VR Lab

Final Report

Soumil Gupta

12 May, 2022

Introduction

In this report I will summarize the work and development that I have done this semester for the VR Lab. As part of the VR Lab, I have been working on the Polarization Electric Field lab. This lab has already been in development for a couple of years, and I have been working on the fine tuning of the lab to make it more playable and add more improved functionality. There are a couple of key elements that I have worked on for the lab. The first are the electric field arrows, optimizing the frame rate by changing the dielectrics, organizing and decluttering the Unity folder, and coding the capacitor versus time graph. There were more things too, such as a lot of reading and googling I had to do this semester to learn about git and unity programming. I also did some aesthetic changes, as part of future projects for the polarization electric field. A major part of being able to improve the lab was in understanding the lab. I looked over all the code and asked Andrew questions about the code to have a better understanding of what the thought process was for the previous developers of the lab when making the original code and unity scene.

Electric Field Arrows

Visual Changes:

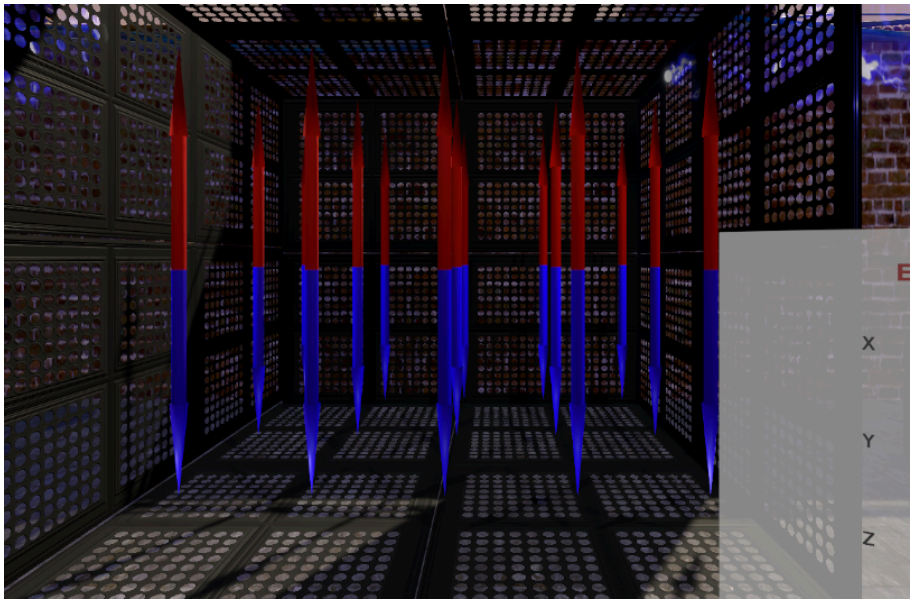

Figure 1: Initial Change to Arrows. Added tips on each end of the cylinder.

Figure 2: Created a new prefab via blendr that only points in the positive direction. Had to do many changes to code because with the new prefab comes more nuances in scaling and rotation to take care of.

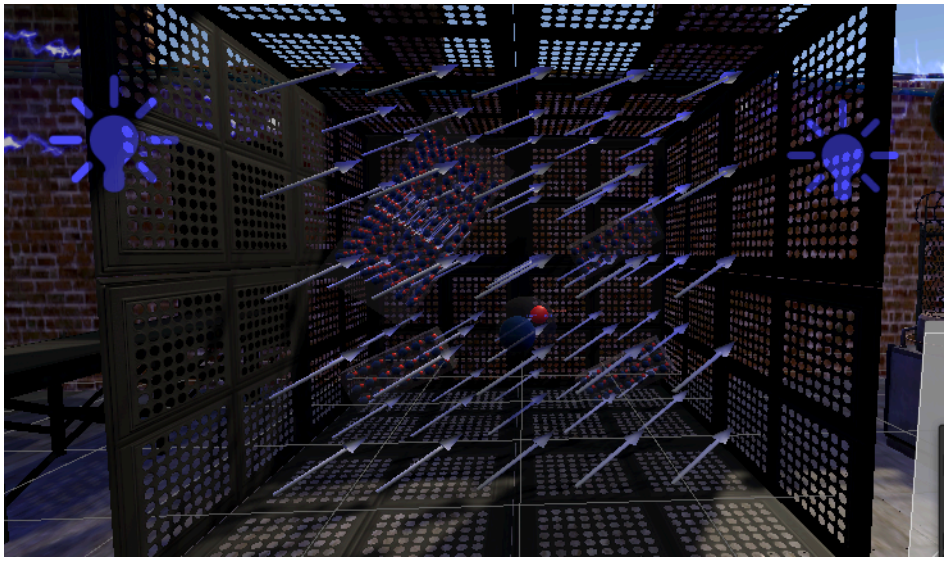

Figure 3: Added multiple layers of arrows to make it look like a vector field.

