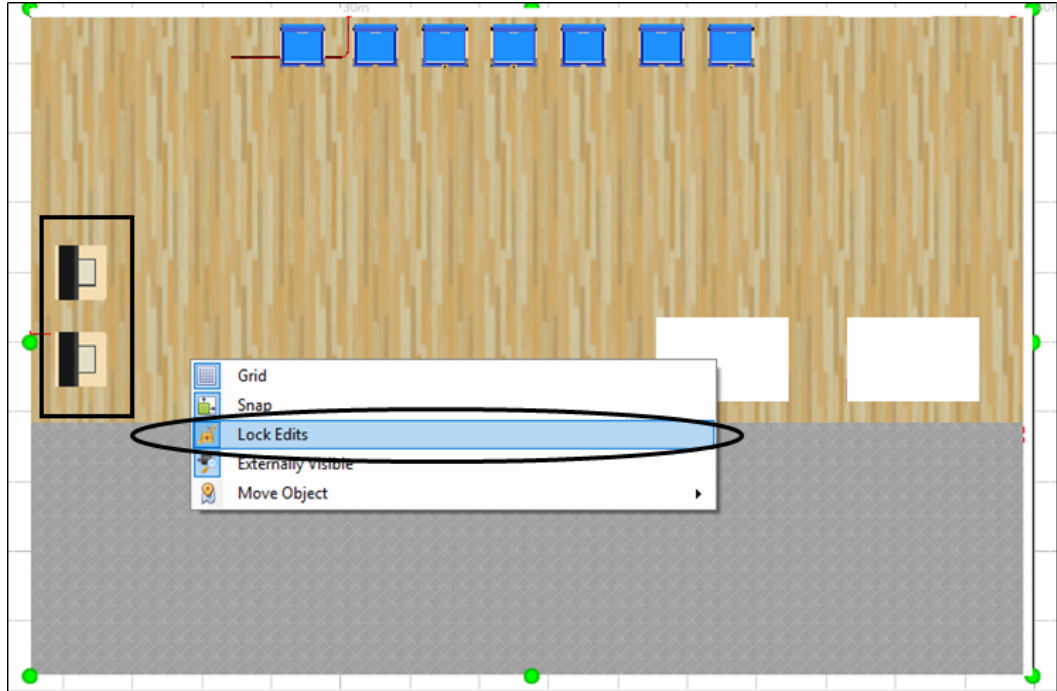
2. Arranging Server Objects

After the voting equipment has been set in place, it is time to arrange the preprogrammed server objects. The following sections describe how to set up Server objects in a Visual Simulation model.
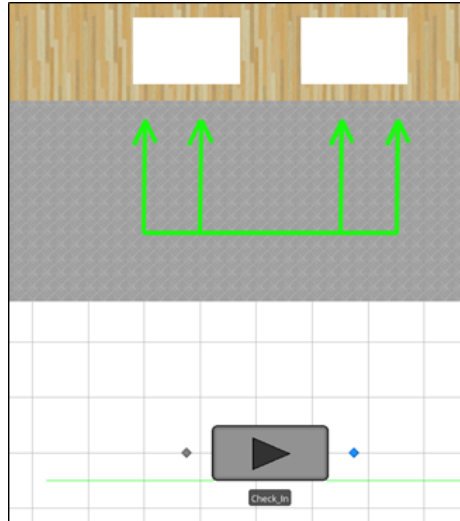
2.1. Source Object

