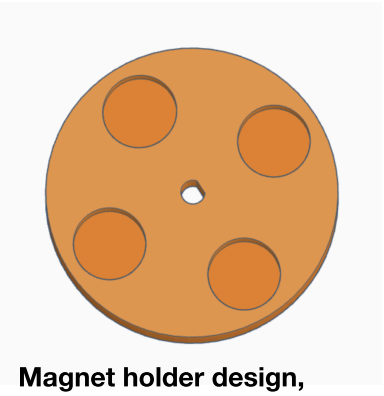
Technical details on Tension

Stepper motor

At the core of *Tension*, the steel cable is excitated by a four-pole alternating field, created by a 3D-printed disc, which contains magnets on each 90° angle. To turn, a NEMA-17 stepper motor is driven by a DRV8825. The DRV8825 is a bipolar stepper motor driver with adjustable current limiting up to 45 V. It has an important feature which is the STEP-pin, which can be pulled either high or low. By pulling the pin *high*, my stepper motor takes a 0.9° step for a total of 400 steps per revolution. So for it to spin, I need to pull it *'high'* and *'low'* quickly to make the disc with the magnets rotate.



Magnet holder design, *Tinkercad*

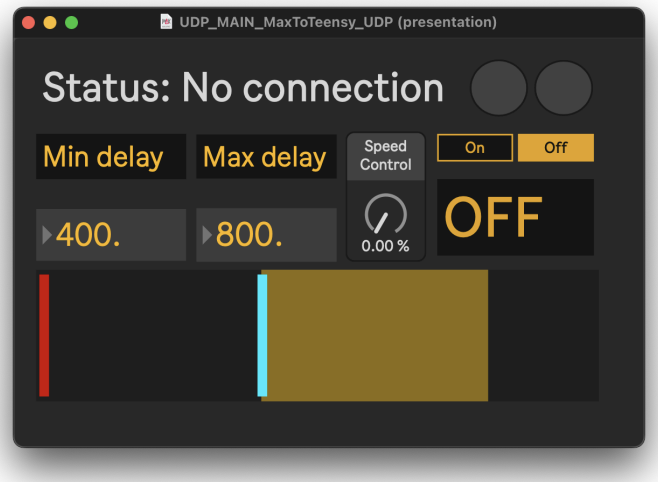
Controlling the motor

For controlling the stepper driver & motor, I've chosen to use a Teensy 4.1, mainly because of its fast upload speed and ethernet capabilities. Teensy 4.1 would be my way to communicate with the engine, first using Midi, later using serial data and eventually I would shift to controlling the Teensy 4.1 with OSC. The reason for using OSC instead of serial or midi is its ability to send data without package loss over long distances. This is rather important, given that the motor would be out of my sight and up in the air.

Communicating with Teensy

Given that the teensy would be out of my sight, I needed a way to get communication going with the teensy. Step one would be an interface using Max 9. After this, I created a 'handshake'. The steps were as follows:

1. Teensy sends 'Ping'
2. Max responds 'Pong'
3. Teensy sends 'Confirmed'



Engine interface, *Max MSP*

This process is repeated every 100ms, to be sure that the connection is live. Apart from the handshake, another important feature I added was the 'actual' motor speed, which was sent back to Max over OSC from the Teensy. The red line indicates the motor speed, the blue line indicates the 'new' motor speed and the yellow part indicates the range of control by my Behringer BCF2000.

Control

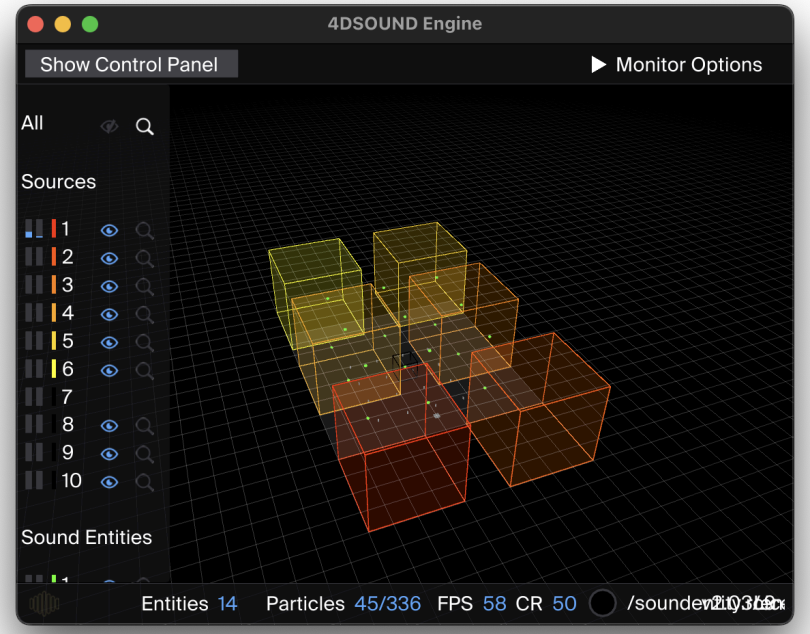
The electronics of the Tension were controlled by me, using my BCF2000 with a custom performance setup I had prepared in Max. The control was focussed on audio, so no control over the spatialisation.

Spatialisation

For the spatialisation, I chose to use the 4D-sound software. The audio was split up into 6 channels paired per location, with each their distinctive sounds. Closer to the organ I used low frequencies and an amplitude LFO, and further away in the back of the church I used modal resonators to match the scale of F minor, where the organ was improvising. The sound of the steel cable itself was already very impressive and low in the spectrum, so I chose to be hesitant in using too much modulation and signal processing and focus more on techniques to play the steel cable.

Steel cable & organ

The steel cable was played by Victor, the instrumentalist. He used different types of objects to play the steel cable. He primarily used a bowstring, bowing the steel cable which created a very strong sound. Next to the bowstring, used pliers that he slid over the steel cable. With this technique he was able to create a very different sound, in the sense that could shorten the cable with the pliers, creating a higher pitch. The organist, Micheal Bennet was closely involved in the creation of the piece and had suggested that he could improvise with the organ, leading the piece, reaction to the sound of the steel cable.



Spatial setup with source control, *4DSOUND engine*