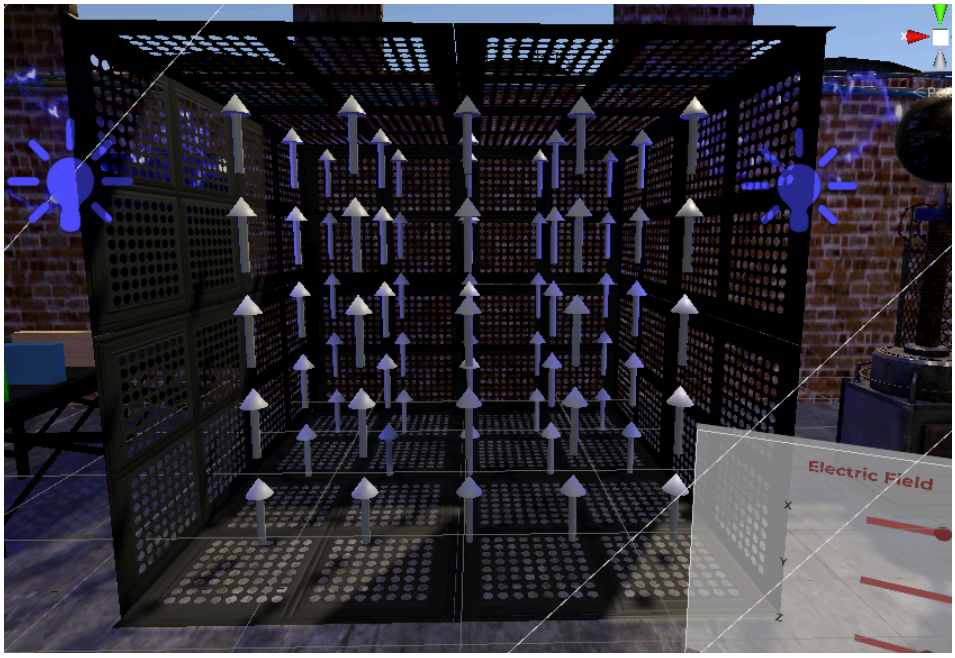

Figure 4: Optimized and Improved Arrow asset and code to avoid awkward looking transformations and less defined renderings of the arrow.

Coding and Design Process:

The electric field arrow had a lot of developments and improvements over the months as we further investigated what we wanted the electric field to look like and improved the

code and visuals accordingly. At the beginning of the semester, there were a lot of places in the code and in unity where transformations were occurring that caused the arrows to elongate to infinite length, be flat, or very skewed. Through much investigation into the functions of the code, I understood the different floating point constants, and the math involved on both parent scaling and local scaling of the arrows to ensure they do not strengthen an unnecessary amount, and so they render nicely.

Here is a full video of the current fully-functioning electric field arrows: https://drive.google.com/file/d/1Rt5NpnM4TkyihUOXk9bOmSHi9QQ3eWcK/view?usp=sharing

I also had to learn blender to make more arrow models with wider cross-sections that would look good when it goes through all the transformations and translations in the lab. I asked Nancy to make the latest rendition of the arrow which is the one we are using in the lab. The dimensions of the initial models was less important once we identified how to change the .fbx files into prefabs that remove some of the rendering issues on the unity rendering machine, and remove some of the odd scaling procedures. To break down how the scaling works for the arrows, we first have the transformation that you can set on unity in terms of scaling. Then, we have two scripts that work with each other to calculate new direction and then new scaling of the arrows. When the arrows were elongating to infinity, I changed some of the code logic to work with our one-sided arrow that required a lack of ambiguity in the scaling code, which was not present when the scripts were initially written to work with one cylinder. Then, I also removed the scaling factors and the layer gameobject and also set predetermined scaling parameters in converted prefab variants of the blender models. Converting blender objects to prefab variants was crucial, because it allowed uniform scaling and proper rendering, and because transformations wouldn't result in subparts of the object colliding with other parts of the same object causing odd renderings. This way, we now have a streamlined method of properly scaling arrows into the game. In fact, with the current code, we can even use other models of the arrow now.

