Musicap

Group 10

Imperial College London
Department of Electrical and Electronic Engineering

‘An Innovative System for Disabled Users to Create High Quality Music’

Second year group project- Final Report

SUPERVISOR
Peter Cheung

TEAM MEMBERS
Hamish Campbell (01194507), Ziwei Chen (01414164), Kai Wen Yoke (01409802), Bernard Tsui (01351908), Edward Turner (01364481), Shiyu Zheng (01334805), Jie Yechen (01390269)

DATE OF SUBMISSION

22.03.2019

TABLE OF CONTENTS

Abstract

The objective of the project is to give a creative outlet to people unable to play conventional musical instruments. In order to achieve this we have designed and built our own musical instrument. We had to find a way to allow disabled people to play our instrument with the often limited movement they had. The decision that we came to was to use head movements to play the notes. The design became a hat with an inertial measurement unit (IMU), mouthpiece and information processing, in order to play the notes. Playing music with the movement of the head is intuitive and enjoyable, especially for younger users. The IMU outputs a value for each of the 3 spatial dimensions, depending on the position of the user's head. This is fed into the arduino, which is connected to a module called the 'music maker'. This module can produce a wide range of sounds, from an in built library, which we map to a specific head movement. This allows for the product to be flexible, as you can easily increase the complexity of the instrument. The mouthpiece determines the volume and duration of the note. We concluded that the device is intuitive but still has a degree of mastery, with the ability to add more notes to more head positions depending on the user's level of skill. This gives the user a similar experience to learning to play a conventional instrument, which is the end goal of the project.

Introduction/Background

There are over 11 million disabled people in the UK. It is estimated that 1 in 50 people in the world live with some form of paralysis[1]. These are the people we want to help by providing them with an enjoyable and stimulating challenge. Some people who do not have full use of their arms or the fine motor control cannot play conventional instruments such as piano or guitar, however, our system will allow them to. Our device can replicate the sound of conventional instruments in a form more accessible to everyone.

The need for the product is more of a philanthropic endeavor rather than a purely profit motivated business venture. Despite this there still needs to be a want and desire for the product. Our correspondence with the 'One Handed Musical Instrument (OHMI) Trust'[2] and market research with a former Imperial student showed us that there is a need for such a project. The instrument industry is hard to break into, but we are selling to people who are not currently considered by the market. This product is innovative and a first of its kind. It also fits a gap in the market.

Design Criteria and How They Are Met by the System

1. **Performance-** The system should be able to continuously give real time feedback to the user, to resemble playing a musical instrument. It will be able to perform consistently so that the user may practice and improve their ability in order to play more complex music.

   The GUI on the laptop provides this feedback for the user, allowing them to physically see the notes that they are playing. The exact performance of the cap is determined by its setup- the device is flexible and can be adjusted to play a more simple set of notes or used to create a more complex composition. This is determined by the coding. The product allows the user to change these settings from a computer with relative ease. We can change the sound of the instrument, the volume and the range of head movements that gives the different notes.

2. **Life in Service-** The product is expected at a minimum to remain operational for a long practice session which could last around 3 hours. The target operational time will be almost twice this length at 5 hours, to provide a buffer. Ideally the product should also work for many years without breaking. However, we cannot properly test this criteria.

   The product worked for the entirety of the demonstration (3 hours) and the testing phase for which we used the prototype for a whole day without failure.

3. **Target Product Cost-** The estimated cost for our product was £150.

The real cost of the product was £120, however, if we manufactured many units, this cost would decrease significantly. Most of the money was used to buy the IMU and music maker shield (approximately £26 each). For more detail see appendix 1.

4. **Ergonomics-** Since the product will be used by people with physical disabilities, all the controls should be easily within reach. As the product could potentially be used for long periods of time in a single sitting, the product must be designed so that the user can adopt a comfortable posture while using it.

   The musicap consists of IMU and mouthpiece, which are solely controlled by head movement and breathing. The heavier electronic components were kept in the 3D printed box, seperate from the cap so that the user does not have to support the weight. We have used the Musicap for four hours in one sitting and have felt no fatigue.

5. **Customer-** Disabled users may be unable to control their body below the neck (i.e. unable to use their hands or have difficulty coordinating hand movements), but has full control above the neck. Possible disabilities: Cerebral palsy, stroke, amputations, Parkinson's. The system must give a cognitive challenge just like any other instrument. Social reasons, like joining a school band, could be a motivation for learning an instrument too.

