AUBURN UNIVERSITY

Samuel Ginn College of Engineering

3D Modeling of Detached Metal Whiskers User Operations Manual

Version 2.0

MECH 4250 – Auburn Group 3, Metal Whiskers Team 2

Shawn Eom, Spencer Hurst, Kurt Knudsen, Nathan Nichols, Carson Reams

Finalized 12/4/2024

Table of Contents

List of Figures

Introduction

This manual contains information regarding the 3D Modeling of Detached Metal Whiskers Simulation, created by two teams of Senior Mechanical Engineering students at Auburn University. This program is designed to utilize the Unity Game and Physics Engine to run simulations that analyze the odds of whisker bridging across circuit boards in various environments and settings. It is intended for use in reliability testing and predictive analysis in the field of electronics, offering a user-friendly interface to assess and mitigate the risks associated with whiskers on and around circuit boards.

The program allows users to customize their simulation, providing control over whisker dimensions and properties, spawning locations, external forces, and circuit board analysis, among other features. The end goal of the program is to give users a realistic simulation of their own boards, which would be easily imported into the program, and study the behavior of whiskers under different mission environments. With highly customizable parameters, users can tailor any simulation to specific, real-world scenarios.

This user manual is designed to be a comprehensive guide to effectively utilizing the 3D Modeling of Detached Metal Whiskers Simulation. By following the instructions and insights provided, you will be able to maximize the program's potential, making informed decisions to enhance the reliability of your electronic designs. Any instruction denoted by a '*' are any notes of workarounds or functions that are not fully implemented. If this is the first time accessing and learning how to utilize this program, continue to: **Getting Started.** If the program is already functioning and a faster, more condensed version of the instruction is needed, refer to **Quick Overview.**

Getting Started

Section 1: GitHub and Repository Access.

To gain access to the simulation, you will need to obtain it through GitHub. GitHub is a powerful version control and software management tool that enables developers to collaborate efficiently and manage their code effectively. By using GitHub, you can easily access the latest version of the simulation, ensuring that you are always up to date with the most recent changes and improvements. GitHub can also facilitate easy updates, allowing users to synchronize their local copies with the latest changes made by developers. This process eliminates the need for extensive redownloading. Additionally, GitHub's version control system tracks all modifications made to the code, allowing developers to provide a comprehensive history of changes to the users as updates come out. This makes it easy to revert to previous versions if needed and understand the evolution of the project over time. The utilization of GitHub for this simulation was vital to allow users to easily access and manage the program files, along with staying updated as future teams develop the program further. The following sections explain how to gain access to this program

There are two main methods that can be used to download and gain access to this simulation. Both methods require a GitHub account. If you do not have an account already, it is vital to create one using the following link and instructions listed on the website:

https://github.com/

Once an account has been made, you may follow one of the two methods to download or gain access to the repository.

Method 1: Using GitHub Desktop (Recommended):

a. If you do not have GitHub Desktop already, refer to the following link to download:

https://github.com/apps/desktop

b. Once downloaded, open the app and log in.
c. From there, look to the top left of the app for the File tab. Click this, and find the option: Clone Repository

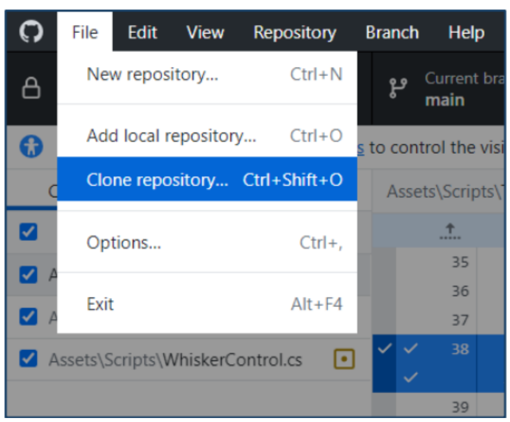


Figure 1: Cloning Repository

d. This will open a new screen. Click on the tab titled URL, and enter in the following URL to the input field:

https://github.com/kak0069/3DStuff.git

e. Define a directory for the repository to be cloned into. Then, click Clone.

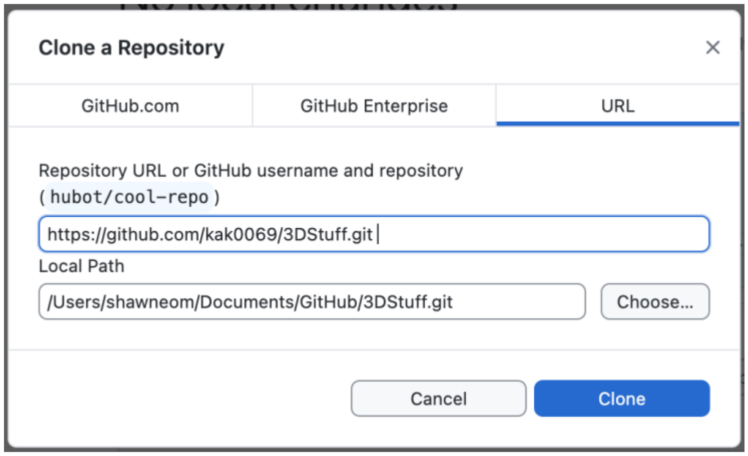


Figure 2: GitHub URL

The repository containing the simulation is now saved to your device, and you may skip to Unity Startup

Method 2: GitHub Website:

This method is more prone to issues, as it revolves around downloading the files instead of cloning them directly onto your device, which takes up more space and is much slower. This method is not recommended but can be used if GitHub Desktop is not accessible.
Assuming a GitHub account has been made, navigate to the following link to access the online version of the simulation repository:

https://github.com/kak0069/3DStuff

a. Click the green Code button and click Download Zip. This will begin the installation of the program files.
b. Once fully downloaded, extract the files to a directory of your choice on your device.

The repository containing the simulation is now saved to your device, and you may continue on to Unity Startup.

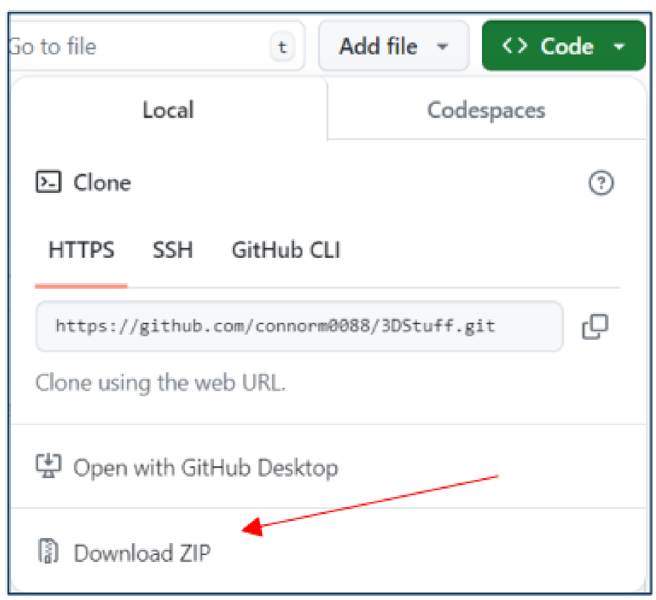


Figure 3: GitHub Repository Webpage

Unity Startup

Section 1: Downloading Unity.

As stated previously, this program runs off of the Unity Game Engine, so it is vital to have the Unity Editor and Unity Hub installed for this program to work as intended. The both the Editor and the Hub can be downloaded by following the official tutorial found on the link below. This program was developed in version 2022.3.19f1 of the Editor. It is highly recommended to get this same version for running the simulation as other versions might have different features and layouts, making it more complicated to follow the instructions.

Unity Hub and Editor Download

Section 2: Unity Hub

Once completed, you will have access to the Unity Editor and the Unity Hub. The Hub is a Unity program used to control and manage all of the project's users may have saved. This will be the main way to access and run the simulation.

To access the simulation, the repository saved in **Getting Started, Section 1** must be added to your Unity Hub. Refer to the following instructions to complete this process:

a. Open Unity Hub.
b. In the Projects tab, click Add. This will bring up your file explorer. *The image for your reference already has projects listed, however, your Unity Hub will most likely be empty.

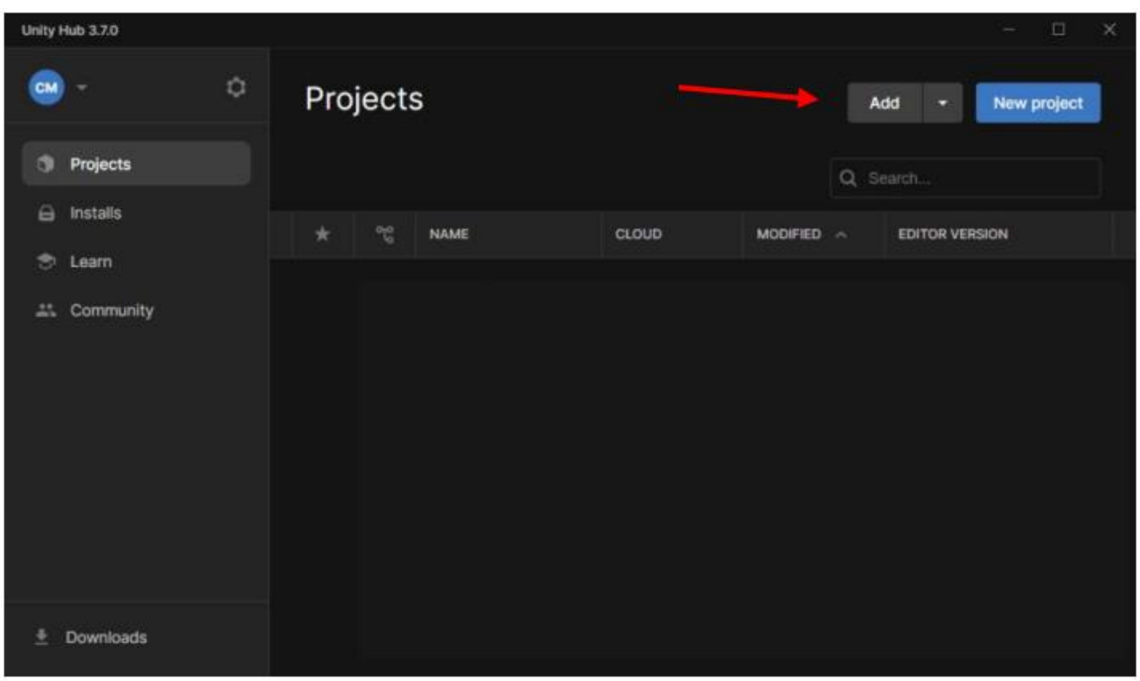


Figure 4: Unity Hub Interface

c. Navigate to the directory where the repository was saved. Click the repository and click Open.

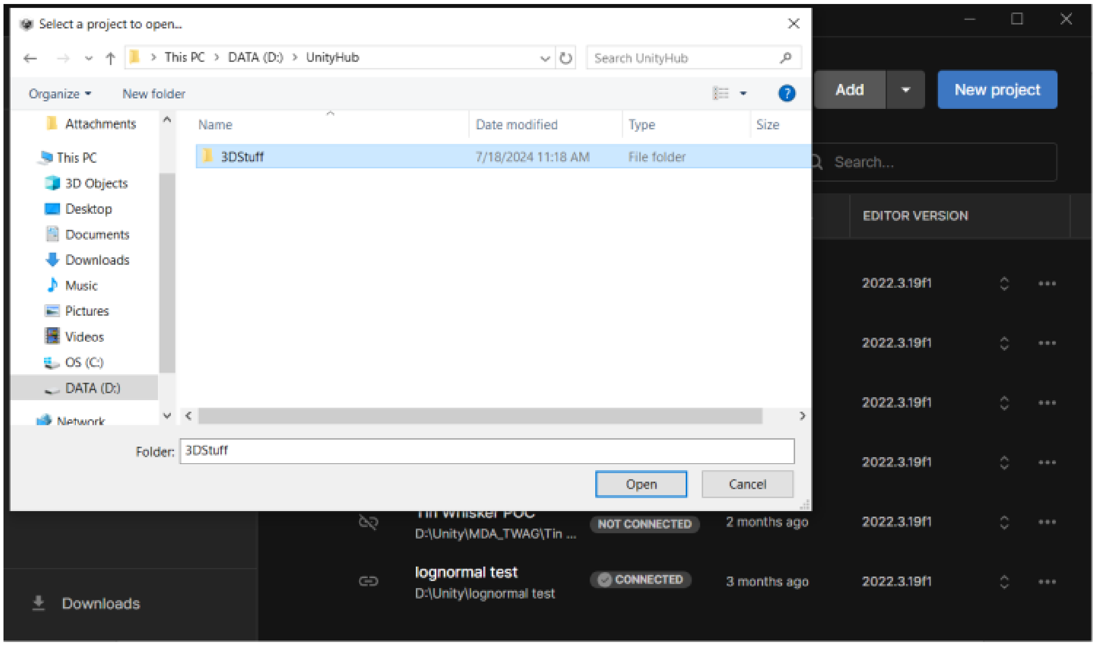


Figure 5: Selecting the Repository

The project should now be available in the Unity Hub. You may move to Section 3: Unity Editor. *NOTE: repository name may be different from the one shown in Figure 5.