Next Steps:

However, one more improvement we can make to the arrows is to make the arrow head and the tail operate separately, so the tail transforms in size to reflect changes in magnitude to the magnitude of the vectors, but we keep the arrow head from being transformed. I understand the theory of how to implement this, but, it will take away our

new ability to place any arrow prefab into the scene, because the solution is to add extra code for a separate asset for the arrow cone and for the arrow tail, each of which will have the same rotation code, the separate arrow cone prefab will require an additional un-scaling function. This is do-able, the question is just whether we want to pursue this. Please let me know if we want to have no scaling at all on the cone based on the latest version of the electric field in the video.

Dielectrics and Frame Rate Optimization

At the beginning of the semester, I was told that when the students last used the lab, it was unplayable, that people were getting anywhere from 0 to 10 frames per second only when doing the first part of the lab.

To investigate this, with the help of Joseph, I learned about the Unity Profiler tool, and investigated which one of the processes in generating the gameplay was causing the most lag spikes in the frame rate graph. It turns out that when you put a dielectric, especially the big dielectric with 400 atoms into the electric field area, a massive lag spike is generated which stays for the duration of the game.

When looking at the code, the problem was that each individual atom was constantly checking whether the object had collided with the electric field or not. When you have hundreds of objects checking every frame checking their status or position relative to another object, that lack of efficiency leads to massive frame drops. To handle this, I redesigned the collider system so there is one collider for each entire dielectric instead of the individual hundreds of atoms, and that one collider gives an interrupt instead of polling signal on update() to have all the atoms display themselves as polarized. I had to redo the code for the colliders to make sure that all the atoms at once polarize when even if the corner of the dielectric is in the electric field, and wrote a new script with Andrew to accomplish this goal. Via changing where different checks were happening and removing hundreds of unnecessary collider objects, we stopped the dielectrics from being able to display only a certain amount of atoms polarized as the dielectric enters the electric field, but allowed the game to become playable again.

