

We decided that Tight VNC would be the distribution of choice for the display nodes. We discovered different pieces of software developed and utilized by NCSA at the University of Illinois at Urbana-Champaign called Display Wall-in-a-Box, or DWiB. They had several materials available for display wall development, and a version of VNC that, while outdated, contained a “region” argument that would allow us to access only certain areas of big display. Our goal became to analyze the implementation of region in Tile Viewer, their version of VNC Viewer, and modify Tight VNC as we saw fit to accommodate for region access.

First, proper packages were installed to get Tile Viewer working on one of the display nodes and the code was compiled from source. The argument took four options, two for the x- and y-coordinates and two for width and height, and displayed a region of the larger display starting at the given point (x, y) at the given width and height. Upon further inspection of Tile Viewer's code, it was clear that it definitely used to be developed alongside Tight VNC Viewer. Therefore, by finding where edits were made in Tile Viewer from the original code, it would be easier to find where adjustments would need to be made in the current version of Tight VNC. One difficulty to overcome, though, was the newer version of Tight VNC utilized X's widgets, which is a more complex development of VNC not used by Tile Viewer. By comparing the older files with the most recent version, the differences were found and work began on editing Tight VNC.

Changes were initially looking at which files were still maintained and which had name or purpose adjustments for the newer version. We determined that the RFB protocol, main, and arguments files would be the main focus and location of the edits to be made. Also, we declared four global variables to represent the changes made in the offset and dimensions of the desired region. We copied over a section of code from Tile Viewer that checked if the incoming arguments were in bounds and checked for the zero case for width and height, correcting them to be equivalent of the x- or y-offset subtracted from the frame buffer width or height of the entire server display size. We then set the given width and height equal to the frame buffer width and height. Those were the only changes needed to resize the screen to the dimensions needed.

Before correctly implementing the offset, -region was added to the list of arguments. The format chosen was 'WxH+x+y'; for example, if the user wanted to see just the area on the second row, third monitor in, they would type in “-region 1280x1024+2560+1024” and the screen would be displayed. This was implemented by parsing the string using sscanf to obtain the values needed. The next task was to correctly implement the offset. We continued matching up code from Tile Viewer with the newer

VNC to find changes, but we came across the issue where once we had the offset working, the screen would leave a trail of white boxes as the mouse was moved. This issue was amended by looking at the screen in “view only” mode. While this would be the only implementation we would potentially need, we wanted to fix this problem in preparation of sending this feature to the developers. A catch in the RFB protocol for redrawing the rectangles appeared to fix the solution, but currently the screen has an abysmal frame rate for larger sizes, an issue which may have been caused by the fix or by another problem.

Next week we will be finishing the implementation of -region, as well as beginning to look more into communication between the server and the display nodes. Hopefully we will have some solutions to how to approach multithreading and faster communication between them.