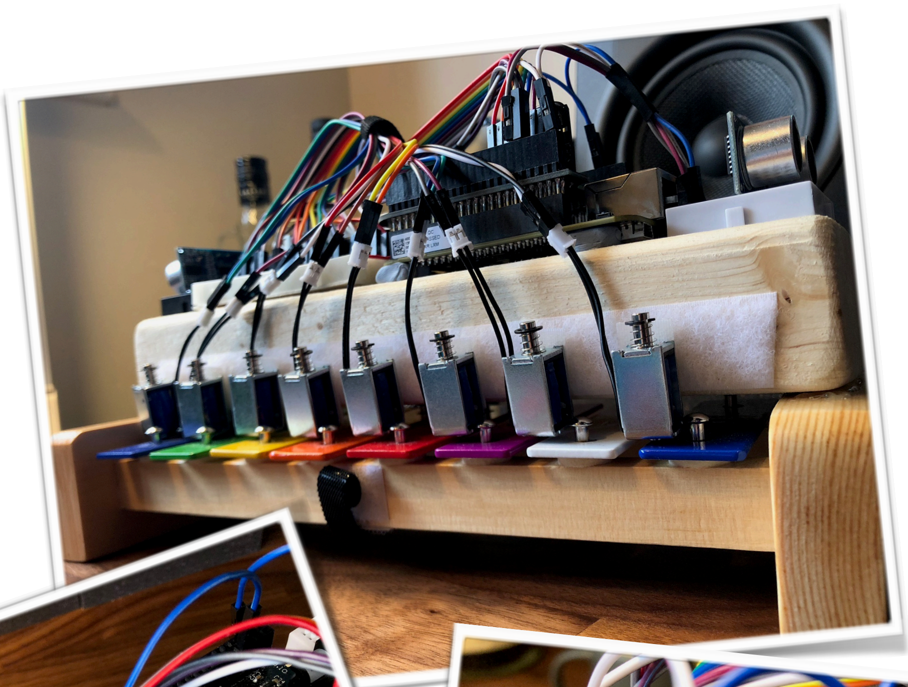
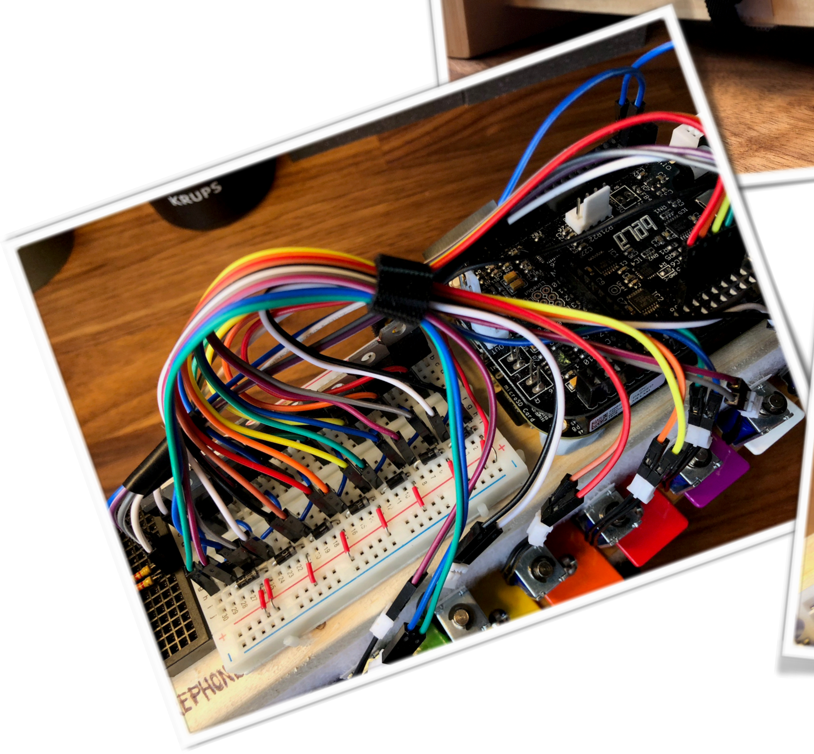
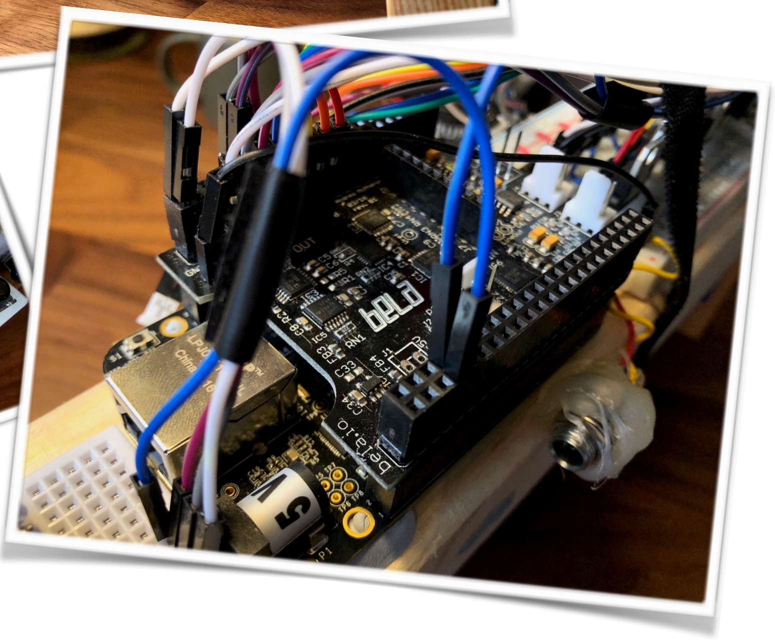
Candidate: 148508
Autumn Term 2017