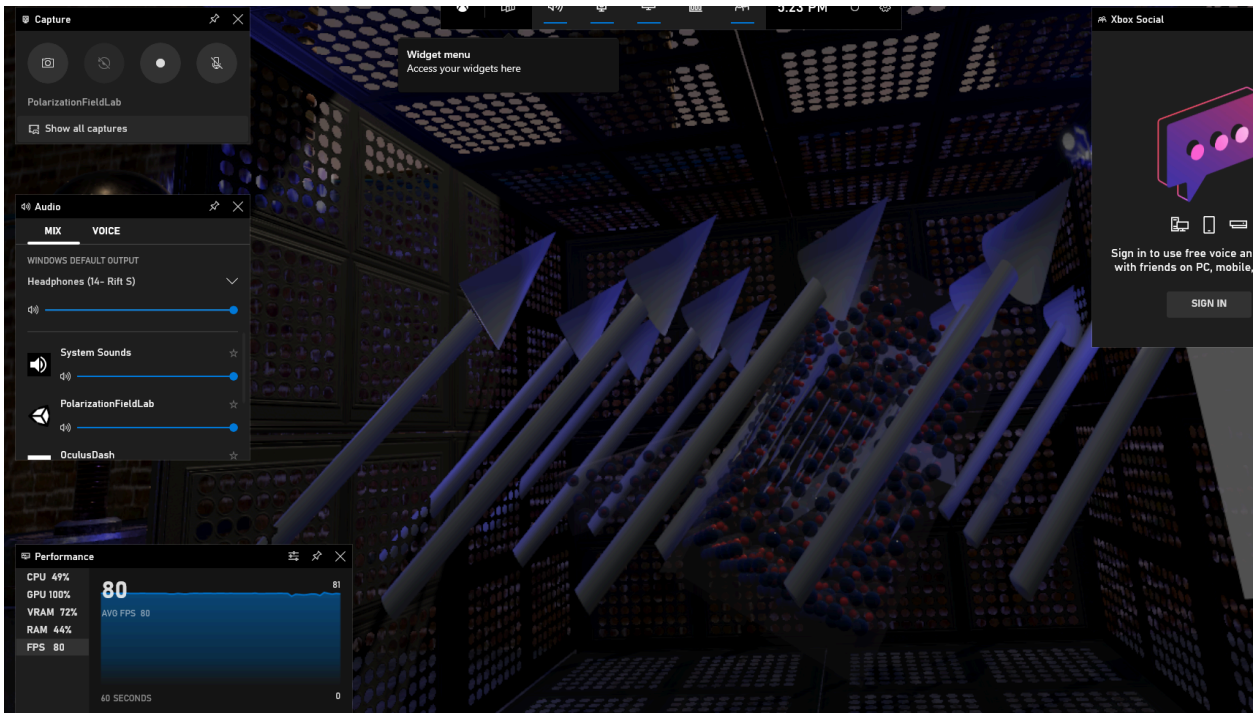

Figure 5: The game now gets at the bare minimum under maximum load 80 frames per second. Even with the latest version of the game, we get a solid 88 frames per second. It is my belief that the game can be run with more frames per second, but there must be some hardware limitation or software restriction on having an even better frame rate, unrelated to the efficiency of the project's code.

Capacitance Graph

Visual Changes:

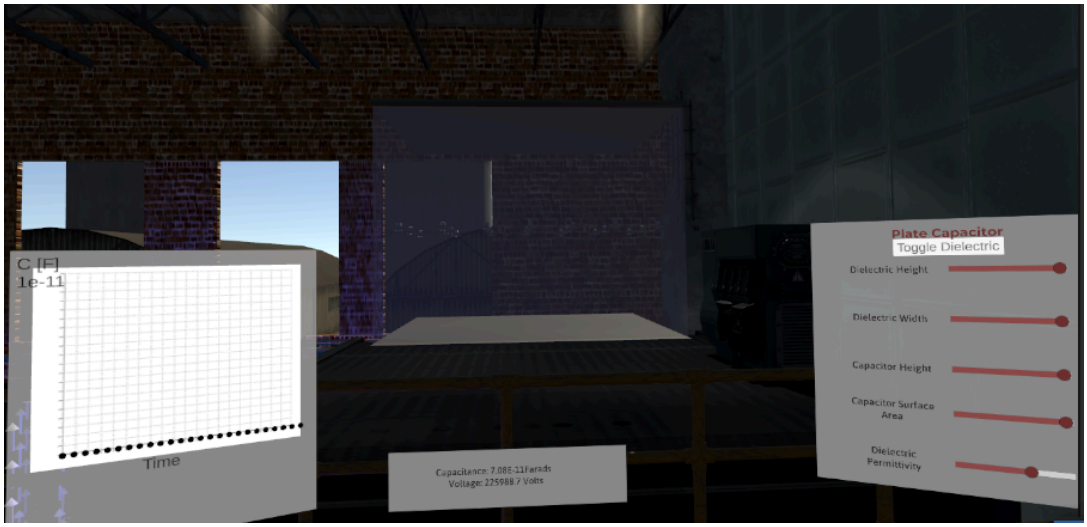

Figure 6: Current Lab Part 2 layout.

We want our player to be able to see a visual representation of how the changes they make to the parameters of the capacitor affect the capacitance, or measure of charge held by the capacitor. Currently, we have situated a graph panel in unity and created a script meant to generate lists of gameobjects, rectangle and circle sprites, that will change position as new capacitance values are generated.

This script is still a work in progress. I can now have the circles be generated, but their position does not yet change with the capacitance, and I need to have the rectangles be generated properly.

Decluttering and Organization

One crucial part of working on a project is being able to access all the relevant scripts and assets whenever you need them, and not having to dig through or be confused by tens of unnecessary folders holding gigabytes of completely irrelevant scripts, scenes, prefabs, materials, etc. With the lab, at one point the repository held over 32 gigabytes of files, so many of which were not being used. By the middle of the semester, we had the

repository down to 12 gigabytes, but that is still a lot to download on new machines, and a lot of unnecessary files. One big problem is pulling for the first time from the repository and opening up scenes, there were hours long loading times for all different prefabs, hundreds of which weren't even needed for the lab. So, I used the asset cleaner to assist me as I went through many folders, double and triple checked files relevance and and usage, and deleted all the unnecessary files. One key note for anybody using the asset cleaner tool, is that the cache has to be reset every time you use it to delete files and its understanding of the directory is not based on the folders you press. In the end, I got the project down to less than 5 gigabytes, and restructured the files so the relevant pieces were higher in the file hierarchy.