Section 3: Unity Editor

Now that the simulation has been cloned from GitHub and added to your Unity Hub, you may now open the simulation. Clicking on the project will open it in the Editor. Please allow a few moments for this to complete. When the project is opened, you should see Unity's interface in the Editor. There are four main sections of the Editor: Hierarchy, Scene/Game window, Inspector, and the Project/Console Tabs. If this is the first time the Editor is being opened, it should be blank with nothing in the main sections. As a brief overview, the following table describes what each section handles when using the Editor:

Editor Layout	
Hierarchy	On the left side of the Editor, the hierarchy lists all the Game Objects in the current scene in a hierarchical order
Scene/Game Window	Scene: Interactive view into the simulation where objects can be manipulated Game: Preview of the simulation before being ran.
Inspector	On the right side of the Editor, it is filled whenever a Game Object is selected and provides important information about the object.
Project/Console Tabs	At the bottom left side of the Editor, these tabs contain project folders and any error messages being printed from the console

It is not necessary to be fully familiar with the Editor and its features to run this program. However, if more information is desired, the official Unity website has countless tutorials and help pages to answer any questions. Figure 6 shows a fresh Editor when opened:

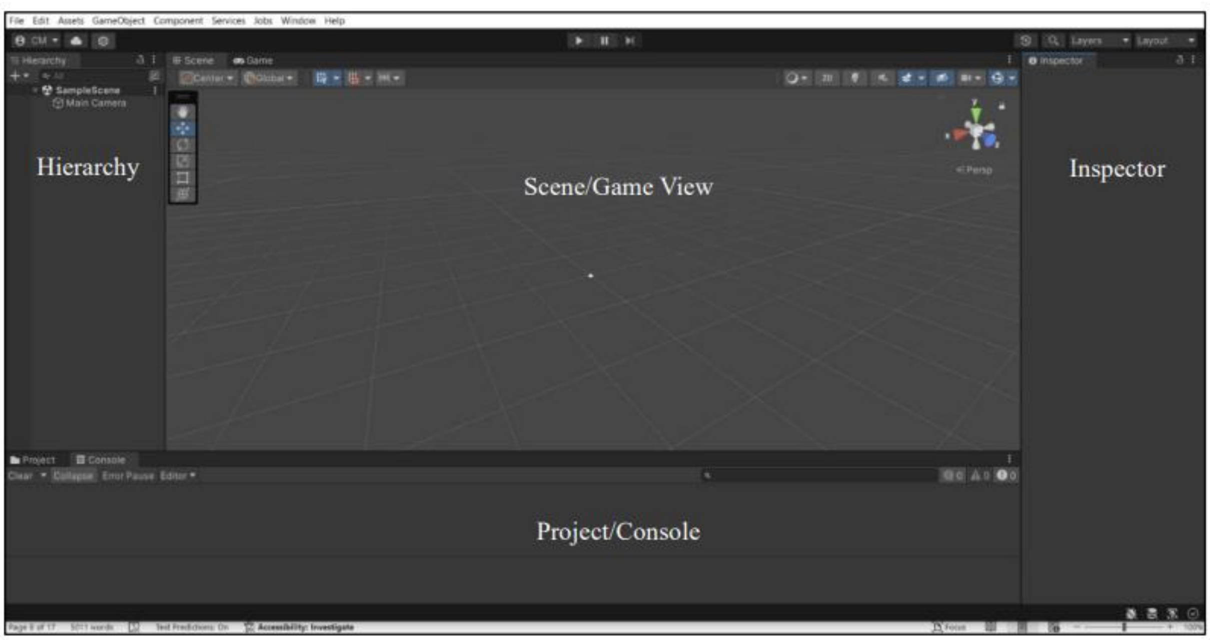


Figure 6: Empty Unity Editor

Section 4: Importing the Simulation

Once in the Editor, you may now complete the process of accessing the simulation. Refer to the following instructions to finalize this process:

a. To fully import the simulation into the fresh Unity Editor, click on Project in the Project/Console Tabs. On the left side of the tab, there should be a folder icon labeled Assets. Click the dropdown next to this folder.

b. Once opened, there will be a file labeled Scenes. Click into this folder.

c. Once inside, there will be one object: a Unity logo with the name SampleScene. Double click this object to fully import the Simulation into your

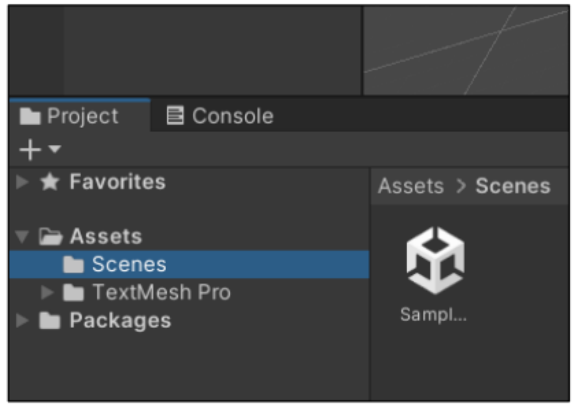


Figure 7: SampleScene

Once these steps have been completed, the simulation is now being hosted by the Editor. To verify this was done correctly, refer to Figure 8 to compare your Editor. These steps will not have to be repeated as the simulation is now saved to your Unity Hub and can be reopened at any time.

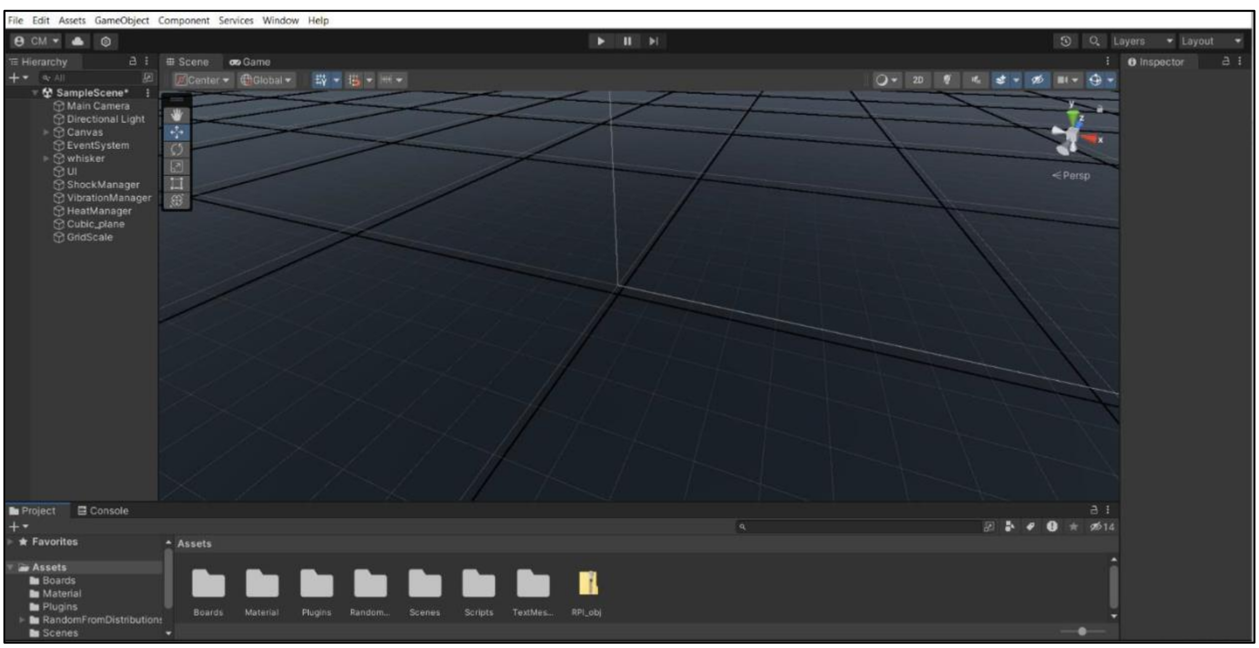


Figure 8: Simulation Editor Screen

You may now proceed to **Setting up your Simulation.**

Setting up your Simulation:

Section 1: Loading in your Desired Board

To get the most accurate results for your desired board, it will need to be imported into Unity as a .obj file. Other file types can be imported into Unity successfully, but some manipulation of the components may be required to get them to function properly. Many programs such as Altium, Blendr, Autodesk Eagle, and others have the capabilities to produce .obj files while maintaining the important contours and components of your circuit board.

a. Once you have your desired PCB in a compatible file type, open the folder containing the 3D object. Next you will need to access the Boards folder that came with the simulation. This can be done through the Unity Editor, under the Project tab. Click to open the Assets folder, and find a new folder titled Boards, then click into the folder. Once here, drag and drop your board into the folder. The board will now be present in the Project tab. * Disregard preexisting boards in Figure 10. These will not be present in your project.

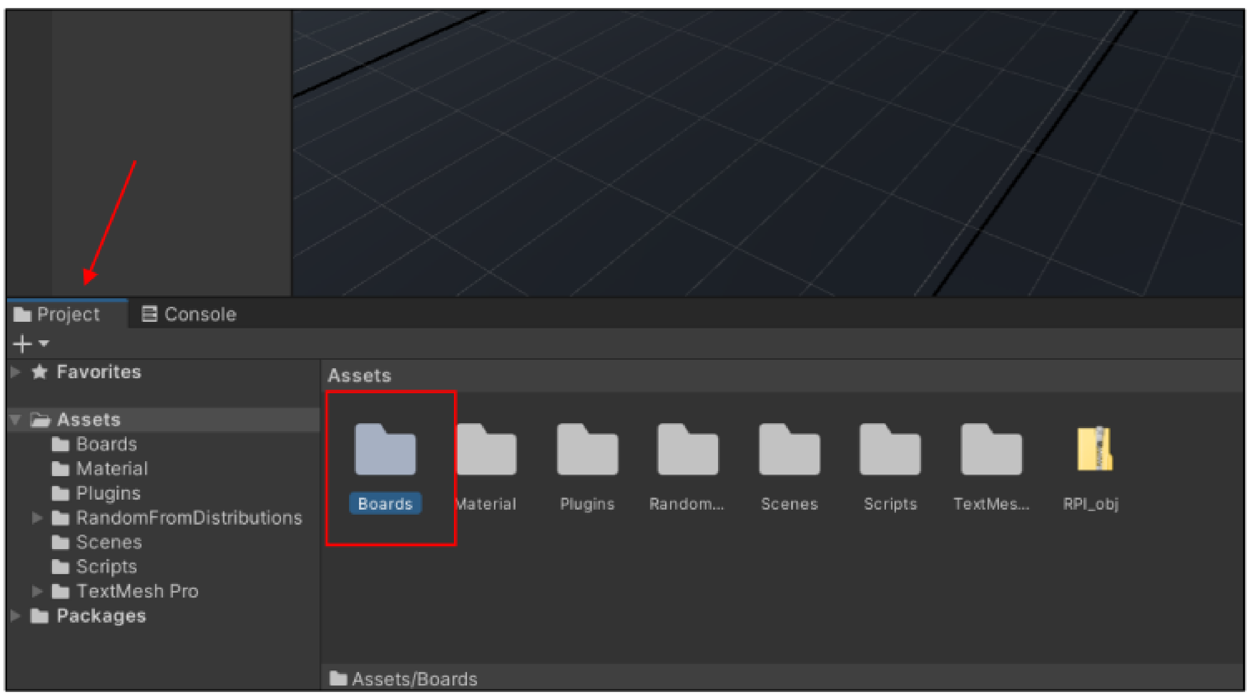


Figure 9: Accessing Board Folder

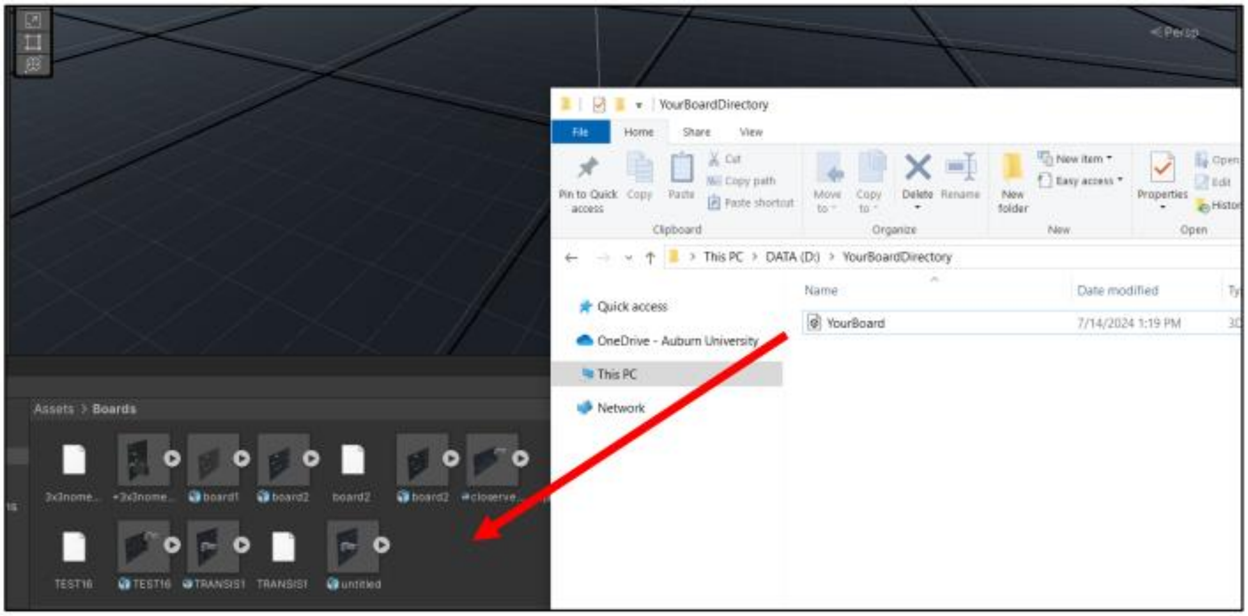


Figure 10: Importing your PCB

b. Click on your board and drag it into the Scene View. It does not matter where it is placed in the Scene, as through the following steps and the background code, it will be placed for you. Figure 11 demonstrates this task (Disregard excess boards shown in figure).