   The final design does not require users to have control of their body below their neck, making it suitable for the disabled users we designed the Musicap for.

6. **Quality and Reliability-**

Performance Criterion	**Measurement**	**Expectation**
Accuracy	Note detection following head movement	100% being detected
	Volume of sound produced by mouthpiece	100% correct sound produced
Reliability	Operation lifespan	Works for at least a year
Mean time before failure (MTBF)/ Mean time before replacement (MTBR)	Operation lifespan	Multiple years, similar to other instruments. MTBR is the same as MTBF because instrument will be so inexpensive after mass production that it would be more economical to buy a new instrument.

After spending some time practising with the Musicap, we were able to consistently play the same notes by replicating the head movements. With some practice, we were also able to play notes at the desired volume by varying how hard we blew consistently.

7. **Time-scale-** The project is expected to finish before the demonstration (19th March). For more specific deadlines e.g. completion date for each module, a Gantt chart should be used.

   We successfully met all the deadlines through effective time management. The predicted Gantt chart and actual Gantt chart can be found in the appendix.

8. **Testing-** To test the product, we can have multiple volunteers use the product to ensure the design works for different people. Testing will occur at each stage of the project, for example, we must test the IMU, the mouthpiece and the other components individually once they have been done. The product should also be tested on disabled users to assess the quality of user experience.

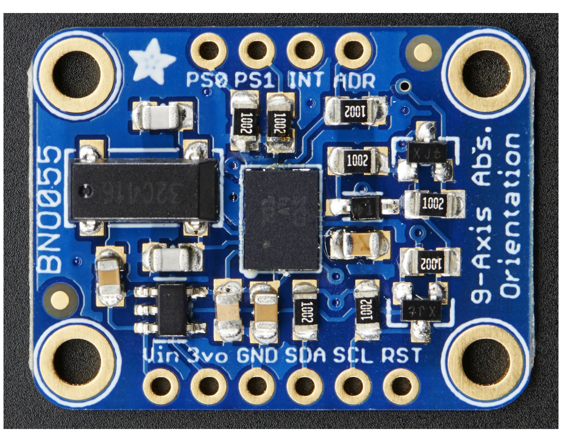
   We did rigorous testing with the finalised device by taking it in turns playing a short song. It worked well for all users, but we have not tried it out on disabled users yet.

9. **Safety-** Since there is little to no reason why the hardware of the product will need to be accessed by the user, there should be no exposed components e.g. wires.
   We placed as much of the electronic hardware into specially designed 3D-printed boxes as possible, thus protecting the user from electrical hazards.

Concept Development

Inertial Measurement Unit

In order to detect head movement, the Adafruit BNO55 Absolute Orientation Sensor[3] is used.



(Figure 1: Sensor Module)

Sensor description

The main feature of the sensor module is the BNO55 IMU which consists of an accelerometer, a magnetometer and a gyroscope and is thus a 9-axis IMU. It processes raw sensor data internally using sensor fusion algorithms and outputs the real orientation data for 3D space. Raw outputs from the gyroscope, accelerometer and the magnetometer before processing by sensor fusion algorithms can also be accessed.

In-built sensor fusion algorithm

The in-built sensor fusion algorithm is the crux of the sensor. Typically it is complex and computationally intensive to obtain a steady 3-axis orientation data output from the raw values of the gyroscope, accelerometer and magnetometer. However, the BNO55 comes inbuilt with a sensor fusion algorithm and a high speed ARM Cortex-M0 based processor that makes it possible to compute 3D orientation accurately in real-time. These are the reasons why we chose to use this module.

9-axis IMU - Gyroscope accelerometer, magnetometer

The absolute orientation data is computed using the raw angular velocity data from the gyroscope, the acceleration data from the accelerometer and the magnetic field strength from the magnetometer. All three quantities are required for accurate calculation of absolute orientation which involves integration, because having only the gyroscope and accelerometer (i.e. in a 6-axis IMU) would lead to integration drift. Over time, small errors will build up quickly in each direction and eventually lead to a drift away from the real data. Adding a magnetometer helps the sensing algorithm compensate for the drift over much longer periods of time and makes the calculation of absolute orientation more robust. This is of utmost importance in the Musicap because the position of the head should consistently match with a particular note for the user to learn how to use it as an instrument.