    2.1.1. First, **Left Click** the Server object labeled *Source1.* **Left Click** again and **hold** to move the Source and position it such that the blue output node is at the **bottom left corner** of the Floorspace.
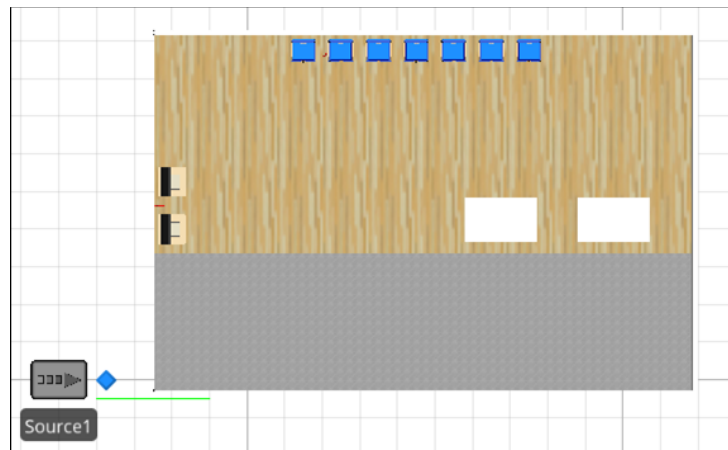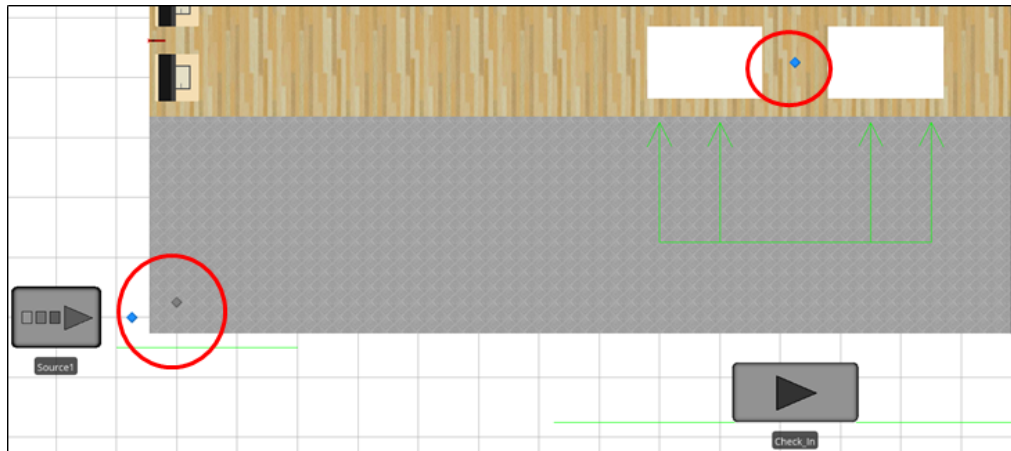
2.2.   Check_In Server Object
   2.2.1.   **Left click twice** on the *Check_In* server object. Using the cursor, **move** the *Check_In* server object such that the green arrows are **positioned in front** of the two tables as shown below. Ensure that the queue vertices are at least **6 feet** apart, so that voters are an appropriate distance from one another.


   2.2.2.   **Select** the gray *Input Node* to the **left** of the *Check_In* Server object. Using the cursor, **place** the *Input Node* at the **inside bottom left corner** of the floor space **adjacent** to the blue *Output Node* of the Source Object.
   2.2.3.   **Select** the blue *Output Node* to the right of the Check_In Server object. Using the Cursor, **place** the Output node **between** the two tables.
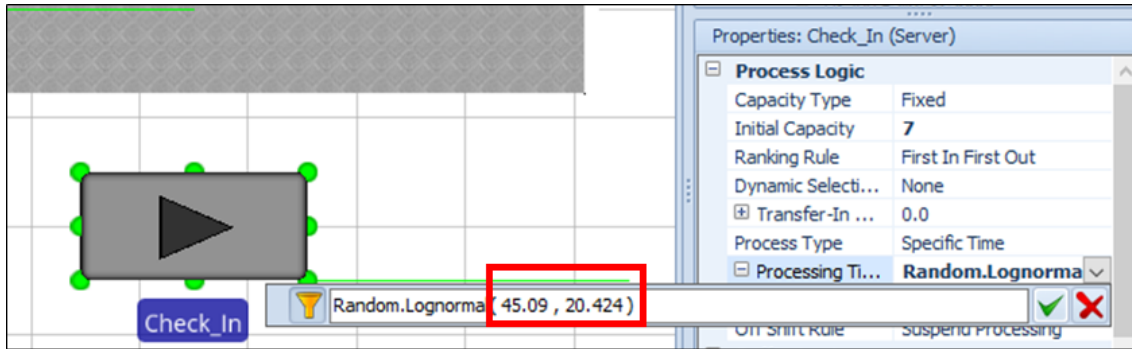