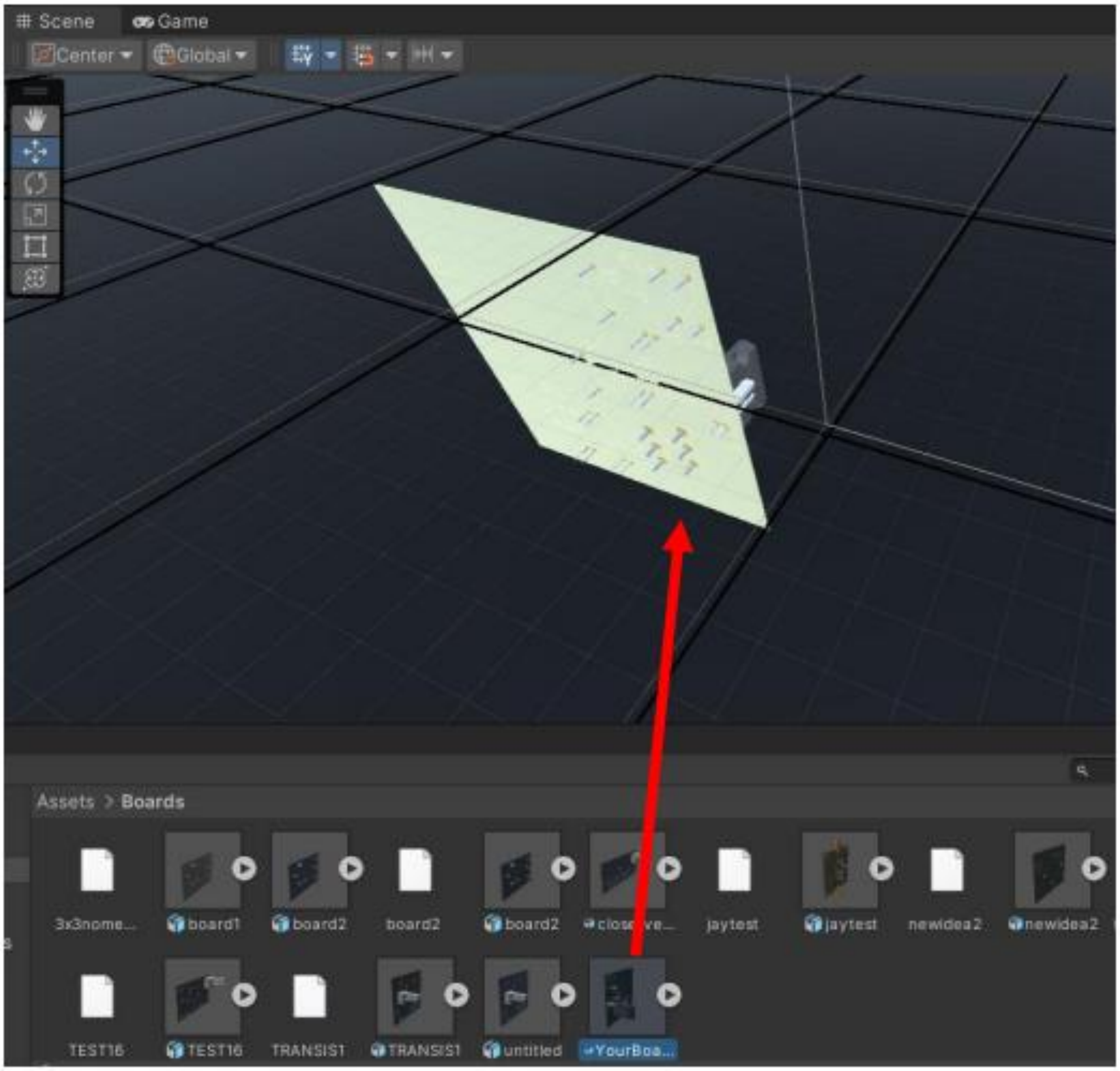


Figure 11: Dragging in the PCB

c. Correct the Orientation of your board in the case it does not import correctly. This can be done by selecting the board, and then looking in the Inspector tab. There should be a subsection called Transform, Rotation, where you can edit the rotation of the board to orient it properly. Example: typing in 90 under the X axis rotation will rotate the board 90 degrees in the positive X direction. Once this is done, the board should now be ready for the setup.

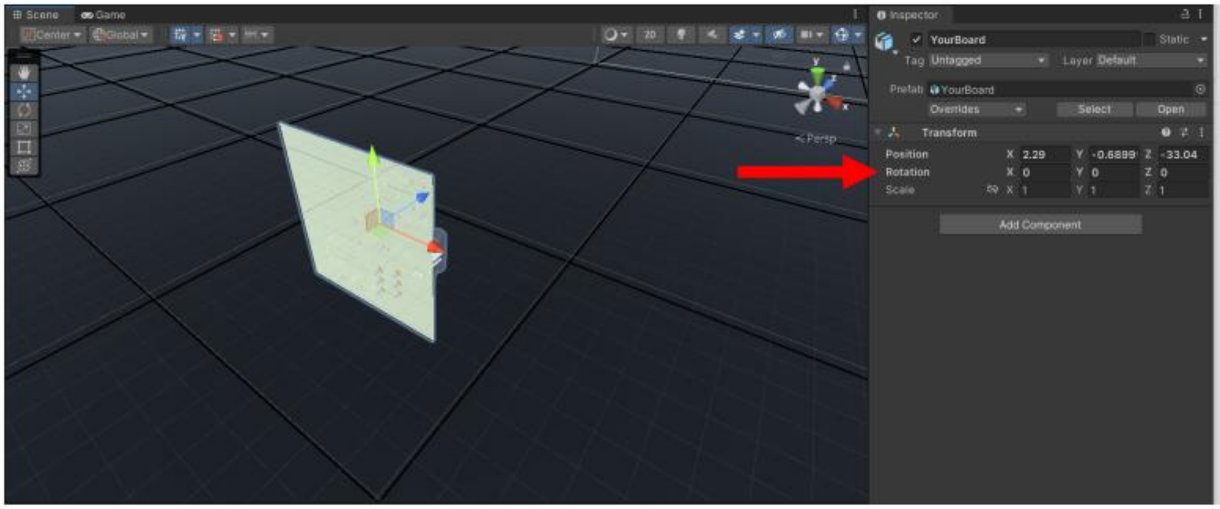
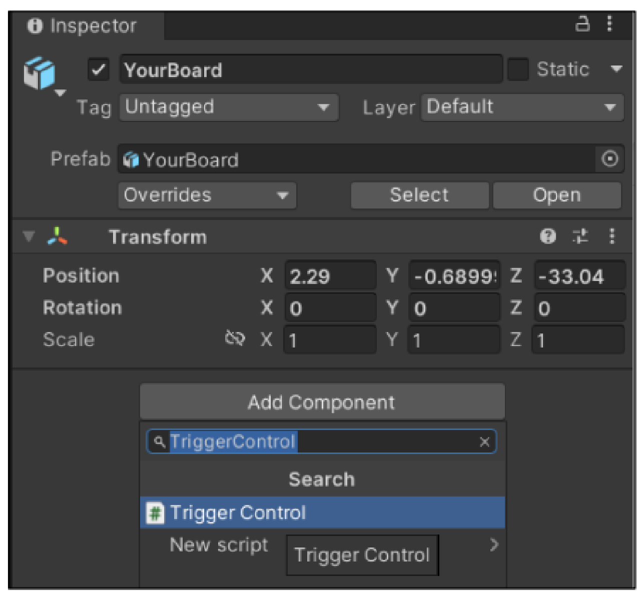


Figure 12: Orienting your board

Section 2: Preparing your board for Simulation

A few easy steps are required in order for the simulation to run as needed. These involve using the unity editor to attach certain scripts/objects to the board in order to effectively run the simulation.

a. If the previous section was completed properly, the board should now be in the Scene. Select your board and look to the Inspector. There will be a button called Add Component. Clicking this will bring up a search bar. Type in the bar: TriggerControl, and an icon that looks like a paper with a green pound symbol with the matching name should show up. Click this item. The script will be added as a component, and a new field will be added to the Inspector for your board.*



*NOTE: as of this current rendition of the program, the PCB can be loaded into the simulation, but will not have the proper material properties to be able to interact with whiskers to simulate a bridge

Figure 13: Adding Components

b. Now, you should see a few new fields in the Inspector under the section Trigger Control. There is an empty field titled Trigger Material. Click the small circle to the right of this field, which will bring up a text input field. Type "TriggerMaterial" and click the item that comes up.

c. Finally, directly under the Trigger Material field should be one titled Material Input. Similar to the last step, click the small circle to the right of the field, but instead type "MaterialInput", and click the item that comes up. Leave all other fields in this section untouched. Your board is now ready to be run through the simulation. Figure 14 shows what the fields should look like once completed properly.

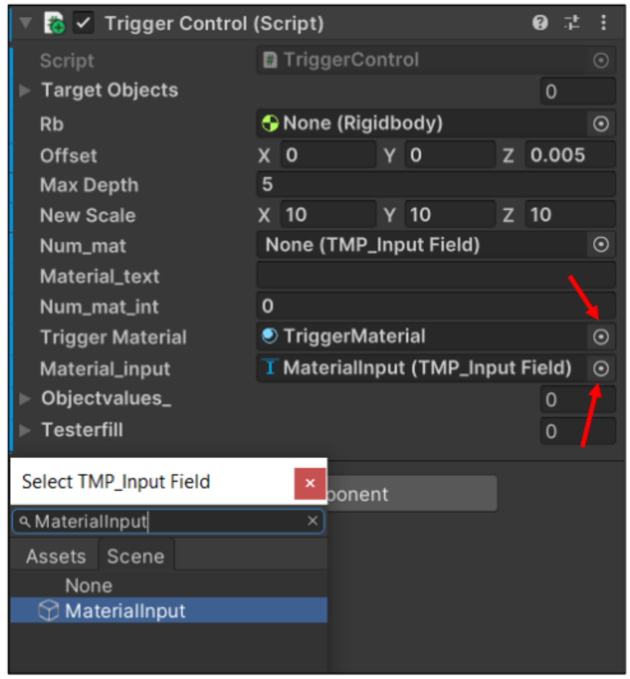


Figure 14: Finalizing the Board

The PCB of choice can now be correctly analyzed in the simulation. Keep in mind that these steps will need to be repeated as more boards are imported into the simulation but will not need to be repeated for ones already set up. If there are multiple boards in the Scene and only one is required for testing, you can turn off the others by selecting them and looking at the top left of the Inspector window. There, you will see a small check mark indicating that the board is 'on.' Clicking this check mark will turn the board off, allowing Unity to retain the board's setup but remove it from the Scene until it is needed again.

Running the Simulation (Walkthrough):

Section 1: Camera Controls

Now that the boards are set up, the simulation can be run. To do this, look to the top of your Unity Editor. Click the Play Button to start the program.

