Nvidia Mosaic and UniCAVE on the CEWIT Silo

Alex Cuba 111560392

Abstract

In the Center of Excellence in Wireless and Information Technology (CEWIT) building on the Stony Brook campus, there is a stationary virtual reality system known as the Silo, a cylindrical setup of dozens of screens meant to fully immerse a user into its projected environment. However, for such a system to work properly without performance hiccups, one must optimize the system as much as possible. This paper will go over my method and process for defining these optimizations on the system using a combination of the Nvidia Mosaic command line application, as well as the UniCAVE plugin for Unity. By combining these two technologies, I was able to create a Unity prefab that accurately projects built in Unity assets across a thirty-two display node with little performance drops. In the future, this work can eventually be expanded to the whole of the Silo, as well as other stationary virtual reality projects.

1. Introduction

The CEWIT Silo is an impressive piece of technology, with over half a billion pixels across its entire display. For such a massive display, each node of thirty-two screens are each controlled by a system containing eight GPUs. This means that each GPU controls four screens. For such a massive system to remain in sync across various nodes, a lot of energy and processing power is required, and as such it would be wise to attempt to optimize your immersive application as much as possible. This goal can be achieved through the use of Nvidia Mosaic, as well as UniCAVE. Each of these systems allow for various performance saves and when combined can have massive effects on the system as a whole.

Firstly, Nvidia Mosaic is a command line application[1] which allows for users to combine multiple screens into one conjoined display. While this initially might sound similar to the basic display settings in Windows for stretching a display across multiple screens, Nvidia Mosaic combines these screens on a computing level. This means that if you combine the screens using Nvidia Mosaic, windows will only see the input as one unified desktop, rather than several smaller screens. This helps avoid the performance degradation one would acquire when trying to stretch across multiple displays on the operating system level.

On the other hand, UniCAVE is a Unity plugin which allows a user to reconstruct their multiscreen setup within Unity itself, using a combination of different scripts provided with the software. This allows Unity to render each display separately, and provides a performance boost over just running one fullscreen Unity application across the entire system. More details into both these technologies can be found later within this paper, as well as in the attached documents which give instructions on how to run each system.

This paper is split into multiple sections in order to provide the clearest understanding of my process on this project. Section Two will go over the methodology behind each of these technologies, including what work I did and how I was able to reach the final working state that they are currently in. The third section will go over the results of this work, the fourth section will go over future work that can be done with the technology, and the fifth section will conclude this paper.

2. Methodology

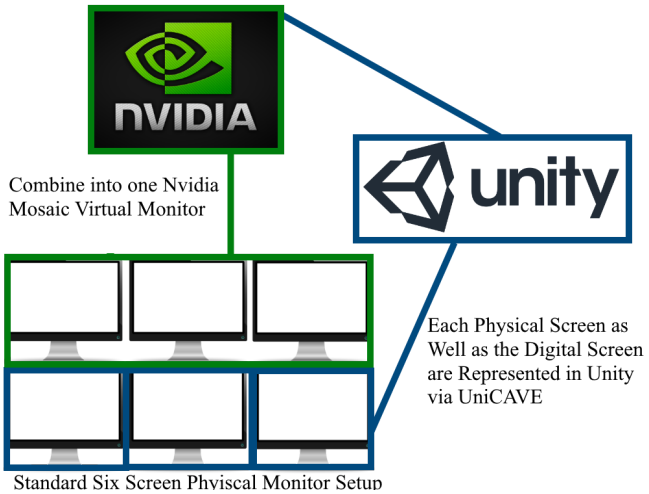

Figure One: A visual representation of how the physical screens, Nvidia Mosaic, and UniCAVE combine into one cohesive system.

This section will go over the process I went through in order to create a working system for both the six screen setup in the New Computer Science (NCS) virtual reality lab as well as the thirty-two screen node of the Silo setup in CEWIT. This section will be detailed in the chronological order of my developments, starting with the Nvidia Mosaic and then moving on to the UniCAVE.

2.1 Nvidia Mosaic

Before working on the UniCAVE portion of this project, I started with the Nvidia Mosaic application. It was done this way because UniCAVE requires the specific monitor layout of your setup to be hardcoded into the scene, so redoing the

[1] It is also possible to use Nvidia Mosaic without command lines through the use of the Nvidia Control Panel, but that was not used for this project.

monitor layout with Nvidia Mosaic afterwards would require me to completely redo the UniCAVE setup. The process of creating working Nvidia Mosaics for both setups took roughly a semester, with UniCAVE work taking up the bulk of the second semester.