Scene Edits and Aesthetics

Conveyor Belt:

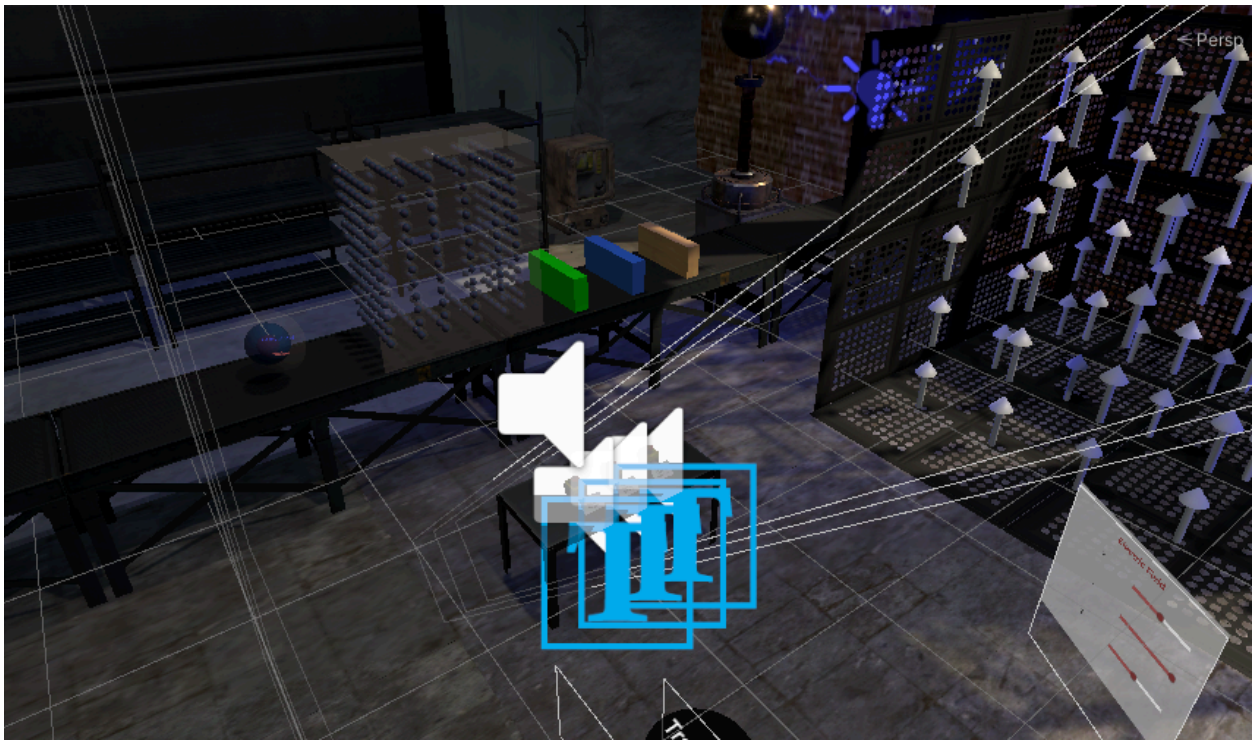

Figure 7: The conveyor belt is closer and therefore feels more reachable and part of the experience for the player.

One improvement we wanted to make was optimizing the conveyor belt and making it closer to the player. To move the conveyor belt closer, each item on the conveyor belt has hard-coded values for speed and position to move in, so it looks like it is moving because of the static conveyor belt. I had to change these values and also maneuvered other items around to make the scene still not look odd, and also so the dielectrics actually look like they are resting on the conveyor belt.

We were thinking of revamping the conveyor belt to make it not hard-coded so it is applicable to other labs. However, it was not a priority although Andrew did do some work on potential scripts and algorithms for a conveyor belt that works with any lab.