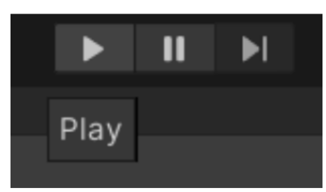


Figure 15: Play Button

a. When in Play Mode, users can navigate the camera around the simulation to get the best views of their PCB/whisker bridges. There are directions in the bottom left corner which tell the user how to use the camera function where LMB is 'Left Mouse Button' and RMB is 'Right Mouse Button'. To the camera, hold the left mouse button and move the mouse in any direction. Additionally, from the direction the camera is facing, the E and Q keys can be used to move the camera vertically up or down and clicking the RMB or the Down Arrow Key will reset the camera orientation to level with the board. On the bottom right of the Interface, there is a section titled 'Camera', which has three buttons that when pressed, set the camera to preset conditions. Standard is a front facing ariel view, Top Down is a bird's eye view from directly over the board, and Side is a side facing view from the left side of the PCB. NOTE these buttons are also toggleable when in "close-up" view of the whisker, which will be explained in a later section.

Figure 16 shows what the simulation would look like with a PCB imported in. There are five main areas of the interface that will be covered in the following sections in detail. As a brief overview: each section handles a different part of the simulation and allows users to either give inputs to fully customize the simulation and save any data necessary or run and manage the simulation. Each simulation comes with a scale grid, which are the black lines laid across the board and the entire Scene. These grids are all 1x1 cm squares.

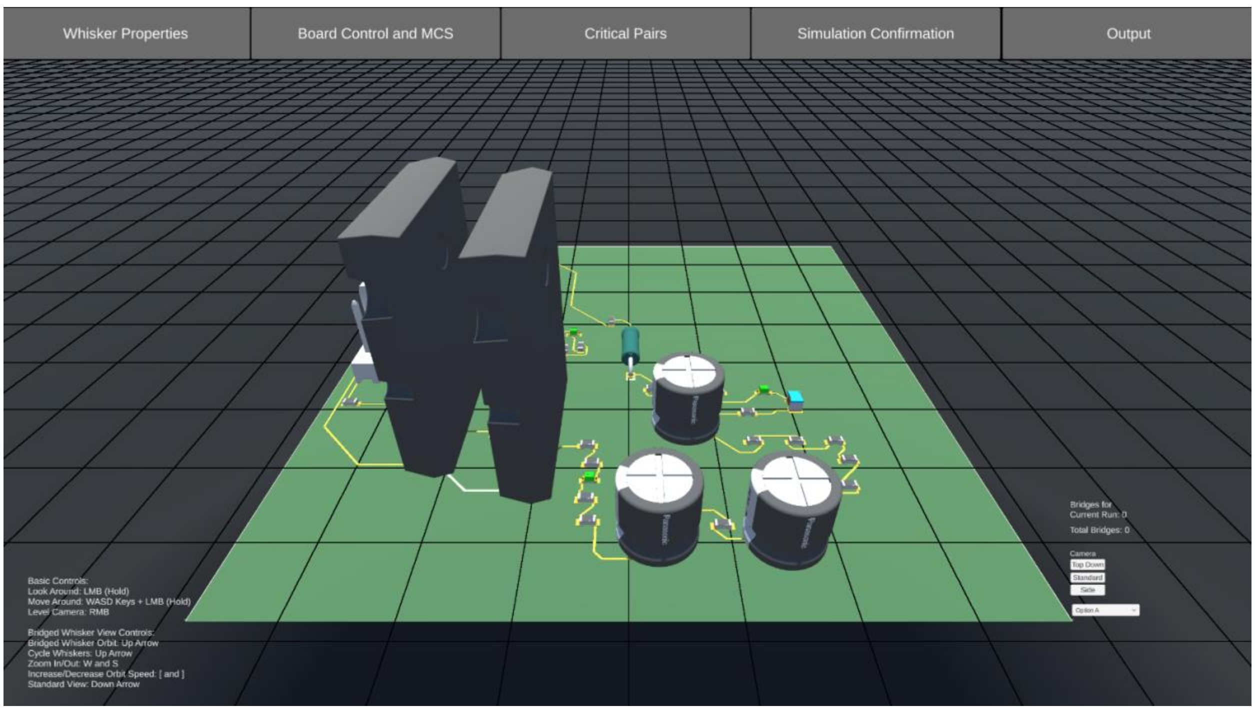


Figure 16: Simulation Interface

Tab 1: Whisker Properties

This section of the interface handles all of the user inputs required to generate whiskers into the Scene, along with the input field that allows users to tell the program what materials on their board are to be seen as conductive. Below is a table giving a description of what each part of the Interface does:

Whisker Properties Inputs	
Length Mu [µm]	Takes in the value and uses it as the location parameter or the mean value for the Lengths of the whiskers depending on if the distribution is Lognormal or Normal.
Length Sigma [µm]	Takes in the value and uses it as the location parameter or the mean value for the diameters of the whiskers depending on if the distribution is Lognormal or Normal.
Width Mu [µm]	Takes in the value and uses it as the location parameter or the mean value for the diameters of the whiskers depending on if the distribution is Lognormal or Normal
Width Sigma [µm]	Takes in the value and uses it as the scale parameter or the standard deviation for the diameters of the whiskers depending on if the distribution is Lognormal or Normal
# of Whiskers [µm]	Sets how many whiskers are to be generated.
Whisker Material	A dropdown list that allows users to select from Tin, Zinc, or Cadmium, changing the base properties of the whiskers.
Distribution Selection	A dropdown list that sets what distribution type is to be used for the generation of the whiskers. Defaults to Lognormal.
Conductive Material Selection *	A text field that when material names are entered in, the program searches the components of the board for the specific material listed, and if it finds a match, it labels that object as Conductive.
Whisker Spawn Point X, Y, Z	A text field that alters the area in which the whiskers can be randomly generated. X goes into and out of the screen, Y goes left and right of the screen, and Z goes up and down of the screen. Units are in centimeters
X, Y, Z Coordinate	A text field that alters the center of the whisker spawn area mentioned above in the X, Y, and Z directions, with units in centimeters
Toggle Spawn Area	Checking this box will make the spawn area visible, and will make the alterations made in

	the spawn point and coordinates visible to the user without obstructing visibility of the simulation

a. The values for your Mu and Sigma inputs can now be inputted in micrometers and choose your number of whiskers, along with selecting any desired options from the dropdown lists. Verify that these parameters are in line with the distribution being, as the whiskers may not spawn in or be visible depending on the size of the inputs and the distribution being selected. The whisker material may now be selected from the dropdown as well.
b. Then the conductive material can be set. This will be done by entering the name of the material into the input field, and then clearing the field before entering in more conductive materials if there are multiple. This can handle any number of conductive materials, but once the conductive materials are set, they cannot be undone except by exiting Play mode and refreshing the simulation.
c. The whisker spawn area can be set via the Whisker Spawn Point X, Y, and Z input fields and can be made visible by toggling the Whisker Spawn Area box. The area is centered around a point, which is the red sphere is at the bottom of the center point. The center area if the Whisker Spawn Point can be moved via the X, Y, and Z coordinate input fields which allows for the spawn area to be moved anywhere that is desired.

*NOTE: Many outside programs used for creating models of circuit boards do not handle the materials of the components very well, regardless of which program is being used. Often times, instead of a material being labeled correctly (e.g. copper), it will be labeled as mat_12, or a variation of that format, which will then be how it appears in Unity. It is important to scan through your board after importing and see what material names have been assigned to properly use this program. These could change depending on the program the board gets exported from. To view these materials, open the Boards folder in the Project tab, under Assets. Click the small arrow next to your board. This will open up a list of all the components on the board and any materials that were imported alongside.

Tab 2: Board Control and MCS

This section of the interface all board manipulations such as Shock, Vibration, Gravity, Walls/Ceiling, and Board Rotation. This section also gives control over the Monte Carlo Simulation (MCS). The two tables below describe the Board Manipulation and MCS functions.

Monte Carlo | Data Storage Inputs

# of Iterations	Tells the simulation how many iterations to run before coming to a stop.
Directory	Input field for users to place their specific/desired location for their data on their device.
File Name	Input field for users to set the file name of their .csv sheet, which will contain the simulation data.
Save Button	Saves the Directory and File Name inputs into the code.
Heatmap Toggle	Toggles the Heatmap on or off.
Save Inputs	Saves the user inputs into the file with the MCS and Bridging report
Load Inputs	Allows the user to load in a set of inputs used by a previous simulation using the output file from said simulation.

Board Manipulation	
Select Gravity	Allows users to choose from 4 gravity settings: Earth, Mars, Moon, and Custom. Custom Gravity allows the user to input values (in $\frac{m}{s^2}$) for "Gravitational Forces" in the X, Y, and Z directions
Add Walls	Toggle to add walls and a ceiling, based user input. This wall will be made around the perimeter of the PCB
Wall Height and Ceiling Toggle	Text field that allows users to input a wall height in millimeters. When the ceiling is toggled, clicking the "Add Walls" button will also add a cap on the walls.
Rotate Board X, Y, and Z	Text field that allows user to input a value for degrees of rotation of the PCB in the X, Y, and Z directions.
Spin Board X, Y, and Z	Text field that allows user to input a value for degrees of rotation per second of the PCB in the x, y, and z directions.
Shock	Triggers a shock based on user inputs
Vibration	Triggers a vibration based on user inputs

a. The number of iterations for the simulation to run through can be set. It is recommended to enter in “1” for this field until the whiskers have their desired dimensions and spawning area.
b. Enter in a directory on your personal device and a file name for the csv file to give the simulation a location to store the data into. Press Save once ready. If an incorrect directory or file name is entered, reenter the correct name and press Save again.
c. Optionally, click the Heatmap Toggle to turn on the Heatmap. The components containing the conductive materials you have given in **Section 2-b** will be highlighted in green, and a scale of the gradient will appear on the screen, ranging from 0-5+ whiskers, moving from light green to dark red. More details about this feature can be found in **Running the Simulation: Section 6.**
d. Gravity can be changed and customized to mimic any extreme or non-parallel gravity force. Click the gravity dropdown, the options are Earth, Moon, Mars, and Custom Gravity, and upon choosing any planetary body the gravity will mimic the desired choice. Upon choosing Custom Gravity three more input fields will appear Custom Gravity X, Y, and Z which any value can be inputted in any or all three fields which allows for customized gravity.
e. Walls and a ceiling can be added to the simulation with users desired height as the input. When “Add Walls” is pressed, the simulation generates a semi-transparent wall stuck to the perimeter of the CCA. If the optional “Ceiling” toggle is checked, a semi-transparent ceiling will the cap the walls that are created. Both the walls and ceilings will interact with whiskers and contain them within the bounds of the CCA and the walls. As the wall is attached to the CCA all board manipulations will also affect the walls and ceiling.
f. The board can be given an initial rotational offset and also given a spinning rate for each iteration. The board resets to the initial rotation value at the beginning of each iteration. The initial rotation value can be inputted by the Rotate Board X, Y, and Z input fields which when the value is inputted the board rotates automatically to the desired rotation. The spinning rate can be set by the Spin Board X, Y, and Z input fields which when the values are inputted the board will automatically start spinning at the desired rates to show the user the speed.
g. The simulation can subject the board to shock and vibrations, where the user designates an amplitude (both shock and vibrations) in newtons, as well as a frequency (vibration only) in Hz, for a given duration (vibration only). Users can check the “Use in Simulation” toggle for both options, and the simulation will shock or vibrate the board automatically for each iteration when a simulation is run. Optionally, the user can trigger the shock or vibration by pressing the “Shock” or “Vibration” button.

Tab 3: Critical Pairs

This section allows users to choose 2 conductors as critical pairs from their PCB to get easier access to information and statistics related to the chosen conductors. While this function does not necessarily provide more information within the simulation, the output report and the accompanying excel analysis file has sections that output statistics regarding these specific critical pairs. The table below describes the critical pair functions.