Therephone

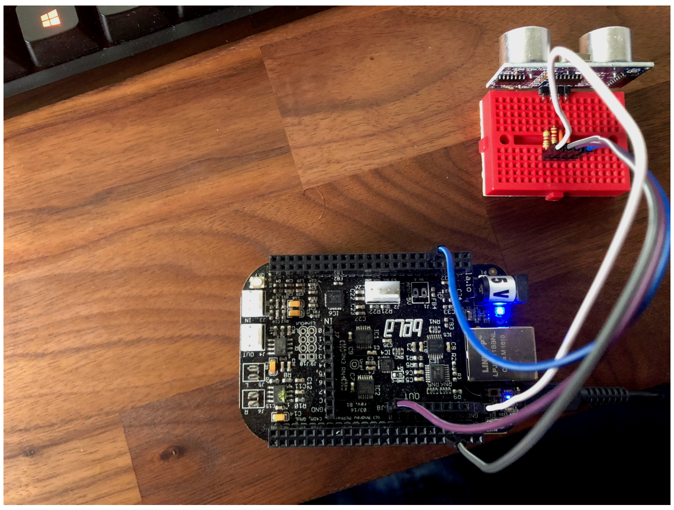
Project Journal





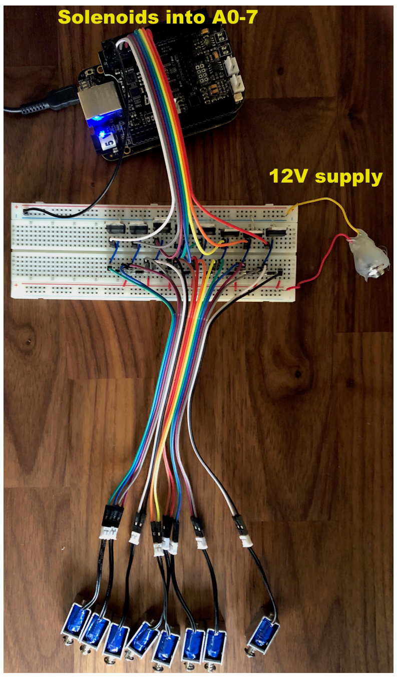


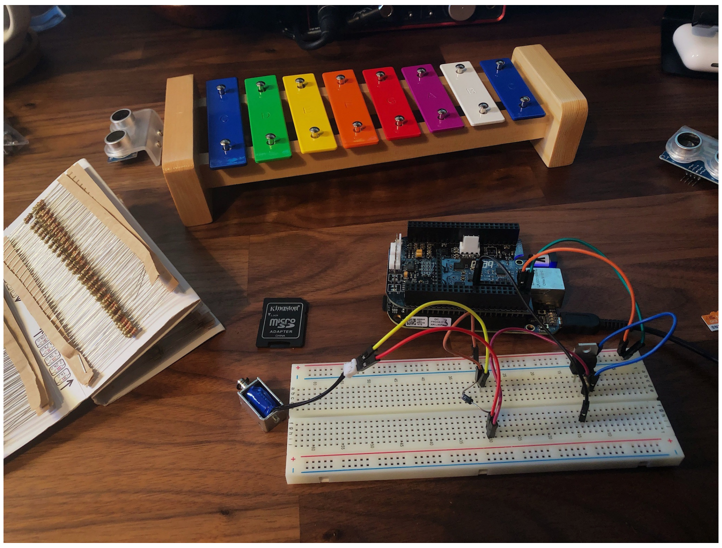
Development Journal



27th November 2017

Development for this project has started with connecting both sensors and actuators, making sure that they work, that I have the right connections and circuits.



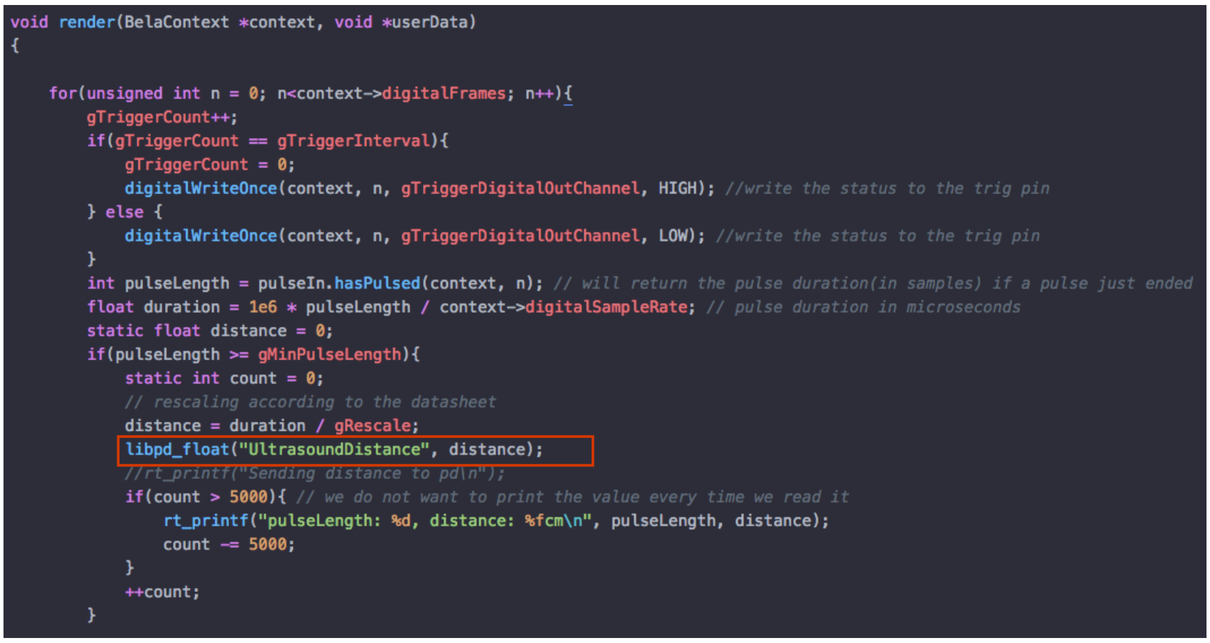


The ultrasound sensors normally require the analog outputs on the Bela, however with some tweaking and help from Giulio Moro (Bela Forums, PhD researcher at Queen Mary U, London), I have managed to get them connected to the digital outputs as I need the analog outputs for the 8 solenoids. The solenoids followed the same basic circuit as in the seminars, but multiplied by 8.