Pin assignment

The absolute orientation data from BNO55 is fed back to the Arduino through the following pin assignments. The pins Vin, GND, SCL and SDA of the BNO55 are connected to 5V, GND, A4 and A5 of the Arduino respectively. The SCL and SDA are the output data from the BNO55 and they can only be connected to pins A4 and A5 which are specifically designed for the two boards to communicate with each other via the I2C synchronous serial protocol.

Arduino Code

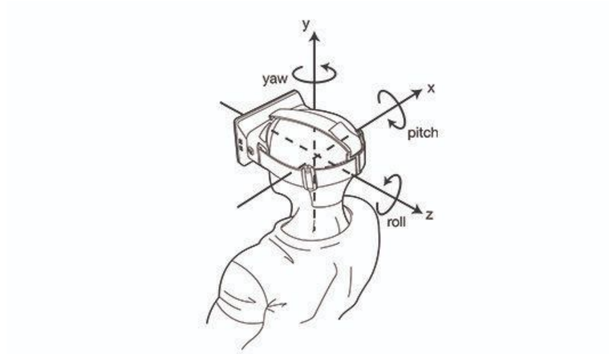
The Adafruit sensor module is compatible with the Adafruit_BNO55 driver[3] for the Arduino. This driver is based on the Adafruit Unified Sensor system, which provides an interface that takes in data from any supported sensor and returns the data in SI units. Using this driver it is easy to retrieve three axis orientation data (in Euler angles) in the Arduino code. Raw sensor data can also be accessed directly before processing by the sensor fusion algorithms.

```
//initialize bno55 data variables
sensors_event_t event;
bno.getEvent(&event);

//notes mapping
double orientx = event.orientation.x;
double orienty = event.orientation.y;
double orientz = event.orientation.z;
```
(Figure 2: Retrieving sensor data in Arduino code using adafruit_BNO55 driver)

Output Data Description

There are 3 directions (X,Y,Z) of orientation referring to the outputs from BNO55.



(Figure 3: Demonstration of the project)

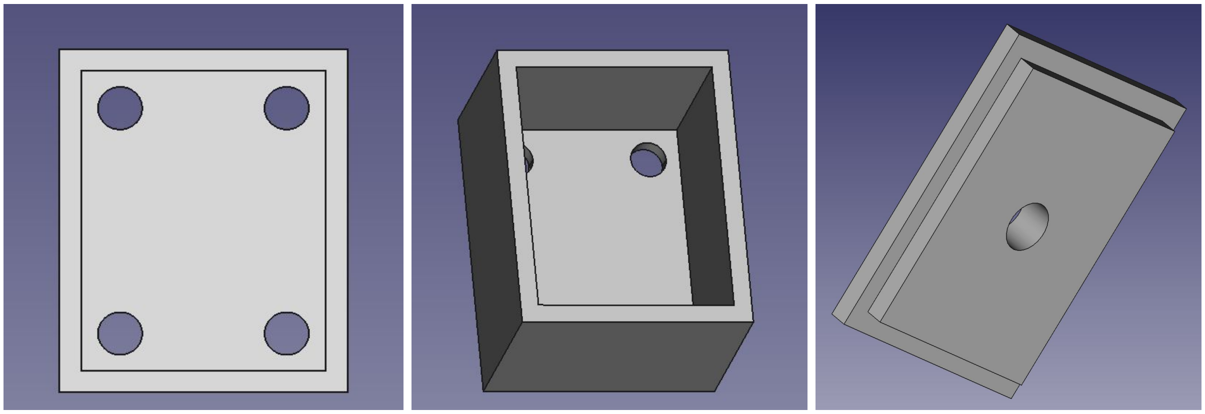
Quantity	X	Y	Z
Euler Angle	0 to 359	-90 to 90	-180 to 180

(Table 1: Euler Angel)

The reason why angular velocity and acceleration are not being used is that it is difficult for the user to move their head with a specific speed in the desired range, since the user will first have to locate the correct note position. It also causes dizziness if the user moves too quickly. Therefore, only orientation of the head motion is employed by using Y and Z directions. The X direction has been ignored because it was unstable according to our testing. It is a relative value which varies according to the user's position, while Y and Z are absolute values, which take the horizontal plane as the reference.