   2.2.4.   **Select** the green Input Queue on the **left** of the Check_In server object. Using the cursor, **move** the Input queue just below the **bottom** of the floor space with the **left edge** aligned with the gray Input Node.
   2.2.5.   **Select** the green circle at the **right end** of the Input Queue, and it will turn red. Use the cursor to **extend** the green line as shown in the center image below

2.2.6.   **Select** the green Input Queue, then **place** it on the floor space as shown in the image below on the right.

2.2.7.   To represent a **6 foot** distance between voters, **select** the Input Queue and look to the Queue Tools tab on the **ribbon** in the upper part of the screen. Within the Appearance menu, **select** the Point Queue.

2.2.8.   **Adjust** the vertices that are present on the Input Queue to be **6 feet** apart.

2.2.9.   **Add** more vertices as required (i.e., to match the capacity of the input buffer for the *Check_In* server) using the Queue Tools tab, ensuring that each added vertex is **6 feet** away from all others.
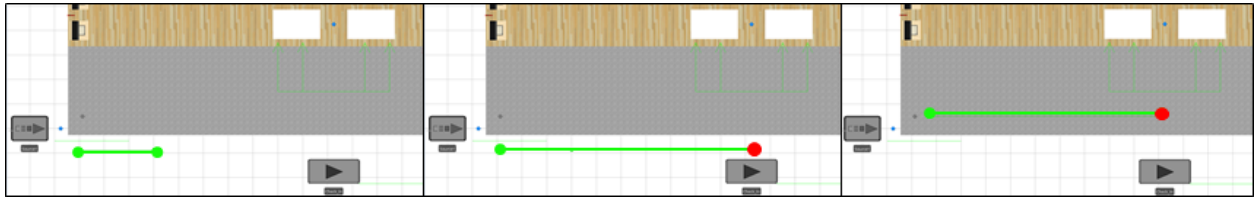
2.2.10.   The Check_In processing times are preprogrammed based on Time Study observations collected at various Rhode Island polling places during the 2018 midterm elections. The following steps can be used to modify the processing time behavior based on other historical data sources.

2.2.10.1.   Compute the mean and standard deviation of process time observations, in seconds, from the historical dataset.

2.2.10.2.   Left click to Select the Check_In Server Object

2.2.10.3.   Left Click on the cell to the left of the Processing Time label in the Properties Table on the right.

2.2.10.4.   Left Click on the dropdown menu button exposing the contents inside the text box.

2.2.10.5.   Replace the values inside the parenthesis with the historical mean and standard deviation from step 2.2.7.1.

2.2.10.6.   Select the green checkmark button to confirm these changes.

Properties: Check_In (Server)

Process Logic	
Capacity Type	Fixed
Initial Capacity	**7**
Ranking Rule	First In First Out
Dynamic Selecti...	None
⊞ Transfer-In ...	0.0
Process Type	Specific Time
⊟ Processing Ti...	**Random.Lognorma** ∨
Off Shift Rule	Suspend Processing

Check_In ▽ Random.Lognorma **(45.09 , 20.424)** ✔ ✘

2.3.   Voting Booth Server Object

2.3.1.   **Select** the Voting_Booth Server object. Using the cursor, **position** the Voting_Booth Server object, as shown below on the left.

2.3.2.   **Left click** on the **bottom** of the green Service Queue exposing the green vertex nodes. Using the cursor, **adjust** each of the **bottom** nodes (left or right) to be **roughly centered** with a single voting booth. Ensure that each vertex is at least **6 feet** away from other vertices at this station.

2.3.3.   **Select** the Voting_Booth Server object. Using the cursor, **position** the Voting_Booth Server object as shown below in the center image.