The process of getting Nvidia Mosaic into a working state took longer than I initially expected. I believe this delay is due to my lack of experience with command line based programs, as well as the lack of documentation for the Nvidia Mosaic program available online. For about the first four to eight eight weeks of me working on this project, I was reviewing the previous semesters' command line scripts that were provided to me and trying to understand and reverse engineer them in order to update them for the newer systems.

Eventually, I was able to locate some documentation provided by Nvidia on their download page for the Nvidia Mosaic executable. It was a short .txt file that provided a rough overview of the parameters available to the user. This documentation is linked in the batch file instructions text document provided with this paper. This documentation provided details on how to properly construct the layout you desired, as well as provided other information on various parameters, such as the resolution and bezel corrections. After this, I was able to properly construct a working Mosaic batch file for the NCS tiled setup after only a week or two, much faster than I had originally anticipated. An example of this script is provided in Figure Two.

I wrote two batch files for the tiled setup in NCS. One combined all six screens into one monitor, while another split the six screens into four monitors: one across the top, and three on the bottom. This script is the one pictured in Figure Two.

```
set mode_arg=set
set res_arg=2560,1440,59.951
.\configureMosaic.exe %mode_arg% ^
                      rows=1 cols=3 ^
                      out=0,0 out=0,1 out=0,2 ^
                      res=%res_arg% ^
                      nextgrid ^
                      rows=1 cols=1 ^
                      out=1,0 ^
                      res=%res_arg% ^
                      nextgrid ^
                      rows=1 cols=1 ^
                      out=1,1 ^
                      res=%res_arg% ^
                      nextgrid ^
                      rows=1 cols=1 ^
                      out=1,2 ^
                      res=%res_arg% ^
```

Figure Two: An example of an Nvidia Mosaic batch file.

After these attempts were successful, I moved over to the Silo where I immediately encountered several problems:

1) **Bezel Correction does not work across all thirty-two screens.** When initially testing on only sixteen screens, I was able to get functional bezel correction. However, once I expanded to the thirty-two screens, the bezel correction no longer worked, regardless of what variables I would assign to it. I suspect this is some issue with the Nvidia executable itself, however I was unable to find any information on this.
2) **A full thirty-two screen setup breaks the node.** Once again, I was able to get a sixteen-screen setup into one working monitor, however Nvidia Mosaic does not work properly when trying to turn a full thirty-two screen node into one monitor. In fact, it winds up completely breaking and results in having to restore a previous version of windows. I presume that whatever restrictions are causing this issue is the same issue resulting in the bezel not working. I was not able to find a resolution for this either.
3) **Nvidia Mosaic topology does not allow multiple GPUs across multiple monitors.** Basically, this means that Nvidia Mosaic either wants all GPUs to be part of the same monitor, or for each GPU to have its own unique monitor. The executable does not allow for multiple GPUs to be part of the same monitor if another monitor already exists, as it throws a "NO_VALID_GPU_TOPOLOGY" error. This prevents us from creating setups such as splitting the screen into half (four GPUs per monitor) or quarters (two GPUs per monitor).

As a result of these issues, there is currently only one working Nvidia Mosaic batch file for the Silo node, which divides the screen into eights (one GPU per monitor). It can be presumed that one can replicate an "all screens in one monitor" setup by disabling the Nvidia Mosaic and using the built in Windows "stretch across all displays" option, though this would probably have some degradation in performance.

2.2 UniCAVE

In comparison to the Nvidia Mosaic, getting a properly working UniCAVE setup went significantly faster than the Nvidia Mosaic setup, mainly because of my familiarity with Unity, as well as there being more easily obtainable documentation. While the process to get UniCAVE working was quite simple, there are still several problems that I had encountered throughout the development process, which will now be discussed.

The first issue I encountered when working on the NCS version of my UniCAVE project was that the built executable would crash a few seconds after start up, which I quickly resolved to be an issue with how I set up the network management scripts in my project, so overall this

issue was quite minor. The second issue I had was a larger one, as I could not seem to get the screens to display the way they were supposed to, despite the way they were positioned in the project was correct. After some research further into the documentation, I found this error was because I did not properly initialize the offset values for each screen in the scene. After doing this, the screens loaded properly, and I felt confident enough that I would be able to move my project to work on the Silo.