At this point, the solenoids were working with a basic Pd patch. However I was struggling with the sensors.

29th November 2017

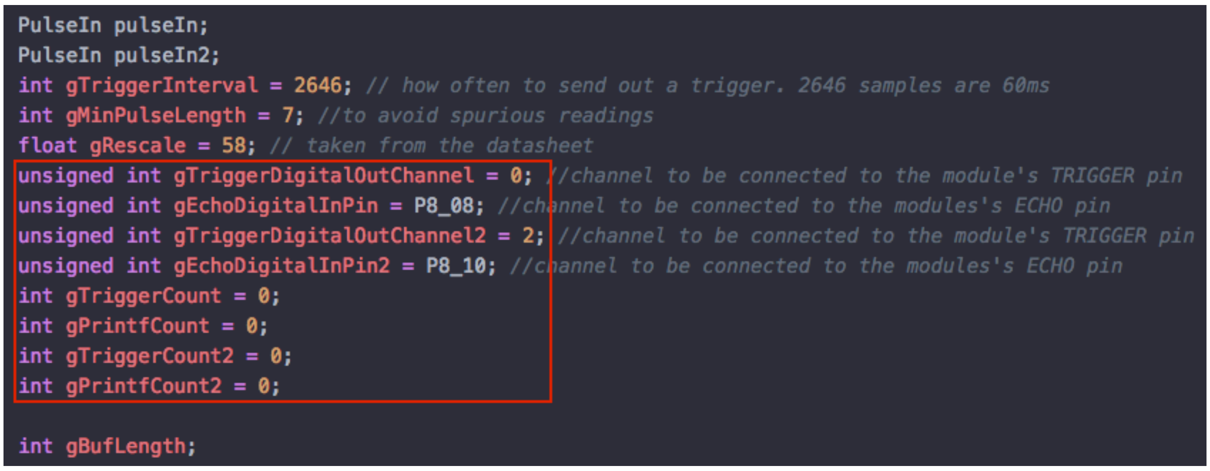
After learning how to tweak the C++ in the example ultrasound sensor render file, I managed to get input from the sensors into the Bela IDE.



```
void render(BelaContext *context, void *userData)
{

    for(unsigned int n = 0; n<context->digitalFrames; n++){
        gTriggerCount++;
        if(gTriggerCount == gTriggerInterval){
            gTriggerCount = 0;
            digitalWriteOnce(context, n, gTriggerDigitalOutChannel, HIGH); //write the status to the trig pin
        } else {
            digitalWriteOnce(context, n, gTriggerDigitalOutChannel, LOW); //write the status to the trig pin
        }
        int pulseLength = pulseIn.hasPulsed(context, n); // will return the pulse duration(in samples) if a pulse just ended
        float duration = 1e6 * pulseLength / context->digitalSampleRate; // pulse duration in microseconds
        static float distance = 0;
        if(pulseLength >= gMinPulseLength){
            static int count = 0;
            // rescaling according to the datasheet
            distance = duration / gRescale;
            libpd_float("UltrasoundDistance", distance);
            //rt_printf("Sending distance to pd\n");
            if(count > 5000){ // we do not want to print the value every time we read it
                rt_printf("pulseLength: %d, distance: %fcm\n", pulseLength, distance);
                count -= 5000;
            }
            ++count;
        }
```

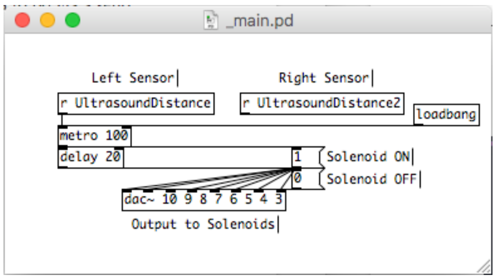
" libpd_float("UltrasoundDistance", distance); " is what passes the sensor data to the libpd part of Bela, and allows me to create a [r UltrasoundDistance] object in PureData to receive the data. The below screenshot shows the duplicated variables for both ultrasound sensors.