2.3.4.   Using the cursor, **position** the blue Output Node as shown **(yellow)** below on the right. **Position** the gray Input Node as shown **(red)** below on the right.
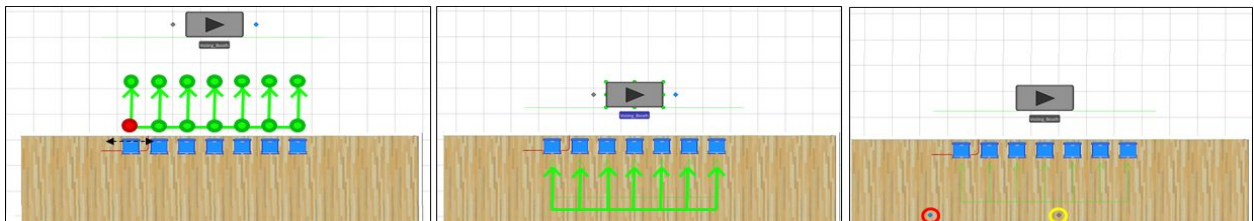

2.3.5.   The Voting_Booth processing times are preprogrammed based on Time Study observations collected in Providence, Rhode Island during the 2018 midterm elections. Select the Voting_Booth server object and follow steps 2.2.7.3 - 2.2.7.6 to modify the processing time behavior based on other historical data sources.

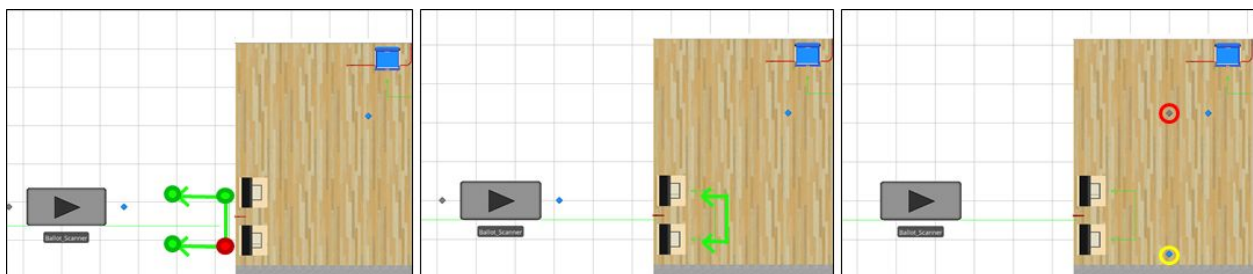2.4.   Ballot Scanner Server Object

2.4.1.   **Select** the *Ballot_Scanner* Server object. Using the cursor, **position** the Ballot_Scanner Server object as shown below on the left.

2.4.2.   The Ballot_Scanner processing times are preprogrammed based on a two sheet ballot. Use the following steps to **adjust** the processing time behavior to according the **actual sheet count** of the ballot:

2.4.2.1. **Left Click** to **Select** the *Ballot_Scanner* Server Object.
2.4.2.2. **Left Click** on the cell to the left of the *Processing Time* label in the Properties Table on the right.
2.4.2.3. **Left Click** on the dropdown menu button exposing the contents inside the text box.
2.4.2.4. The value immediately before the closing ")" at the end of the text indicates the sheet count of the ballot. **Replace** the current value of 2 with the **actual sheet count** of the ballot.
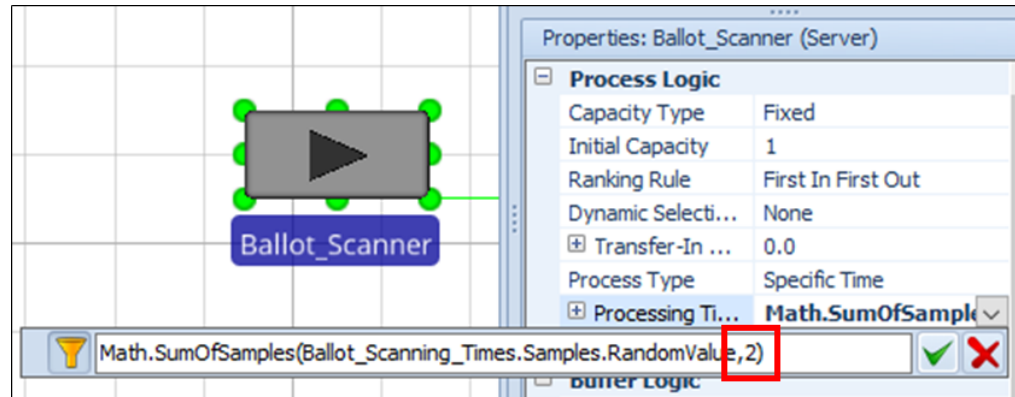