Structural Design

Since the IMU needed to be mounted on top of the cap to allow the user's movements to be recorded, the issue of how to mount it had to be solved. To give the module maximum protection and allow for a sleek design, it was decided that a 3D printed box was the best solution. This box also had to somehow be mounted on the cap. By sewing the box on to the hat, it allows maximum comfort and safety for the user, while ensuring a robust and strong connection to the cap. The holes in the box, seen below, match the screw holes in the IMU module allowing the two parts to be joined easily. A simple CAD design was created that used a semi-detachable lid to allow access to the module when needed:

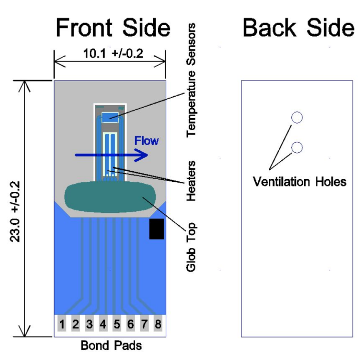


(Figure 4: 3D printed box design for IMU)

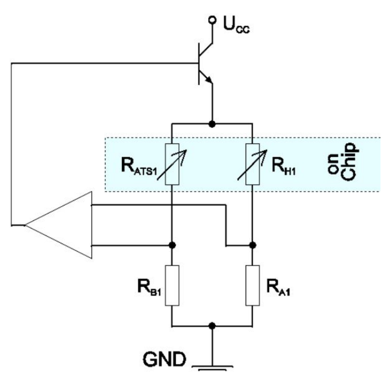
This box can be can be seen in figure 4 as the orange box mounted directly on top of the cap.

Mouthpiece

The first sensor that was purchased was the ‘LMM-H03 Mass Air FLow Sensor’[5]. This sensor was preferred over pressure sensors as it would be much easier to design the housing around it. It also meant the housing could be designed later as the LMM-H03 could be tested on its own, as opposed to the pressure sensor where the housing would have to be designed first to ensure that it was able to measure the user’s breath. The circuit functions on the same principle as a hot wire anemometer. Inside the chip, there are two heaters and two thermistors that can be connected via the bond pads of the chip to the circuit.



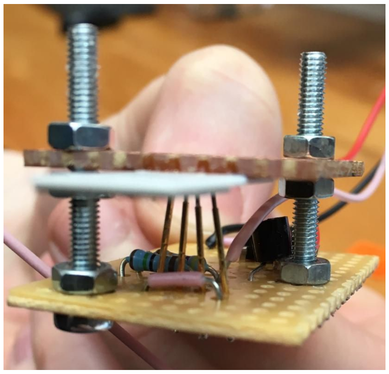
(Figure 5: Overview of sensor)



(Figure 6: Wheatstone-bridge circuit)

Technical Reasoning

Spring test points were used to connect the chip to the Wheatstone-bridge on the veroboard. This decision was made because the area of the pads were very small, making it difficult to solder wires to. Another benefit of using the spring test points was it allowed the chip to be removed from the module while further adjustments were made to the circuit, reducing the risk of damaging the chip. The spring test points were soldered onto the veroboard with the circuit while the sensor was stuck to a smaller veroboard which was attached to the main board with screws to apply pressure in order to make contact with the test points.



(Figure 7: Spring test points)

The op-amp in the circuit keeps the voltages at the two inputs the same. The resistor chain with the heater was connected to the inverting input (-) while the resistor chain with the thermistor was connected to the non-inverting input (+). The thermistor was tested by passing a heat source over it to heat it up and blowing on it to cool it down. It was found that the resistance was proportional to the temperature i.e. $R \alpha T$.

When air is blown over the sensor, it cools the thermistor down; decreasing the resistance. This causes the voltage at the non-inverting input to be greater than at the inverting input. The voltage at the output of the op-amp increases, which will increase the voltage to the two resistor chains. When the voltage across the heater is increased, it heats up more. Once the heater has heated the thermistor to the original temperature, before the disturbance was introduced (blowing over it), the ratio of resistors are the same again so the two voltages going into the op-amp are the same. The circuit effectively tries to keep the thermistor at a constant temperature by varying the voltage across the heater to compensate for how much the thermistor is cooled down by someone blowing over it. The BJT (BC337) in the circuit is connected as a common collector, acting as a voltage follower to ensure a sufficient current is supplied to the heater.