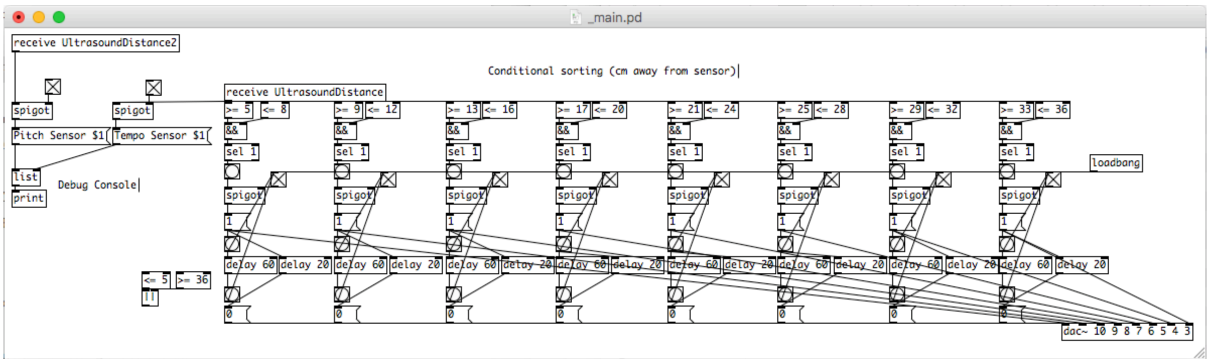
```
PulseIn pulseIn;
PulseIn pulseIn2;
int gTriggerInterval = 2646; // how often to send out a trigger. 2646 samples are 60ms
int gMinPulseLength = 7; //to avoid spurious readings
float gRescale = 58; // taken from the datasheet
unsigned int gTriggerDigitalOutChannel = 0; //channel to be connected to the module's TRIGGER pin
unsigned int gEchoDigitalInPin = P8_08; //channel to be connected to the modules's ECHO pin
unsigned int gTriggerDigitalOutChannel2 = 2; //channel to be connected to the module's TRIGGER pin
unsigned int gEchoDigitalInPin2 = P8_10; //channel to be connected to the modules's ECHO pin
int gTriggerCount = 0;
int gPrintfCount = 0;
int gTriggerCount2 = 0;
int gPrintfCount2 = 0;

int gBufLength;
```

6th December 2017

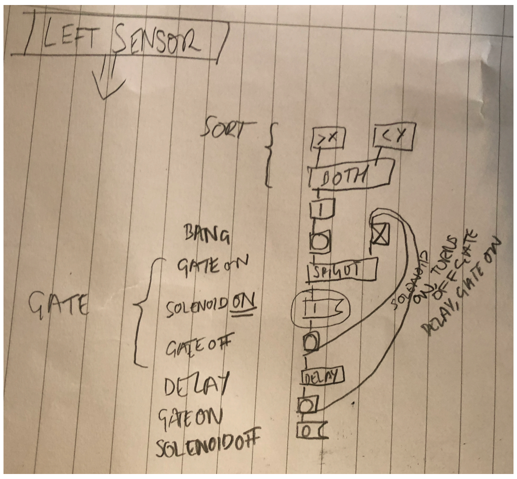


Commenting on my work in PureData proved to be an invaluable way to keep track of changes. The right screenshot shows my starting patch. I quickly got lost in my own code after implementing changes without testing after each time however. Below shows a screenshot of me trying to make sense of all the changes that I had implemented in a short time. Each column was previously part of an abstraction called _timing, that I scrapped and tried to work out why the patch wasn't working.