However, the Silo had its own set of problems that differed from the ones I had encountered on the NCS six screen setup. The first major hurdle I had to overcome was that despite the project properly loading into full screen on the NCS setup, it would not do the same on the Silo. Instead, it would only full screen on one of the Nvidia Mosaic monitors, meaning it was occupying a space that was one eighth of the correct size. I suspect that this difference was because of the larger amount of screens in the Silo node, and it resulted in some compatibility issues with the full screening capabilities of UniCAVE. To resolve this issue, I instead adjusted the project settings to run the executable in a window whose size is the full resolution of the node. In order to make sure the window ran in borderless mode, I created a batch file. The result of this fix gives the appearance of the application running on full screen, with no visible sign of the desktop below. This can be seen in Figure Three. The final issue I came across is the stereoscopic visuals on both the Silo and NCS tilted setup. On the Silo, my project in stereo would just be duplicated across the top and bottom, as seen in Figure Four. I initially tried to resolve this issue with the help of PhD student Saeed Boor Boor, but we concluded that there were some issues with the hardware (whether that be the cables or the monitors) that was preventing the system from correctly booting into stereo. We also attempted this on the NCS tiled setup, only to get the same result. In the end, this issue remains unresolved.

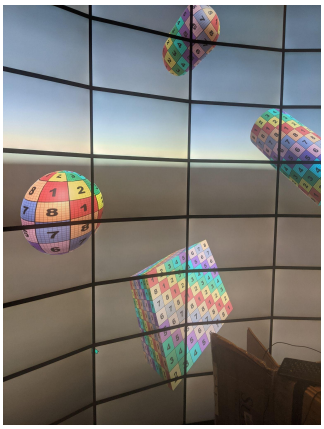

Figure Three (Left): The Silo UniCAVE project without stereo turned on. The image is quite smooth and blends near seamlessly between the screens.
Figure Four (Right): The incorrect display of the Silo UniCAVE project once the stereo is attempting to activate. Notice that the image is replicated between the top half and bottom half of the node.

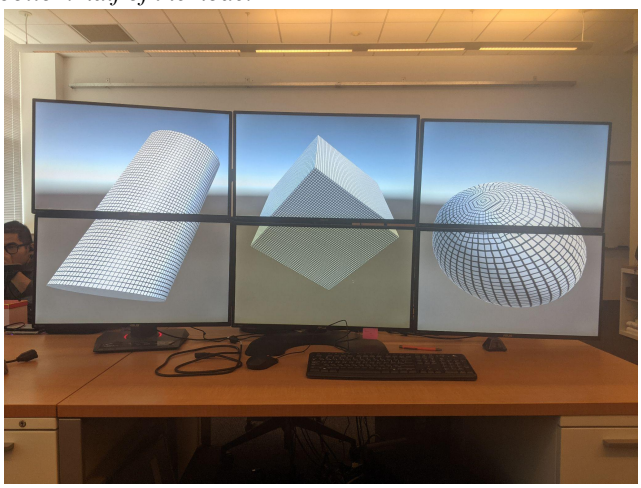

Figure Five: The working UniCAVE prefab on the NCS six screen titled setup, without the stereo activated. Note that the shapes do not line up correctly due to the lack of bezel correction, as well as the physical screens being out of line with each other.

3. Results

After two semesters of work, the resulting products are a total of three batch files for various Nvidia Mosaic setups across the two systems (two are for the NCS tiled setup, while one is for the Silo node), as well as a working UniCAVE prefab for each. Once the stereo issues on each system are resolved, these prefabs are very likely to also present a stereo image as well, even though minor adjustments might need to be made. These projects also come with their own detailed instructions on how to set up and use them, in the scenarios where one who is not familiar with the systems will have to use them in the future. All these supplemental materials are included in the Google Drive folder where this paper is originally located.

4. Future Work

There are several areas in which future work can be done on this project. The most obvious area is introducing stereo to both systems. As mentioned previously, I believe that this will be quite an easy goal to achieve once each system has its respective hardware and software issues resolved. Simply turning on the stereo in the Unity project settings should be all that is needed. Another area of future work that I believe would be easy to implement is testing the UniCAVE projects with the external OBJ files or prefabs, as both scenes I constructed use the default shapes provided with Unity. Compatibility issues seem like they would be unlikely, but it is worth testing as it can help produce a more detailed scene for a user to experience.

Besides additional detailing, future work can be done to make the scene more interactive with a user as well. Things such as animations or giving the user the ability to move around the scene can greatly improve the immersiveness of

the setup, as the scene itself would feel more alive and the user can explore at their own pace.

Once the other nodes of the Silo are up and running, future work will need to be done in order to extend both my UniCAVE project and Nvidia Mosaic batch files to work across multiple nodes. Documentation on how to do this will be included with the downloads of my work.

5. Conclusion

To conclude, my main contribution with this project was a highly modifiable Nvidia Mosaic and UniCAVE system that can work on either the six screen NCS lab setup or, more importantly, the thirty-two screen Silo node in the CEWIT building. Included with these projects are in-depth documentation detailing the systems, how they work, and what one can do in the case where they need to modify any parameters. Overall, I am very happy with the end result and I am glad I was given the opportunity to work on this project as it helped me learn many things such as command line scripts, as well as expand my knowledge of various programs such as Unity.