2.4.3. **Left click** on the **bottom** of the green Service Queue, exposing the green vertex nodes. Using the cursor, **adjust** each of the **bottom nodes** (up or down) to be **roughly centered** with a single ballot scanner.
2.4.4. **Select** the *Ballot_Scanner* Server object. Using the cursor, **position** the Ballot_Scanner Server object as shown below in the center image.
2.4.5. Using the cursor, **position** the blue *Output Node* as shown **(yellow)** below on the right. **Position** the gray *Input Node* as shown **(red)** below on the right.


2.4.6. **Left click** on the green *Input Queue* of the Ballot_Scanner object, exposing the green vertex nodes.
2.4.7. **Select** the **left** vertex node, and it will turn red. Use the cursor to **position** the selected vertex node next to the gray *Input Node* as shown in **red.**

2.4.8.   **Select** the other vertex node, and it will turn red. Use the cursor to **position** the selected vertex node next to the green *Service Queue* as shown below in **yellow,** ensuring that the two vertices are positioned at least **6 feet** from one another.
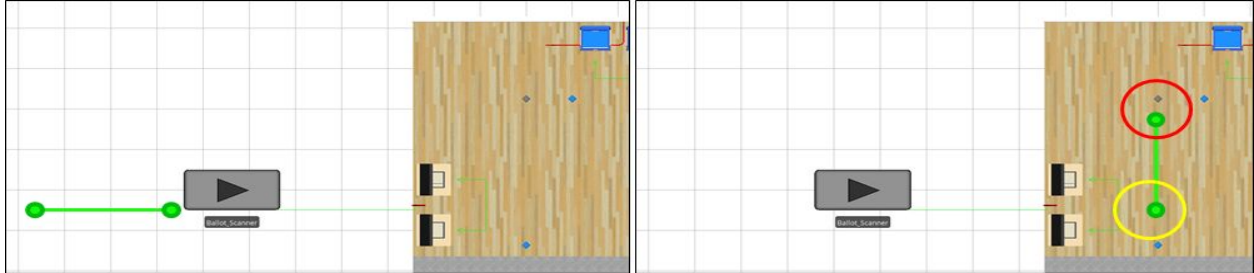
2.5.   Sink Server Object
2.5.1.   **Select** the object labeled *Sink1*. Use the cursor to **position** the Sink 1 object **above** the Source1 object as shown below.
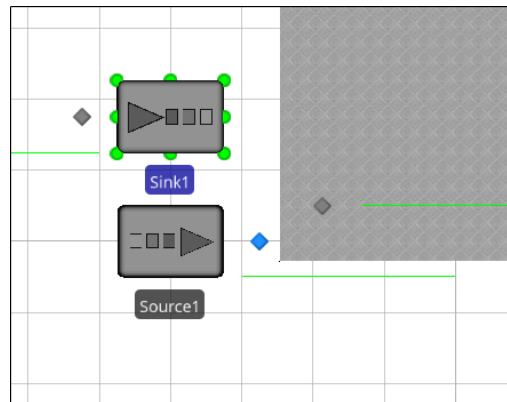

3.   Adding Routing Logic
The last step in setting up the visual simulation model is to add paths for routing voters in through the simulation model in the correct sequence. Slight changes to the default properties must also be made to each path to accurately control the flow of voters.

3.1.   **Left click ONCE** to **Select** the *Path* icon from the Standard Library on the left of the screen.
3.2.   **Select** the blue *Output Node* of Source1 as the starting point of the first path with a **single Left Click**.