Critical Pairs	
Conductor 1	Dropdown for users to choose the first of two conductors as a critical pair
Conductor 2	Dropdown for users to choose the second of two conductors as a critical pair
Refresh	Button to refresh the two conductor dropdowns
Add Critical Pair	Button for user to add their two chosen conductors as a critical pair
Remove Critical Pair	Button for user to remove their two chosen conductors as a critical pair
Critical Pairs Panel	Scrollable Panel for users to view added critical pairs

a. For the dropdowns to populate with the list of conductors, all settings in the previous two tabs must be input. After all inputs in the previous two tabs are entered, the refresh button can be pressed to populate the dropdowns
b. There is a possibility of selecting the same conductors in both dropdowns and adding it as a critical, so ensure the two conductors chosen are different.
c. To remove a critical pair, the conductors must be picked within the dropdowns in the order in which they are shown.
    a. For example, if in the conductor panel a critical pair is shown as “3.3V – GND”, Conductor 1’s dropdown must be 3.3V and Conductor 2’s dropdown must be GND to remove the critical pair

Tab 4: Simulation Confirmation

This section of the Interface allows users to spawn in whiskers, reload them to better fit to the PCB, begin and stop the simulation. Below is the table describing what each interface object handles in this area:

Simulation Confirmation Controls	
Spawn Whiskers	Initially spawns in whiskers based off all of the previously described inputs.
Reload	Allows users to load fresh whiskers into the Scene after making a change to either their

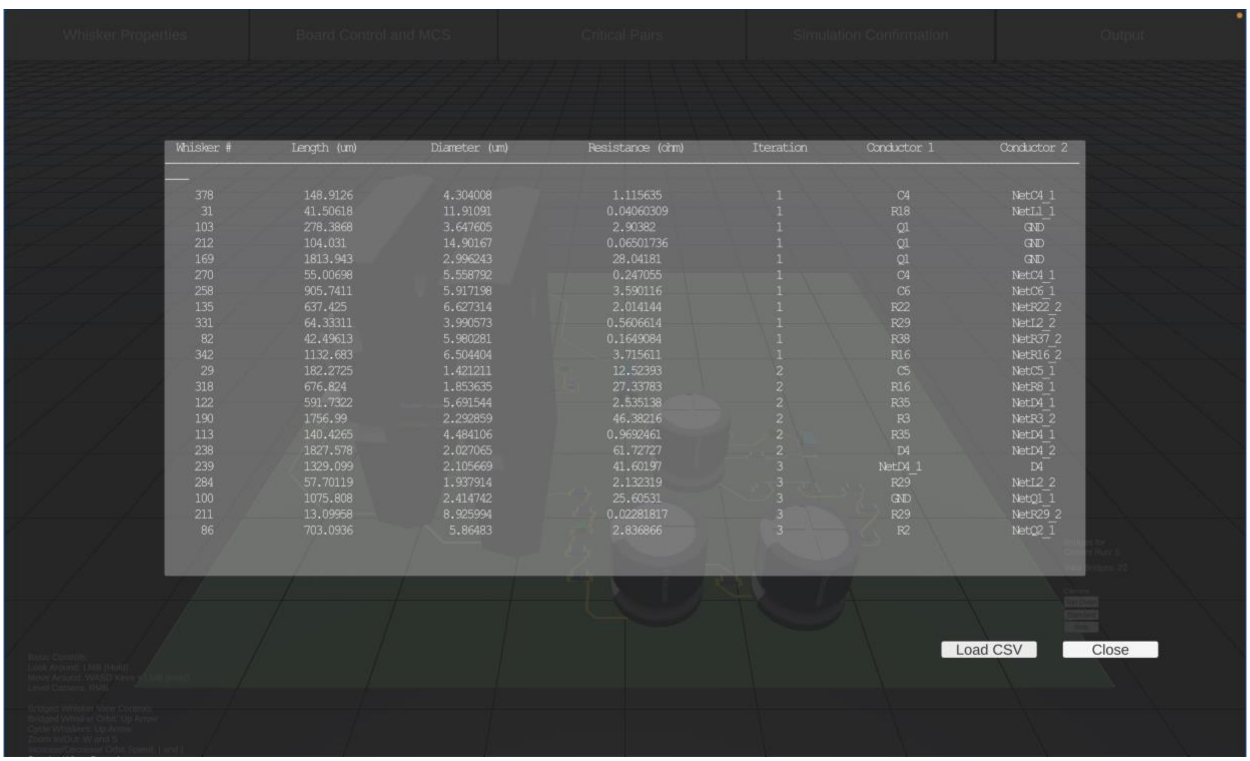
	dimensions, number in the Scene, or coordinate location over the board.
Start Simulation	Confirms the gravity and officially starts the simulation. Pressing this will run the simulation through all the iterations desired.
Reset Gravity	Turns off the gravity, and the iteration counter, which allows the whiskers or data storage to be changed to different parameters.
Reset Sim	Clears and stops the iteration counter, and clears bridges formed – total and per run.
Hide UI	Toggle that hides the user inputs.
Hide Grid/Raise/Lower Grid	Series of a toggle and buttons that allow the user to turn off or on the scale grid, along with raise or lower it to better fit their PCB.
Screenshot	Button for users to take a screenshot of the simulation with all user interface elements removed at any time
Enable Automatic Screenshots	Toggle for users to allow the simulation to automatically take screenshots as described above after the end of each iteration

a. Once the previous sections and steps are completed, all inputs should be filled in and any selections from the dropdown lists should have been made. The Start button should now be pressed to begin the simulation. This will spawn in the whiskers based off the parameters chosen.
b. Next, you should verify that the whiskers are spawning in with the desired dimensions over the appropriate areas. This can be done initially by clicking “Spawn Whiskers”. If any parameter must be changed, they can be altered and pressing the Reload button will spawn new whiskers into the scene based off of the new parameters. Repeat this step until the whiskers are generating over the entire board and are the appropriate size for your simulation needs.
c. If the whiskers are correctly sized and generating over your PCB, confirm that the directory and file path are as desired, along with the number of iterations for the simulation to run through.
d. Finally, press the Start Simulation button to confirm all settings, and fully begin the simulation. The whiskers should now be dropping towards the board, and every 5 seconds, they will be cleared along with generating a new set with the same parameters you have entered in. The whiskers creating bridges will be highlighted red and will be tracked, while the number of bridges being made will be printed to the screen on the right. The data from the simulation will be stored into the file name you have provided.

e. This section also contains the screenshot feature. Pressing the button will hide all user interface elements and take an image of the program, with the board in the exact orientation the user left it. This can be done automatically for each iteration by checking the "Enable Automatic Screenshot" toggle.
f. Optionally, the UI (camera controls and instructions panel) can be hidden or shown by checking the "Hide UI" toggle.
g. By default, a 1 cm by 1 cm grid is displayed just under the CCA in the simulation. This simulation can be moved up, down, or hidden using the corresponding buttons in this tab.

Tab 5: Output

This section of the user interface shows an abridged version of the output file described in the section titled "**Tab 2: Board Control and MCS**". When clicked, a panel is opened with information pertaining to bridged whiskers and the data related to them including whisker number, length, diameter, resistance, iteration, conductor 1, and conductor 2. Units for measurements are displayed in micrometers, resistance is shown in ohms, and conductors 1 and 2 display the two conductors that bridged with the whisker. Figure 17 below shows and example output panel.



Whisker #	Length (um)	Diameter (um)	Resistance (ohm)	Iteration	Conductor 1	Conductor 2
378	148.9126	4.304008	1.115635	1	C4	NetC4_1
31	41.50618	11.91091	0.04060309	1	R18	NetL1_1
103	278.3868	3.647605	2.90382	1	Q1	GND
212	104.031	14.90167	0.06501736	1	Q1	GND
169	1813.943	2.996243	28.04181	1	Q1	GND
270	55.00698	5.558792	0.247055	1	C4	NetC4_1
258	905.7411	5.917198	3.590116	1	C6	NetC6_1
135	637.425	6.627314	2.014144	1	R22	NetR22_2
331	64.33311	3.990573	0.5606614	1	R29	NetL2_2
82	42.49613	5.980281	0.1649084	1	R38	NetR37_2
342	1132.683	6.504404	3.715611	1	R16	NetR16_2
29	182.2725	1.421211	12.52393	2	C5	NetC5_1
318	676.824	1.853635	27.33783	2	R16	NetR8_1
122	591.7322	5.691544	2.535138	2	R35	NetD4_1
190	1756.99	2.292859	46.38216	2	R3	NetR3_2
113	140.4265	4.484106	0.9692461	2	R35	NetD4_1
238	1827.578	2.027065	61.72727	2	D4	NetD4_2
239	1329.099	2.105669	41.60197	3	NetD4_1	D4
284	57.70119	1.937914	2.132319	3	R29	NetL2_2
100	1075.808	2.414742	25.60531	3	GND	NetQ1_1
211	13.09958	8.925994	0.02281817	3	R29	NetR29_2
86	703.0936	5.86483	2.836866	3	R2	NetQ2_1

Figure 171: Example Output Panel

a. By default, the output panel will say "No CSV Found". The data can be displayed by clicking on the "Load CSV" button in the bottom right of the panel. This button should be pressed only after a simulation is ran. The output may not accurately display the information if multiple simulations were run in succession.

b. To exit this panel and return to the simulation, press the “Close” button in the bottom right.

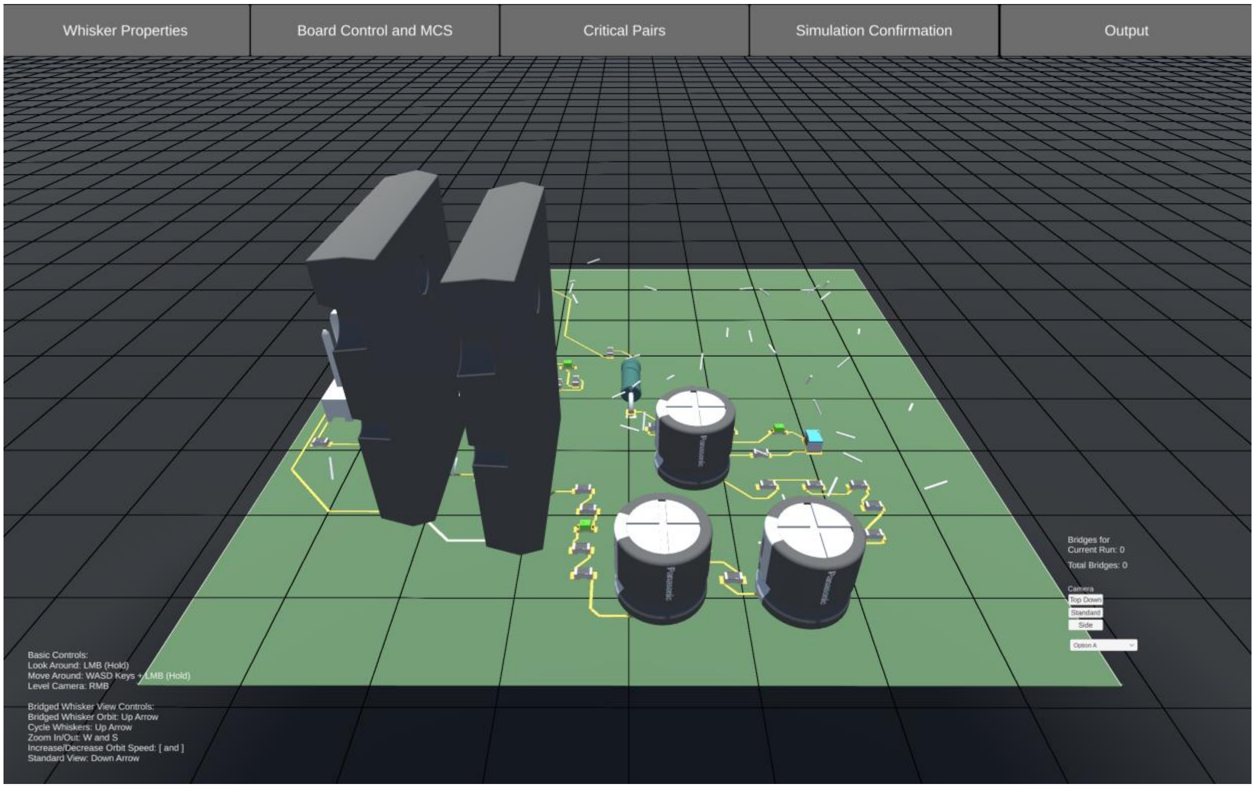


Figure 18: Simulation Running

Section 2: Heatmap

Once your simulation has been completed. A message will appear on the left side of the screen in green: “Simulation Complete!”. The Heatmap toggle located in the **Monte Carlo | Data Storage** section of the Interface can be turned on if it has not been already. Post-simulation, assuming there were some bridges formed, some components will have changed colors representing how many bridges each component contributed to making. This allows a visual of what components are most likely to fail due to whisker bridging.

Section 3: External Forces

As mentioned in Section 5, there are buttons in the user interface that open the external force input fields. To use them properly, please refer to the following steps:

Shock | Impact

a. To apply the mechanical shock or impact force to the board, first press the Board Control and MCS tab which will drop down the correlated tab. There will be an input field for the value of force, a dropdown list that allows the force to be applied to any axis, a toggle, and a new Shock button underneath the dropdown.
b. *Enter in a value for the Amplitude (N) text field. This will tell the program how many Newtons of force to apply to the board. Decimal and negative values are accepted. Entering in a negative value will push the board in the opposite direction.
c. Select an axis to apply the force to through the dropdown. The options are X, Y, and Z.
d. Finally, press the Shock button to apply the shock. This will apply the force to the board based off the input and direction given.
e. The toggle can then be checked. Clicking this will run the shock continuously, using the inputs provided by the user, while running multiple iterations of the simulation.