The voltage at the non-inverting input is connected to the arduino's analog input (A3). This voltage varies from 1.27V to 1.52V, depending on if the user is blowing into the mouthpiece and how hard they are blowing.

When blowing across the sensor (that is heated slightly above room temperature), there was no noticeable change in the voltage, therefore a temperature much higher than room temperature was chosen to ensure that there would be a more noticeable change when breath was passing over the sensor. In order to pick the resistor values, a soldering iron heated to 100℃ was placed over the sensor to heat the thermistor up to around 80℃, the resistance of the thermistor and heater were measured (R_{ATS1} = 1527Ω, R_{H1} = 46Ω). Resistors values R_{B1} = 1500Ω,

R_{A1} = 47Ω were picked to match the measured values to set the operating point to the temperature defined by the soldering iron (≈80℃).

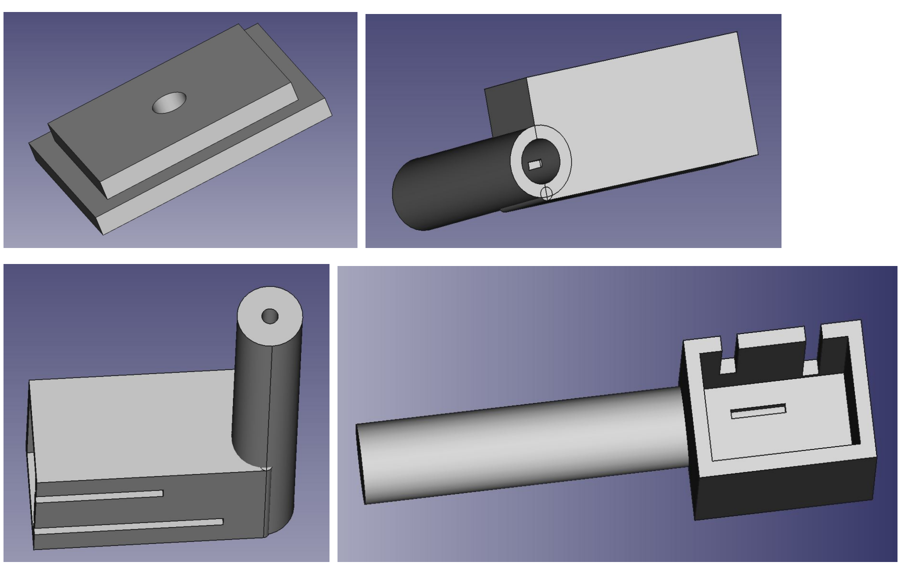
The MC33078 (op-amp) datasheet[6] was consulted to ensure that it was compatible with the circuit. Since the resistors used in the circuit were all ≤ 1600Ω, the input impedance of the op-amp was high enough to not significantly impact the behavior of the circuit. The maximum bias current of the op-amp is 800nA. Since the resistors used were quite small (≤ 1600Ω), the current drawn by the op-amp is not significant compared to the overall current in the circuit. The common mode voltage range of the op-amp is ∓14. Since the power supply to the circuit ranges from 0-5V, the inputs will be within the range required by the op-amp.

Structural Design

The structural design for the mouthpiece proved challenging in the development stages, however, in the end a successful design was created. Three main difficulties (and thus design rationale) had to be considered:

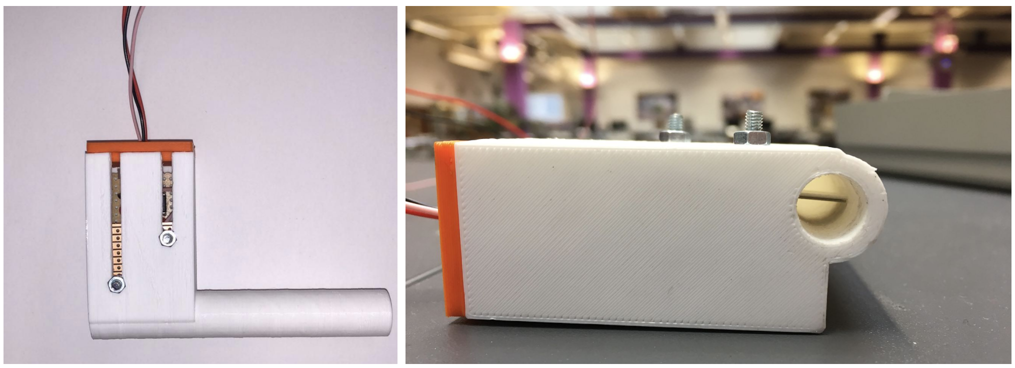
1. The hole for air to be blown through by the user, could not be too wide, otherwise the user would use all of their breath too quickly. This was a design point that we noticed when researching the mouthpieces of other instruments such as clarinets/recorders.
2. The hole also couldn't be too small, or the flow rate would be too high. Having experienced one flow sensor break already, it was clear that the sensor had to be treated with care.
3. The electronics required to produce the necessary signal using the sensor, had to somehow be encased. This would stop the user from unintentionally damaging the design. This was especially important because of the 3-dimensional design of the circuit (due to the spring test points) which meant if it was not treated with care, the test points may move from the required positions.