3.3. Use the cursor to **draw** a path from the gray *Output Node* of Source1 to the blue *Input Node* of Check_In Server object. A **single Left Click** may be used to **add** vertices to the path along the way.

3.4. **Select** the *Input Node* of the Check_In Server Object with a **single Left Click**. This path is now completed.

3.5. Navigate to the *Properties Table* on the **right** side of the screen and **change** *Allow Passing* from True to **False**.
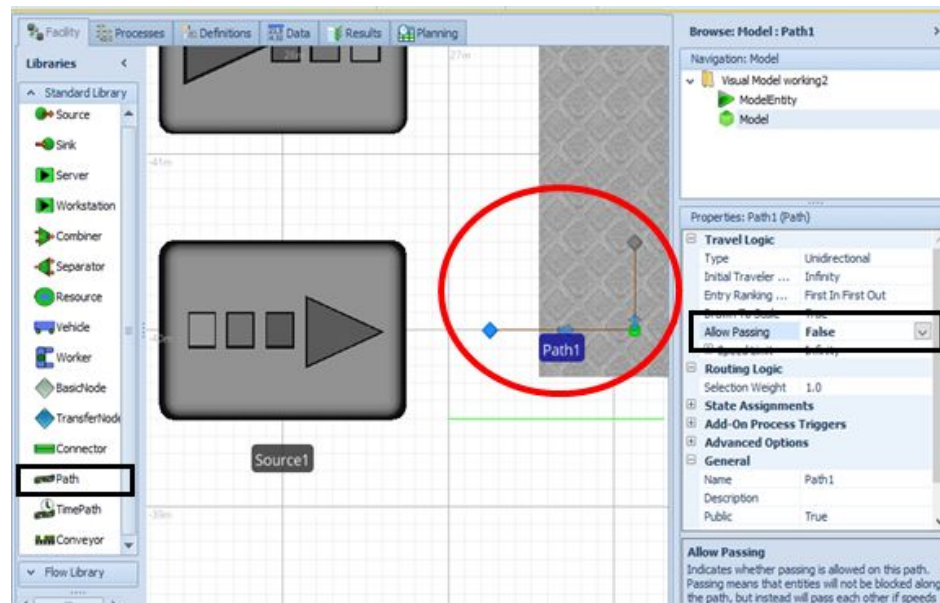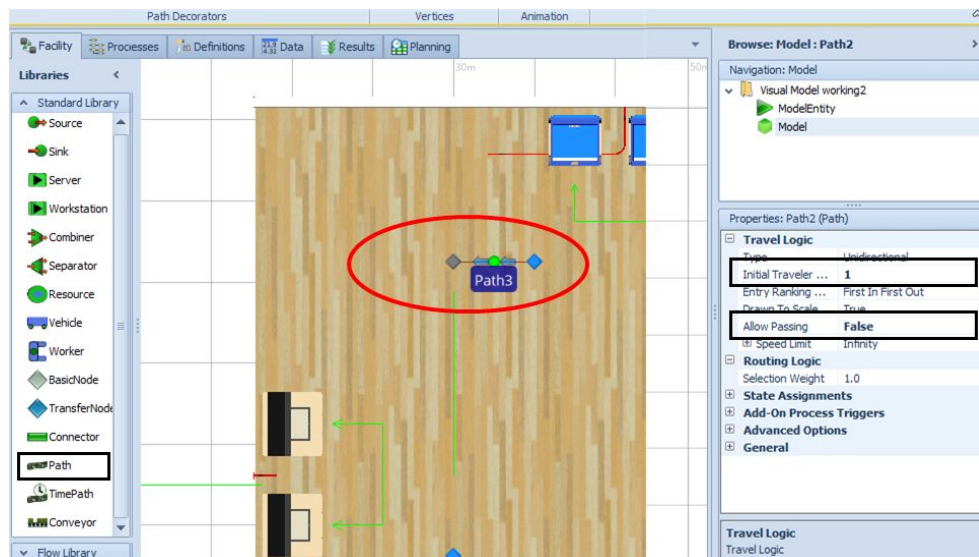

3.6. Use the cursor to **draw** a path from the gray *Output Node* of the Check_In Server Object to blue *Input Node* of the Vothing_Booth Server object as shown in the image below step 3.7.