* NOTE: any shock value over 10 N could cause whiskers to be pushed dramatically in the direction of choice. It is highly recommended to operate in the 0-10 N range.

Vibration

a. To apply a vibration to the board, first press the Board Control and MCS tab which will drop down the correlated tab. Similarly to the shock, there will be an input field for the Amplitude of the vibration, a dropdown list that can select the axis to apply the vibration, a toggle, a new vibration button, and inputs for the Frequency and Duration of the vibration.
b. Enter in values for the Amplitude, Frequency, and Duration*. A smaller amplitude (e.g. 0.2) and a larger frequency (e.g. 5-10) will provide a proper vibration, whereas a larger amplitude with a smaller frequency will provide a much smoother and wider vibration. The amplitude can be a negative value, while the Frequency and Duration cannot be.
c. Select an axis to apply the vibration on from the dropdown list.
d. Press the vibration button to apply the vibration.
e. Check the toggle to on to apply the vibration across multiple iterations when running the simulation fully.

*NOTE: Set the vibration duration to 5 seconds or less before pressing Start Simulation. Failure to do so will result in the vibrations overlapping and moving the board out of place.

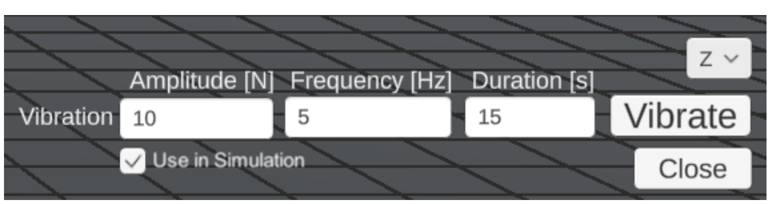


Figure 19: Sample Vibration Input

Custom Gravity

a. To apply the custom gravity to the board, first press the Board Control and MCS tab which will drop down the correlated tab, and then click the select gravity drop down and then click the Custom Gravity option. There will be three input fields, one for Custom Gravity X, one for Custom Gravity Y, and Custom Gravity Z, which apply the gravity force inputted in the correlated coordinate direction.
b. Enter in a value for any or all text fields. This will tell the program how many meters per second square of force to apply to the whiskers. Decimal and negative values are accepted. Entering in a negative value will push the whiskers in the opposite direction.
c. Select an axis to apply the force to through the dropdown. The options are X, Y, and/ or Z.
d. Finally, press the Start Simulation button or Spawn Whisker button and the whiskers will have the custom gravity applied to them.

Miscellaneous User Interface

Aside from the tabs, there are two other main sections on the user interface: the instructions panel in the bottom left and the camera/bridging interface in the bottom right.

Instructions Panel

The bottom left corner contains most keypad instructions as shown in Figure 20

Figure 20: Instructions Panel

The instructions panel contains basic controls information for the user to use when in different camera modes.

For basic camera movement, the user can use the left mouse button and pan (as stated before) to look around the simulation. The camera itself can be moved around using the WASD keys while holding the left mouse button, and the camera can be leveled using the right mouse button or the down-arrow key.

Bridged Whisker View controls apply when viewing bridged whiskers close up. This view can be entered by pressing the up-arrow key. By default, the camera will orbit around the bridged whisker. The whisker of focus will be shown in the bottom right corner and will be elaborated on later. The speed at which the camera orbits around can changed using the '[' key to slow down, and the ']' key to speed it up. The camera can be moved closer to the whisker by pressing the W key, and the camera can be moved further away using the S key. The up-arrow can be pressed again to cycle to the next whisker (ordered numerically). The user can exit the bridged whisker view by pressing the down-arrow key.

Camera/Bridging Interface

This interface can be found in the bottom right corner and can be seen in Figure 21.

Figure 21: Camera/Bridging Interface

In this section, there are three main sections: Bridging Statistics, Camera Controls, and Bridged Whisker Dropdown. The bridging statistics show the number of bridges for a given set of whiskers spawned. Following it is the sum total of all the bridges throughout the entire simulation. This "Total Bridges" value can be reset by clicking the "Reset Simulation" button.

The next portion are the camera controls. When in the standard view, these buttons can be pressed to move the camera to "Top Down", "Standard", or a "Side" view centered around the entire CCA. When in the bridged whisker view, these buttons will move the camera in the mentioned view centered around the bridged whisker.

The bridged whisker dropdown will be updated after a simulation is ran and be populated with a list of bridged whiskers shown in numerical order. Selecting a whisker will shift the camera into the bridged whisker view if in standard view, or switch the whisker selected if already in bridged whisker view.

Section 4: Resetting the Simulation | Saving New Data

To reset your simulation, start a new set of iterations, or save a new set of data, you will have to reset the simulation. There are a few ways to do this. The first method involves using the Reset Sim button located near the bottom of the screen when viewing the Interface.

a. Click Reset Gravity. This will set the gravity back to 0 and prevent the iterations from running until any new changes to the inputs are made and ready, along with preventing any excess or unwanted data from being stored.
b. (Optional) Clear the Directory and File Name input fields and press Save.
c. Set the Number of Whiskers field to 0 and Press Reload to remove the whiskers from the surface of the board, preventing premature bridging detection.
d. Click the Reset Sim button. This will clear the bridges detected and the iteration counters so a new run can be completed.
e. Enter new parameters into the input fields as needed.
f. Fill in the Directory and input a new File Name to store the new data set into. Press Save. (e.g. if the first .csv file was titled ‘whisker_data’, name the next one something new like ‘vibrationOnData’ or ‘whisker2_data’)
g. Press Reload with the new or original inputs until whisker are desired size and locations.
h. The simulation should now be ready to be run again by pressing Start Simulation.

Alternatively, you can clear all inputs and values and fully reset the simulation by simply pressing the Play button at the top of your Unity Editor to fully stop the program, and then pressing it again to reopen it.

Data Processing

Section 1: Using whiskerOutputAnalysis.xlsx

After the data has been saved into your file of choice, you may navigate to the directory you have provided and open the file. You will see that the data has been saved and formatted. Additionally, in the program files that were saved to your device previously from GitHub, there was an Excel file titled whiskerOutputAnalysis. This .xlsx file is equipped with instructions to allow for more in-depth results from the simulation to be generated and viewed.

Figure 22 shows an example of what the data might look like when stored into the .csv file you have generated. Notice that the file keeps track of every whisker that is generated, along with its dimensions, resistance, and what iteration it was generated in. The same applies to the whiskers that created a bridge. *Data values in the example are exaggerated.

	A	B	C	D	E
1	All Whiskers				
2	Whisker #	Length (un	Width (um	Resistance	Iteration
3	1	2921.423	57.96461	0.120672	1
4	2	5753.229	72.59191	0.151521	1
5	3	6390.803	52.62613	0.32025	1
6	4	2826.367	45.6218	0.188461	1
7	5	2061.689	54.36802	0.096799	1
8	6	2657.678	62.38023	0.094786	1
9	7	1675.75	51.51289	0.087642	1
10	8	5369.641	60.25764	0.205238	1
11	9	6449.581	33.88324	0.779648	1
12	10	2589.204	66.38473	0.081539	1
13	11	2007.121	60.37473	0.076419	1
14	12	2974.402	32.73833	0.385144	1
15	13	1055.391	54.58115	0.049166	1
16	14	1913.361	55.89381	0.084997	1
17	15	3638.534	50.61422	0.197114	1
18	16	3574.214	47.36576	0.2211	1
19	17	2703.053	64.70096	0.089613	1
20	18	8189.121	65.97678	0.261091	1
21	19	3675.912	47.84765	0.222834	1
22	20	6027.659	42.15321	0.470786	1
23	21	4114.887	52.81493	0.20473	1

	G	H	I	J	K	L	M
1	Bridged Whiskers						
2	Whisker #	Length (un	Diameter (	Resistance	Iteration	Conductor	Conductor 2
3	318	3272.759	51.13706	0.173692	1	GND	Q1
4	352	4102.102	55.02091	0.188056	1	GND	Q1
5	26	4421.396	53.90409	0.21118	1	Q1	GND
6	43	3747.569	56.62005	0.162236	1	R28	NetD3_1
7	56	2372.752	45.65566	0.157979	1	C4	NetC4_1
8	198	6977.418	59.15441	0.276731	1	Q2	NetQ2_2
9	117	6817.568	51.01569	0.363545	1	GND	NetQ1_1
10	9	6449.581	33.88324	0.779648	1	NetR11_2	C2
11	381	2518.593	63.03785	0.087961	1	R30	NetR29_2
12	368	1159.62	62.50555	0.041192	1	R11	NetR11_2
13	66	2761.829	66.48866	0.086704	1	Q1	NetQ1_2
14	274	3592.661	43.09268	0.268501	1	NetR11_2	R15
15	260	3992.121	59.58002	0.156077	1	NetR7_2	NetC1_1
16	240	2707.194	62.07954	0.09749	1	R9	NetR8_2
17	387	7638.089	50.51381	0.415433	1	GND	C6
18	30	6010.599	65.75401	0.192935	1	C1	R13
19	1	2921.423	57.96461	0.120672	1	R14	NetR13_2
20	2	5753.229	72.59191	0.151521	1	NetC5_1	C5
21	178	5572.637	46.1104	0.363747	1	GND	NetQ1_1
22	49	1550.037	53.73452	0.074503	1	D4	NetD4_1
23	341	2707.687	50.81097	0.145553	1	NetD4_1	NetR35_2

	O	P	Q	R	S	T	U
1	Critical Bridged Whiskers						
2	Whisker #	Length (un	Diameter (	Resistance	Iteration	Conductor	Conductor 2
3	120	3336.264	91.95851	0.054754	4	R35	R36
4	271	4621.871	51.9052	0.238085	4	C1	NetC1_1
5	112	2761.398	60.62378	0.104275	9	R35	R36
6	167	6544.978	59.42219	0.257245	14	R35	R36
7	206	5232.003	46.96705	0.329168	15	R35	R36

	W	X
1	Simulation Inputs	
2	Parameter	Value
3	Length Mu	8
4	Length Sig	0.5
5	Width Mu	4
6	Width Sign	0.2
7	# Of Whisk	400
8	Conductiv	copper
9	Whisker M	Tin
10	Distributio	LogNormal
11	Whisker Sp	0
12	Whisker Sp	0
13	Whisker Sp	0
14	X-Coord	3
15	Y-Coord	1
16	Z-Coord	3
17	Gravity	Earth
18	# of Iteratio	30
19	Directory F	C:\test
20	Save File N	DemoDataLog2
21	Rotation-X	-90
22	Shock Acti	N
23	Vibration A	N

Figure 22: CSV File Data

To take this data and get the results properly, refer to the following instructions for guidance:

a. Open whiskerOutputAnalysis.xlsx.
b. Open the file containing your simulation data.
c. Copy and paste the values from your data set to the designated location in whiskerOutputAnalysis. Each section should be pasted in separately (All Whisker, Bridged Whiskers, etc.)
d. Can easily select all data to copy by selecting the top left cell and then while holding shift select the top right cell in that section. Then press 'ctrl+shift+down arrow' to select all cells.
e. Once all of the values are pasted into the proper location, there will be instructions listed inside of the file that give details on how to properly analyze the data.