This led the group to design a structure that met all three of the design criteria:



(Figure 8: structural design for mouthpiece)

The thinner opening of the tube is blown into by the user, thus providing a small radius and so meeting rational point 1. However, 2cm down the length of the tube, it tapers out. This provides a wider radius, to reduce air flow strength where the flow sensor is placed (meeting rationale point 2). Finally, a box is joined onto the tube to house the necessary electronics. This has an opening where the flow sensor passes through to the tube so that the temperature sensor is in the middle of the air flow, to give the best possible performance. Gaps in the box also provide rails that are used to secure the position of the electronics using M3 nuts and bolts. Finally, a similar design to the IMU is used for the lid of the housing to allow access when needed. The complete design was printed and used for the prototype:



(Figure 9: 3D printing mouthpiece)

It is clear to see that the electronics are housed away from the user. The flow sensor is also visible inside the tube in figure 8.

Adafruit Music Maker Shield

(Figure 10: Adafruit Music Maker)

The music maker shield converts MIDI messages into audio sound. MIDI mode is not activated by default. To turn it on, GPIO1 needs to be connected to 3V logic and the RX pin should be connected to digital pin 2.

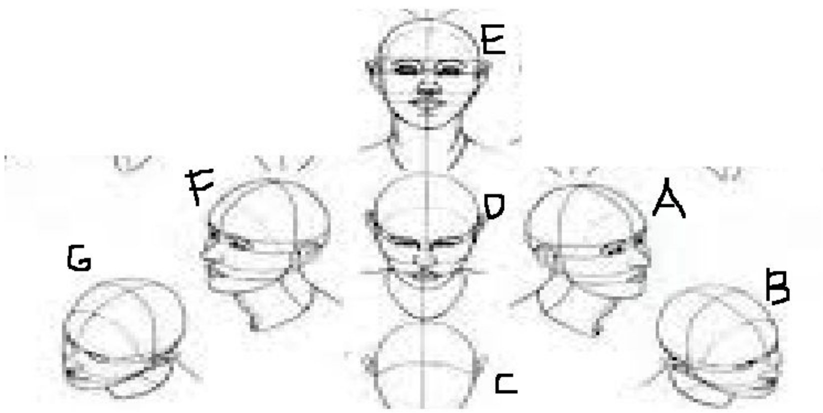
The Adafruit music maker provides various kinds of instrument capabilities that can be selected. The choice of the instrument and the volume is set by the Arduino code. In this project, the piano is chosen since the sound produced is more classical and piano is one of the most common instruments that people like to play.

Mapping MIDI messages to the music maker

The arduino maps the orientation data from the BNO55 IMU to a number from 0-127 representing a musical note. The breath-strength data from the flow sensor is also mapped to a number from 0-127 representing the volume.

Mapping head position to notes

For the purposes of the demo, we simplified the mapping of head positions to their corresponding notes so that our demonstrator could memorize the head positions quickly. The code for the mapping can be found in appendix 3. The following diagram shows how the head positions were mapped to their corresponding notes during the demo:



(Figure 11: Head positions matched to the corresponding notes)

Orientation angle in Y-axis	Orientation angle in Z-axis	Note number (MIDI)	Actual note
10 to 25	120 to 140	60	C
10 to 25	140 to 160	62	D
10 to 25	160 to 180, -180 to -165	64	E
-10 to 10	160 to 180, -180 to -165	65	F
-35 to -10	160 to 180, -180 to -165	67	G
35 to 50	160 to 180, -180 to -165	69	A
50 to 65	160 to 180, -180 to -165	71	B

(Table 2: Table showing how IMU orientation data values are mapped to MIDI notes)

Head positions on the Y and Z axes are mapped to notes in the range of 1 octave. On the Z axis, the user moving their head fully forward corresponds to middle C. The chosen note increases as the head is moved up along the Z axis to note E. On the Y axis, the user tilting their head to fully to the left corresponds to G. The note value increases as the head is tilted further to the right to until B is reached. The mapping of head positions to notes can be customized for different levels of difficulty.