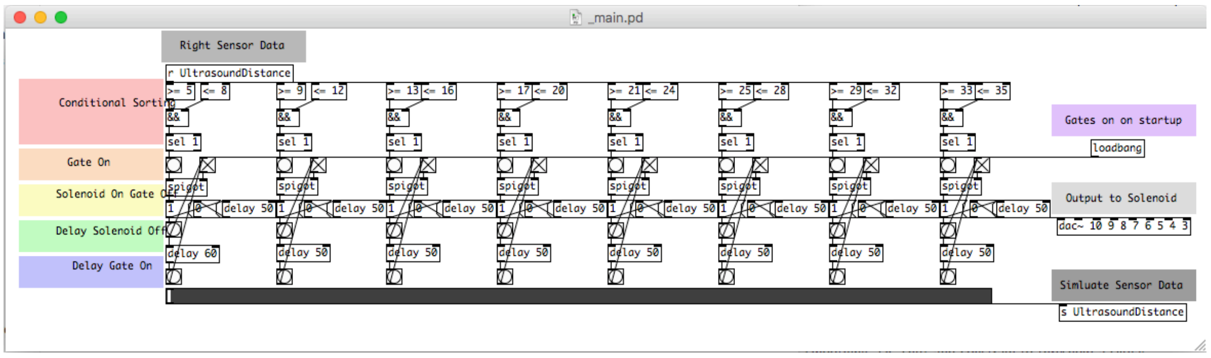


8th January 2018

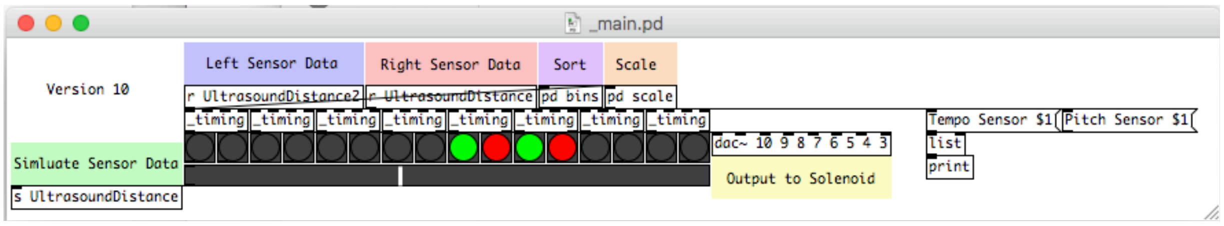
After a break, I came back to the patch and worked on it from the ground up. I drew on paper the process, this helped me visualise with more freedom the effect of each process on the data passing through it. It was through this process that I realised that I was sending the solenoids too little voltage (1V compared to the 2.4V I finally decided on that resulted in the best sound).

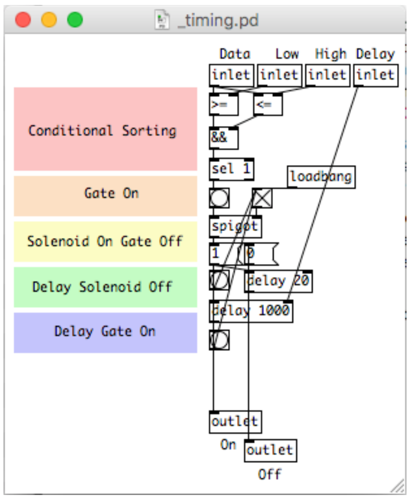


Below shows the ground up approach that I took to remake the patch, with comments at each change in process. At this stage, the patch was working well with the Therephone.



9th January 2018





At this stage in the development process, I was pleased with the patch, so I decided to create an abstraction for the solenoid columns shown in the 8th of January screenshot. The simulated sensor data slider I created helped in debugging this process without having to load up the Bela IDE each time I made a change.

12th January 2018

Committed to the final design changes of the Therephone. I decided on using Velcro to secure the solenoids to the central block, as it means that you can change the position of them if you so wish. Also I moved the ultrasound sensors onto the top of the breadboard, instead of hanging them off the side where they were slightly more precarious.