Copy and Paste Unity Data Below*

Bridged Whiskers

Whisker #	Length (um)	Width (um)	Resistance (ohm)	Iteration	Conductor 1	Conductor 2	Bridged Pair
318	3272.759	51.13706	0.1736917	1	GND	Q1	GND + Q1
352	4102.102	55.02091	0.1880561	1	GND	Q1	GND + Q1
26	4421.396	53.90409	0.2111799	1	Q1	GND	Q1 + GND
43	3747.569	56.62005	0.1622355	1	R28	NetD3_1	R28 + NetD3_1
56	2372.752	45.65566	0.1579791	1	C4	NetC4_1	C4 + NetC4_1
198	6977.418	59.15441	0.2767306	1	Q2	NetQ2_2	Q2 + NetQ2_2
117	6817.568	51.01569	0.3635453	1	GND	NetQ1_1	GND + NetQ1_1
9	6449.581	33.88324	0.7796476	1	NetR11_2	C2	NetR11_2 + C2
381	2518.593	63.03785	0.08796133	1	R30	NetR29_2	R30 + NetR29_2
368	1159.62	62.50555	0.04119219	1	R11	NetR11_2	R11 + NetR11_2
66	2761.829	66.48866	0.08670381	1	Q1	NetQ1_2	Q1 + NetQ1_2
274	3592.661	43.09268	0.268501	1	NetR11_2	R15	NetR11_2 + R15
260	3992.121	59.58002	0.156077	1	NetR7_2	NetC1_1	NetR7_2 + NetC1_1
240	2707.194	62.07954	0.09748974	1	R9	NetR8_2	R9 + NetR8_2

Figure 23: Example of Bridged Whisker Data in whiskerOutputAnalysis

f. Everything within the **General Analysis sheet** will be automatically calculated as seen in Figure 24.

g. Every header with **any asterisk (*) in Figure 23 will give a brief description in the Excel file of the section.**

General Analysis Sheet

Bridged Probability*	
Overall Probability*	
100.00%	
Individual Probability*	
Iteration #	Probability
1	8.00%
2	6.25%
3	8.25%
4	5.50%
5	7.50%
6	6.75%
7	6.75%
8	11.00%
9	8.50%
10	7.75%
11	6.75%
12	7.50%

Critical Bridged Probability*	
Overall Probability*	
13.33%	
Individual Probability*	
Iteration #	Probability
4	0.50%
9	0.25%
14	0.25%
15	0.25%

Summary Statistics*	All Whiskers	Bridged Whiskers	Critical Bridged Whiskers
Average Length (um)	3386.3	4190.6	4499.3
Average Width (um)	55.8	56.3	62.2
Average Resistance (ohm)	0.2	0.2	0.2
Maximum Length (um)	22522.4	22522.4	6545.0
Maximum Width (um)	112.9	99.8	92.0
Maximum Resistance (ohm)	1.5	1.4	0.3
Minimum Length (um)	426.7	1030.2	2761.4
Minimum Width (um)	24.9	28.1	47.0
Minimum Resistane (um)	0.0118	0.0217	0.0548

	Bridged Whiskers	Critical Bridged whiskers
# of Iterations with 0 Bridges	0	26
# of Iterations with at least 1 Bridge	30	4

Figure 24: General Analysis Sheet

h. In the **Bridged Whisker Analysis sheet** right click on the Bridged Pair Analysis table (Figure 25) and click refresh. This should update and load in your data.

i. Next click on the filter symbol in the table, it is the symbol next to the Bridged Pairs header, and make sure all the data you want is selected.

j. Under the Filters column there are Excel slicers where you can choose specific conductor pairs and iterations. Hold the ctrl button while clicking to select multiple. The plot will adjust automatically.

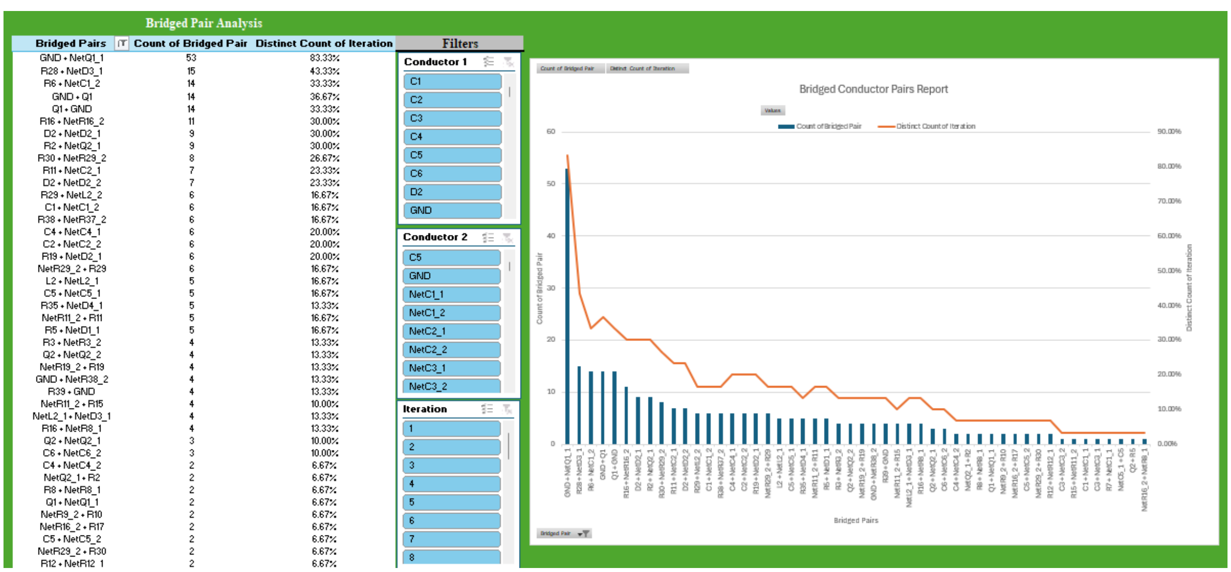


Figure 25: Bridged Pair Analysis Section

k. Similar to the previous steps, click on the table and click refresh. Then click the filter symbol and make sure all your data is selected.
l. Under the Filters column there are Excel slicers where you can choose the number of bridges and number of iterations. Hold the ctrl button while clicking to select multiple.

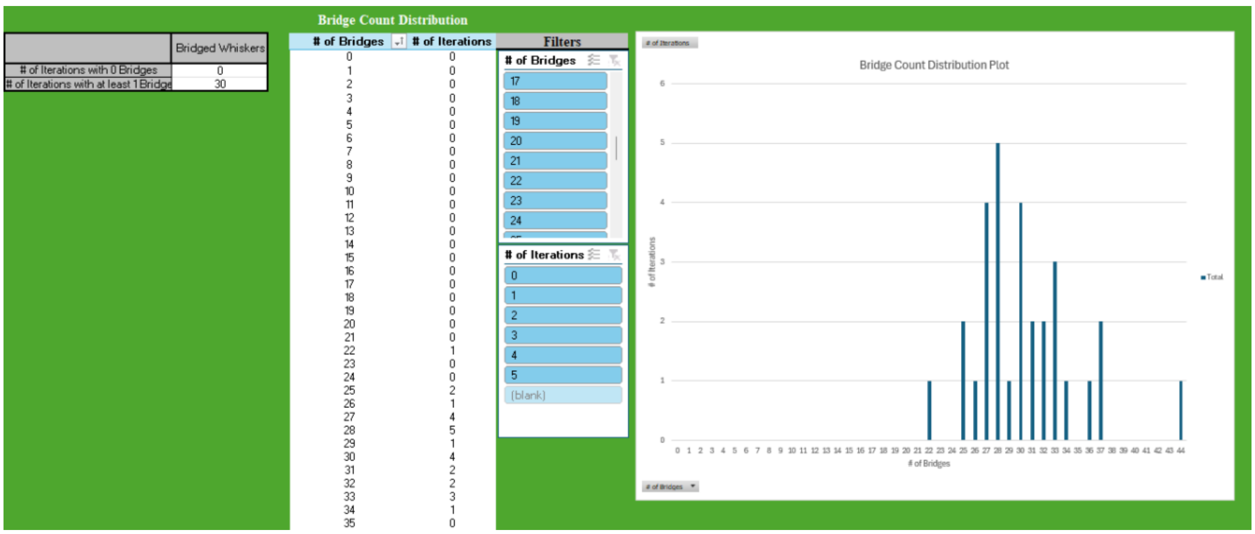


Figure 26: Bridge Count Distribution Section

m. In the **Critical Bridged Whisker Analysis sheet** right click on the Critical Bridged Pair Analysis table (Figure 25) and click refresh. This should update and load in your data.
n. Next click on the filter symbol in the table, it is the symbol next to the Bridged Pairs header, and make sure all the data you want is selected.
o. Under the Filters column there are Excel slicers where you can choose specific conductor pairs and iterations. Hold the ctrl button while clicking to select multiple. The plot will adjust automatically.

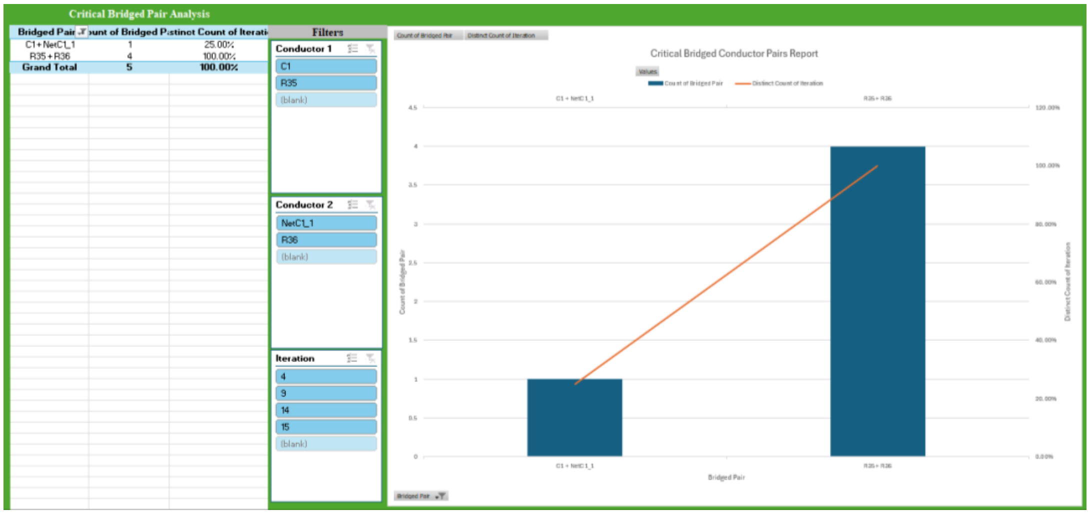


Figure 27: Critical Bridged Pair Analysis Section

p. Similar to the previous steps, click on the table and click refresh. Then click the filter symbol and make sure all your data is selected.
q. Under the Filters column there are Excel slicers where you can choose the number of bridges and number of iterations. Hold the ctrl button while clicking to select multiple.

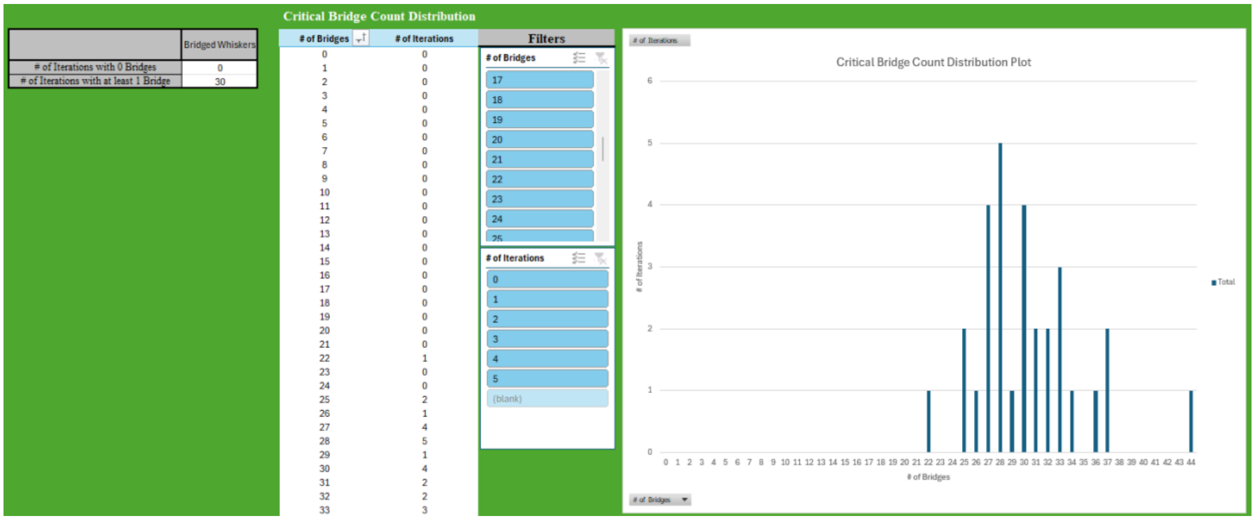


Figure 28: Critical Bridged Count Distribution Section

Extra Features

Section 1: Filtering Objects Based on Material

This is a built-in feature that the Unity Editor offers. What this allows users to do is when in the Scene view (Simulation is not running, users are examining their board or fixing orientation), they can filter objects based on the materials the components are made of. This can be useful when figuring out which components need to be labeled as conductive, as is done in **Running the Simulation, Section 2.**

a. To do this, navigate to your Project tab, into the Assets folder, and then to the Boards folder where your Board should be stored.
b. Press the small arrow to the right of your PCB icon to open up all components of the board.
c. Find any material you wish to see highlighted, and right click to bring up a new menu.
d. Click on Find References in Scene. This will highlight all objects containing the material you selected.

Section 2: Changing Iteration Time Limit

Users can change the time limit between iterations simply through the use of the Unity Editor. The original time limit is 5 seconds but can be increased or decreased depending on the situation at hand. To do this, follow the below instructions:

a. In the Editor, under the Hierarchy, click on the UI object.
b. In the Inspector will be a large section containing many fields and objects. Scroll until a field titled Sim Time Thresh is present.
c. Change this from 5 to any number desired.