Creating a place to add gameplay information, that fits naturally in scene:

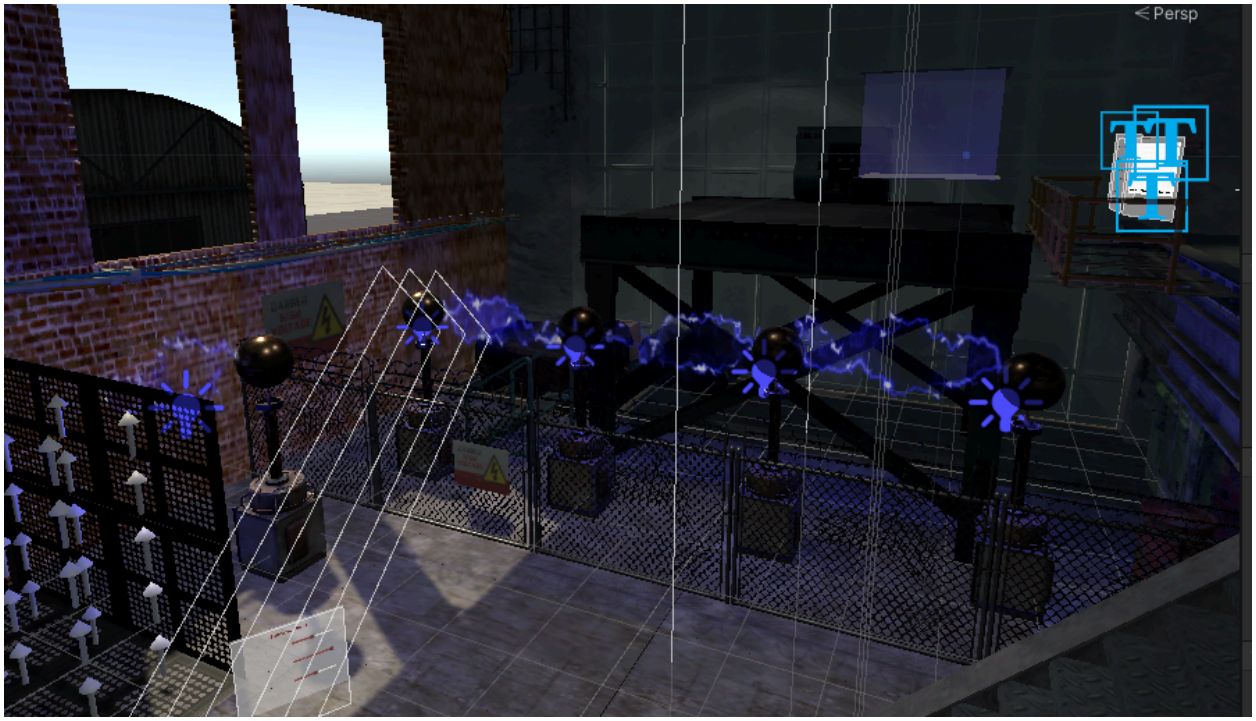

Figure 8: Added fencing and elements to fill up space in lab

The factory setting looked very open and empty to the right of the electric field. I felt that if we wanted to add more instructions that people could refer to on what to do, we could add a bulletin board posted to the right that the player could turn to see. Additionally, the space was quite empty so I added more objects to fill up that space, and add to my

interpreted aesthetic of the scene, which was an industrial age factory that was lacking much safety or cleanliness.

More work needs to be done in terms of aesthetics. I believe the main one is to make all the floating signs and panels that we utilize to interact with the two parts of the lab fit in with the environment. A cold-war era control panel that integrates the graphs and switches would be good, and a review of some assets which are only there to fill up space, and do not even fit logically into the aesthetic of the factory.

Of course, visuals are the least priority in the development of the lab, but when I was advised too by the grad students or had some more extra time, I took the liberty of improving the visuals of the lab.

Conclusion

I enjoyed working with the VR Lab this summer and I would love to continue working in the fall. I cannot work this spring due to multiple commitments I already have going on this summer, but I look forward to contributing to the polarization lab and other labs in the fall. I feel that I learned a lot about unity and programming and about the VR lab this semester. More time developing already-existing labs would be great to hone my skills to potentially one day start a new lab. I am willing to work on the polarization electric field lab more in the fall to bring it to its best state for the ECE 329 class, although I wish to move onto a new lab as well this fall. Also thanks to Andrew, Hsinju, and Joseph for giving advice to help me accomplish the tasks I mentioned in this lab.

To reiterate, these are the next main top priority steps functionality-wise for the future of the Polarization Lab: Capacitance graph and adding a new control scheme. I can also have the arrow cone not scale if need be but the dismorphing of the arrow is gone.