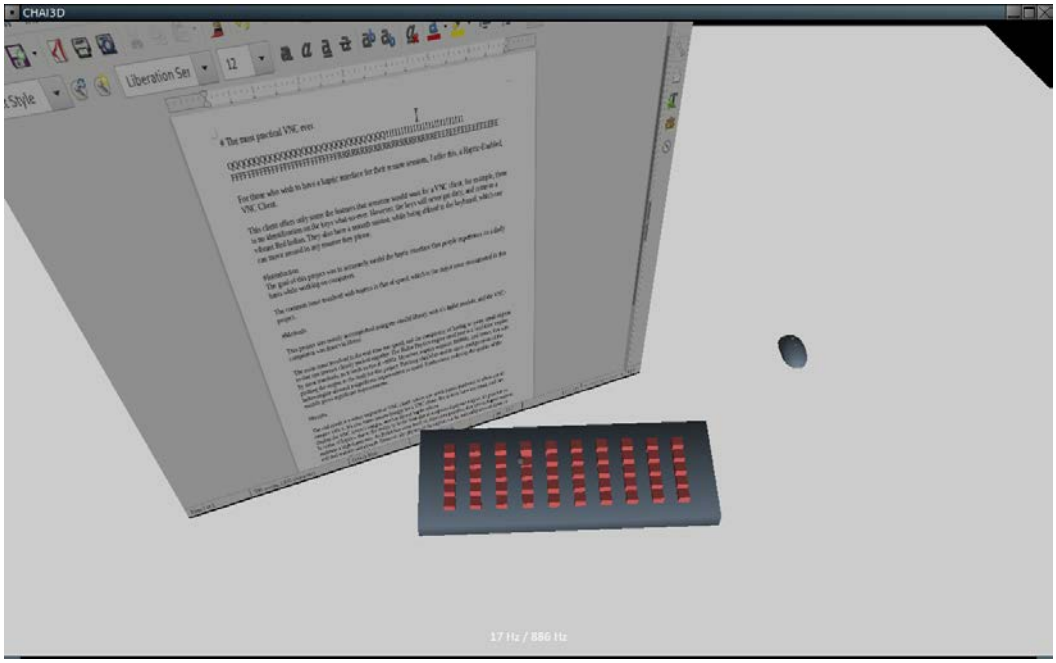
Introduction

The goal of this project is to simulate the good old touch and feel of a computer, but through a haptic device. To accomplish this, it is a VNC client. Thus, any mundane regular task that could be completed in a few minutes, can now span a few hours of trying to get your input in over the lag of VNC, and a not-so-great keyboard... and a mouse that's just for looks.



As with most things in Haptics, the speed of the simulation is, in large part, the hardest part. For this particular case with 50 or 75 keys (which is reduced from a normal keyboard), this is a rather taxing set of collisions to perform a thousand times a second. Especially so if one wants more complex interactions, which were simply forgiven due to already taxing the underlying system.

Methods

The project was mainly accomplished using Chai3d with it's bullet module, and the VNC-lifting done via libvnc. These all doing most of the heavy lifting, the remaining part was to 'simply' assemble it, and watch it barley crawl and the physics to glitch and break.

The majority of the issues encountered were speed related. Through trial-and-error it's possible to find a stable-enough configuration for bullet to run as fast as it can. However, this is still significantly slower than the haptic rate with the 50+ near-proximity object of a keyboard (~30-100Hz). Reducing most models from meshes to simple cubes improved the rate, however loses the fidelity. As chai3d doesn't currently support having 'ghost' collisions while different models are used to render, it is just left as the simple meshes.

Another significant improvement to the project was patching chai3d to enable bit masks for collision detections. This allows the closely packed keys to ignore each other, and the keyboard, for alternative simpler constrains to be used instead, and partially re-implemented for smoother haptic interaction.

The springs used for each key are coded in such a manner, leveraging a nearly-5-axis restriction from the bullet library, and leaving the last to be handled by specialized code.

This however requires the Bullet Library to update more frequently than it can actually handle, but as it is a separate system, with a level of interpolation between physics steps, it is possible to slow this down to maintain a reasonable haptic rate, with the expense of everything becoming slow-motion. To attempt to maximize this, and handle initial system instability, the system dynamically scales the time step with a threshold to stay relatively constant, with whatever the current system can handle.

Results

The end result is a rather impractical VNC client, but a functional one. The haptic feedback is consistently smooth and stable, and one is able to type on the remote system for whatever purposes they want. Including losing horribly at video games.