The sim time will now have a different time step between iterations, increasing or decreasing the speed of the simulation.

Beta Testing

For ease of access to end-users or beta-testers, an online simulation build or a desktop .exe file can be created using Unity. To do so follow the steps listed below:

<u>WebGL build:</u>

In conjunction with GitHub, users can create a web-accessible version of the simulation that cannot be altered unless the GitHub files are updated. The steps to create the build is as follows:

1. Save all changes within Unity and Visual Studios
2. Go to Files > Build Settings and select WebGL as the platform on the left panel. (By default, a build for your OS will be selected, and it may prompt you to install a plugin if it's your first time).
3. Enable Player Settings > Publishing Settings > Decompression fallback
4. On the bottom of the window, click "Build" and save the folder with a name of your choosing.
5. Open the GitHub repository.
6. If this is the first upload of a Beta version, you will need to create a public repository.

7. Once the files are saved, open the folder created by the build and upload the contained files within the new folder individually (Build, index.html, and Template Data).
8. Commit and push the change
9. Wait for page development (may take up to 5-10 minutes)
   - You can check if page development is finished by going to "Actions" in the top bar and checking that the most recent action is marked with a green circle.
   - The link to access the website can also be found in the "Actions" tab in the bar
10. For new editions of the beta, follow the steps within Unity and in the GitHub, upload the new files, commit, and push. Page development may take up to 5-10 minutes.

.exe/.app build:

1. Save all changes within Unity and Visual Studios
2. Go to Files > Build Settings and select "Windows, Mac, and Linux" as the platform on the left panel. (By default, a build for your OS will be selected)
3. Leave all default settings and press "Build"
4. Unity will prompt you to select a name and location for your simulation file
5. The .exe (Windows) or .app (Mac) will be saved in the designated directory and can be opened from this location to run the simulation.

Quick Overview

This section of the manual was designed to give users who are familiar with the simulation and already have it saved into their Unity Editor a brief overview of the instructions as a reference or reminder.

a. Import your board into the scene: Drag and drop your board from the file explorer on your personal device into the Boards folder in the Project tab, under Assets.
b. Orient the board if needed using the rotation parameters in the Inspector.
c. Add the Trigger Control script to the imported board and assign the Trigger Material and Material Input objects in their respective fields.
d. Press the Play button to run the simulation.
e. Fill in your Whisker Properties and search the board for any particular materials using the Conductive Material Selection input field.
f. Fill in your Simulation Control Parameters and Simulation Manipulation.
g. Set your desired number of iterations.
h. Press Start
i. Change parameters and inputs if needed, and press Reload to generate new whiskers.
j. If you are ready to save data, input a directory and file name into the inputs, and press Save.
k. If you desire to have one of the external forces to be applied, open the desired interface, fill in the inputs, and click the toggle to tell the simulation the forces should be applied.
l. Press Start Simulation to begin the simulation

m. Once the simulation is complete, the data will be stored into the csv file with the name you have given.
n. Copy and paste the data from the csv file into the whiskerOutputAnalysis.xlsx file included in the repository.
o. To reset the simulation, press Reset Gravity > Clear the Directory and File Name and press Save > Enter in new parameters and press Reload > Press Reset Sim
p. Alternatively, press the Play button again to fully stop the simulation, and press again to reopen.

Closing

This program has been designed with reliability and ease of use in mind. The goal of this user manual has been to provide a useful guide on how to use this program to its fullest potential. By following the instructions and tips provided, you'll be equipped to navigate every aspect of the program, regardless of experience with programs such as Unity or GitHub. It is highly encouraged to experiment with this tool and get the very most out of what it has to offer.

AUBURN UNIVERSITY
Samuel Ginn College of Engineering

3D Modeling of Detached Metal Whiskers
"How it Works" for .cs Scripts

Version 2.0

MECH 4250 – Auburn Group 3, Metal Whiskers Team 2
Shawn Eom, Spencer Hurst, Kurt Knudsen, Nathan Nichols, Carson Reams
Finalized 12/4/2024

Introduction

This section of the manual contains information regarding the 3D Modeling of Detached Metal Whiskers Simulation, developed by two teams of senior Mechanical Engineering students at Auburn University. This section is designed to be a concise explanation on how certain important and complex components within the simulation work. The CSV, Excel, "CircuitBoard" tag, UI tabs manager, Hierarchy organization, Rotating and Spinning the Board, Whisker Spawning, Importing Circuit Board, and Screenshots are explained in this manual. This is not meant to be a full synopsis of the entire program, but instead a brief guide to key elements of the Metal Whiskers simulation.

CSV:

There are 2 individual scripts that alter/write .csv data: 1) csvReader.cs 2) WhiskerControl.cs

csvReader.cs:

- Reads .csv file written in "WhiskerControl.cs" to display to a Text TMP game object within the Unity Canvas.
- Within this script you can edit:
    - "scriptname.cs" to target which .csv file to read within the Unity game folder

whiskerControl.cs:

- Overall script to dictate whisker spawning, whisker sizing, and overall whisker control. Also manages a list of bridged whisker and conductors.
- Lines XXX – XXX alter .csv files
    - Checks for availability of "BridgedWhiskerData" dataset (dataset created in XXXX.cs)
    - Creates a header for organization of data points into columns (e.g. "Bridged Whisker", "Conductor1", "Iteration #")
    - Reads out stored "BridgedWhiskerData" and organizes the data based on the headers into the .csv
    - Creates 2 .csv files:
        - 1) XXXXX.csv: Checks for "Directory" and "File Name" input and creates a .csv file as described above in the indicated directory
        - 2) XXXXX.csv: Creates a bridged output report csv within Unity's "Assets" folder for reading by csvReader.cs to display output within in-game GUI

UIScript.cs

- private void WriteDataToCSVFile
    - Writes simulation data to a .csv file.

        - Accepts file path and toggles between writing all data or just bridged whisker data.
    - public void WriteDataToCSV
        - Handles paths for saving both full data and bridged whisker data.
        - Calls WriteDataToCSVFile for each file.
    - public void SaveWhiskerData
        - Saves individual whisker data into the DataManager for later writing.
    - private List<KeyValuePair<string, string>> GetSimulationInputs
        - Collects all simulation input values from UI elements and formats them for the .csv.
    - Headers and Formatting
        - Dynamically adjusts headers and rows based on data types (e.g., all whiskers, bridged whiskers).
        - Writes headers first, then iterates through data lists to populate rows.

WhiskerData.cs
- Represents data for individual whiskers in the simulation.
- Separates data for all whiskers and bridged whiskers.

DataManager.cs
- Centralized management of whisker data across the simulation.
- Handles lists for different categories of whiskers.

Excel:

An excel file to be sent out with full/test builds of the Metal Whisker Simulation to perform analyses on the output file as created in "WhiskerControl.cs" (refer to **CSV** section)

- "Data" sheet
    - Area to paste in output data from generated .csv file the simulation creates
    - Each section is separated into different tables to make it easier for future calculations
- "General Analysis" sheet
    - All calculations should be completed automatically via excel formulas
    - Overall Probability is the % of iterations that have at least one bridge over all iterations that were ran in the simulation
    - Individual Probability is the percent of whiskers that bridged over the total # of spawned whiskers for each iteration
    - Summary statistics shows the avg, min, and max length, resistance, and width.
    - # of Iterations with 0 bridges and at least 1 bridge uses the "Helper Data" sheet to help calculate it.

- “Bridged Whisker Analysis” and “Criti Bridged Whisker Analysis” sheets
    - The tables are excel pivot tables which helps in plot generation
    - The Filter column uses excel slicers
    - The plot is a combination chart

“CircuitBoard” Tag:

A tag created to give the blank circuit card game object an assignment as “CircuitBoard”

Utilized to:

- Find and identify the blank circuit card of the PCB (finds the largest object with largest surface area)
- Create walls based on game object(s) tagged/identified as “CircuitBoard” and is outlining the tagged game object.
- At the beginning of the simulation, the game object with the “CircuitBoard” tag becomes a child of the BasePlane game object to allow the game object with the “CircuitBoard” tag to spin in Unity’s x, y, and z coordinate plan; due to the board rotating changes its coordinate axis and is not aligned with Unity’s constant x, y, and z coordinate axis. Parenting the BasePlane game object which will not be rotated but only spun, allows to rotate the CircuitBoard game object while not changing it’s coordinate axis.

UI tabs manager :

- The UI tabs manager, manages the tabs of the UI. It is done by adding many control conditions and acts like a lot of “if statements” so whenever a button is pressed the tabs teleport to the designated position relayed in the statements in the UI tabs manager.
- The UI has a root UI which is the un-activated buttons (non-clicked) and when a button is pressed the respective UI tab appears, the root UI drops down lower, and the new activated button is above the un-active button of the same UI.

  TabManager Lines: 1-4 drops down Root Tab
  5-8 respective tab appears
  9-12 respective tab disappears
  13-16 Puts root tab back
  17-19 Tab 1 offscreen when inactive pushed
  20-22 Tab 2 offscreen when inactive pushed
  23-25 Tab 3 offscreen when inactive pushed
  26-28 Tab 4 offscreen when inactive pushed
  29-33 Tab 3 had weird issue so hardcoding it away.

Hierarchy organization:

- The hierarchy is a bit confusing upon first look but after this section the user should be able to find what they need with relative ease.
- The hierarchy does change at the beginning of running the simulation so a few objects will be in different places compared to before and after starting the simulation. There will be a point of the similarities and differences for ease.
- Similarities of before and after starting the simulation:
    - The objects under "Canvas" do not change and are not affected by the change in UI tabs, such as the "WhiskerOrbitSelect" is still necessary no matter the state of the UI tabs.
    - The objects under "User Interface" does not change due to there is no need for parenting from the beginning of the simulation for it has already been done manually. When you look click a tab and see the buttons, input fields, dropdown, and text, they are all a child of the tab they are in, which makes them easy to edit, move, and easy to find in the hierarchy with increased organization.

- Differences of before and after starting the simulation:
    - The three main items that change at the beginning of the simulation are "WhiskerSpawnPoint", "BasePlane", and the game object that gets the "CircuitBoard" tag attached to it.
    - At the beginning of the simulation the "WhiskerSpawnPoint" becomes a child of the game object with the "CircuitBoard" tag, and then the game object with the "CircuitBoard" tag becomes a child of the "BasePlane" game object.
- These parenting at the beginning of the simulation is vital so that the functions that "BasePlane" and "WhiskerSpawnPoint" are respective to are parent/child to the game object with the "CircuitBoard" tag so it works properly.

Rotate and Spinning the Board:

- There are two main parts before and after clicking the "Start Simulation" button
- Before the user clicks the "Start Simulation" button, the user can adjust the rotation of the board giving it angle that is representative of the actual circuit board; the user can also apply a spin to the board, which will automatically happen to show the user the speed at which the board will spin.
- Once the user clicks the "Start Simulation" button the board resets to the initial rotation values (if the user did not input any rotation values it will reset to 0,0,0) and after resetting the spin value is then also applied unto the circuit board

Whisker Spawning:

- Main points: Whisker Spawn Point, Spawn Area Size, "WhiskerSpawnPoint" game object
- In the first tab "Whisker Properties" there are 2 columns with 3 input fields each, one being the Whisker Spawn Point which is the center point at which the whiskers spawn above, where the red sphere is the center of the area and the whiskers spawn above.
- The next column is the size of the Whisker Spawn Point Area and below the second column is a toggleable box which shows the area at which the whiskers will spawn.
- Then the final point is the "WhiskerSpawnPoint" game object which is the red sphere that is seen when the user moves the whisker spawn point. The way the whiskers are spawned and moved is that when the "Start simulation" button is clicked the whiskers spawn around (0,0,0) and the "WhiskerSpawnPoint" teleports to (0,0,0) as well, then the whiskers become a child of the "WhiskerSpawnPoint" and then the "WhiskerSpawnPoint" teleports back to the inputted place that the user inputted and the whiskers are teleported respectfully to the position of the "WhiskerSpawnPoint".
    - This was necessary to make the "WhiskerSpawnPoint" to work in this way due to the circuit board is centered around (0,0,0) and the whiskers also only spawn around (0,0,0) making it impossible to include desired placement of whiskers without having extra space of undesired area

Importing Circuit Board:

- You can't directly import directly from Altium to Unity without a 3rd party conversion
- To export .obj files you have to export .mtl files with it
- Altium does not export traces or pads, if they do they do it in 2D not 3D
- Blender is specifically used to clean up the geometric data, the whiskers would look like they are connected when they shouldn't, or not connected when they should.
- Some PCB are components have very complex geometry while the simulation only requires simple geometry
- Within Unity, you have to manually attach materials to board and objects, due to the .mtl file couldn't be read well, so you have to manually select it in unity to each component. No good way to automate.

Screenshot:

- There was no good and safe software that could record the simulation.