3.7. Navigate to the *Properties Table* on the **right** side of the screen and make the following changes:

    3.7.1. **Change** the *Allow Passing* property from True to **False**

    3.7.2. **Change** the *Initial Traveler Capacity* property from Infinity to **1**.
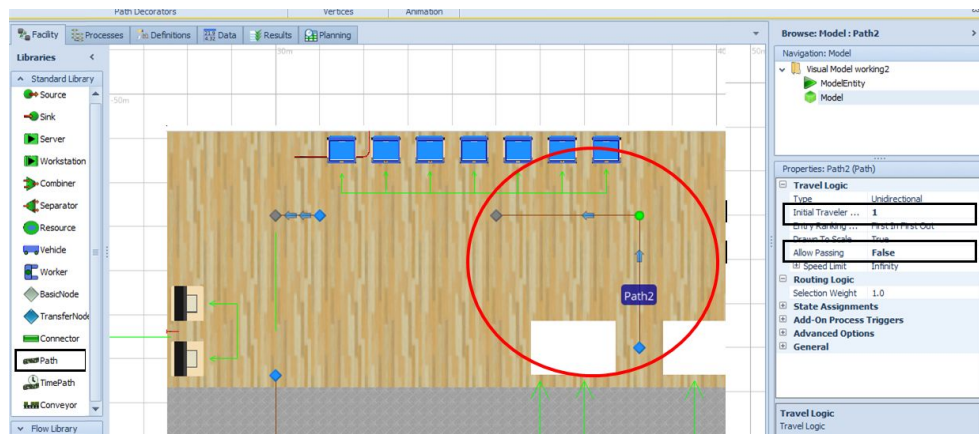
3.8. Use the cursor to **draw** a path from the gray *Output Node* of the Voting_Booth Server Object to the blue *Input Node* of the Ballot_Scanner Server object, as shown below.

3.9. **Repeat** steps **3.7.1** and **3.7.2** in the *Properties Table* for Path3.

3.10. Use the cursor to **draw** a path from the gray *Output Node* of the Ballot_Scanner Server Object to the blue *Input Node* of the Sink1 Server object, as shown below.

3.11. **Repeat** steps **3.7.1** and **3.7.2** in the *Properties Table* for Path4.
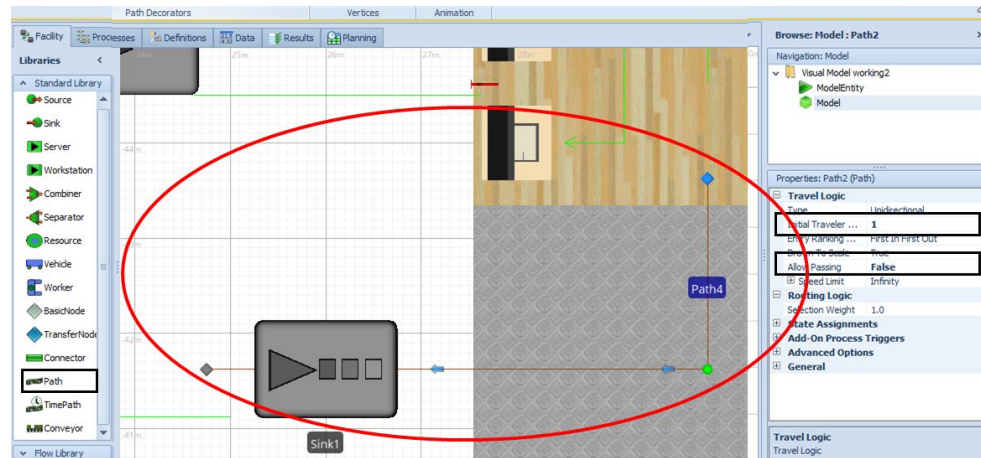
Feedback Section:

Please contact uri.sis.lab@gmail.com with any questions, concerns, or feedback. We will respond as quickly as possible to help resolve any problems.