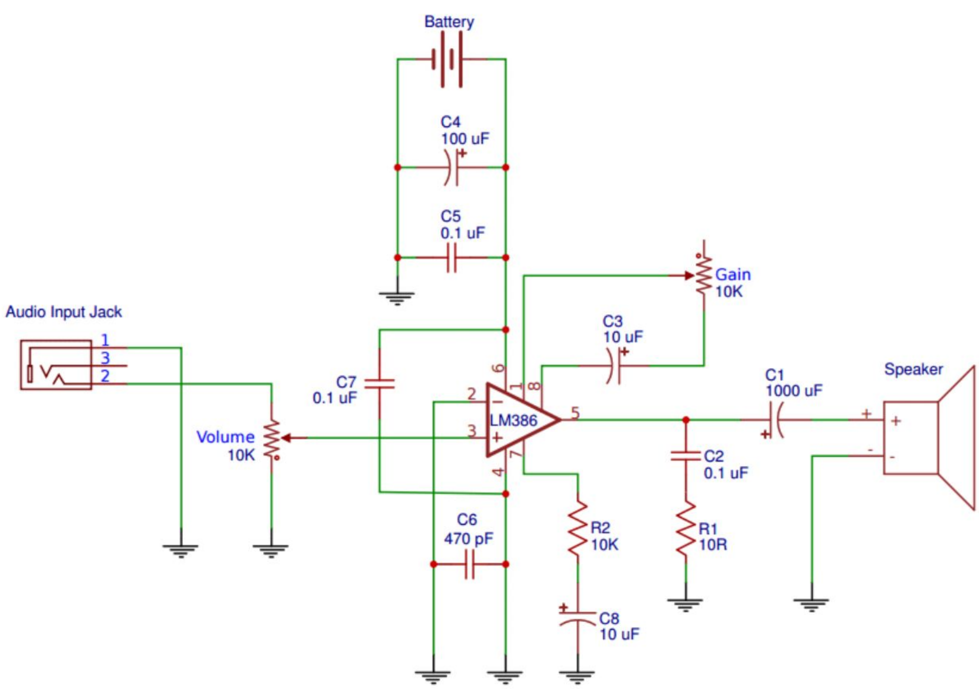
Mapping breath strength to volume

When the user is not blowing into the mouthpiece, there is a DC offset that is read by the Arduino as a value in the range of 260 to 275. Initially, we intended to map the breath-strength to all 127 settings for volume. However, blowing very hard into the mouthpiece yields a value of around 300 on the arduino. A small breath gives about 280. It is not practical to blow hard into the mouthpiece for a long period of time, so we decided to have 3 volume settings: soft, medium and loud. The corresponding arduino code can be found in appendix 3.

Audio Amplifier

As the volume of music produced though the speaker was not loud enough originally, we decided to use an audio op amp to amplify the output. The LM386[7] was chosen since it is a power amplifier designed for use in low voltage consumer applications, especially for audio amplification. It can amplify the input audio signal by anywhere from 20 up to 200 times.

The schematic of the circuit is shown below:



(Figure 12: schematic of amplification circuit)[4]

On the input side of the op amp, C8 and R2 are connected in series to decouple the audio input signal. C6 is a 470pF capacitor which is placed between the positive input signal and ground. It filters the radio interference picked up by the audio input wires. The capacitors between the positive and negative power supply, C4 and C5, are used to decouple the power supply. The 100 μF capacitor will filter low frequency noise while the 0.1 μF capacitor will filter high frequency noise. The 10k variable resistor at the input of the op amp is for volume adjustment.

On the output side, C1 acts as a low pass filter used to remove the noise of the output. C2 in series with R1 will filter out the high frequency components since the output notes to the speaker are low frequency (around 1000Hz).

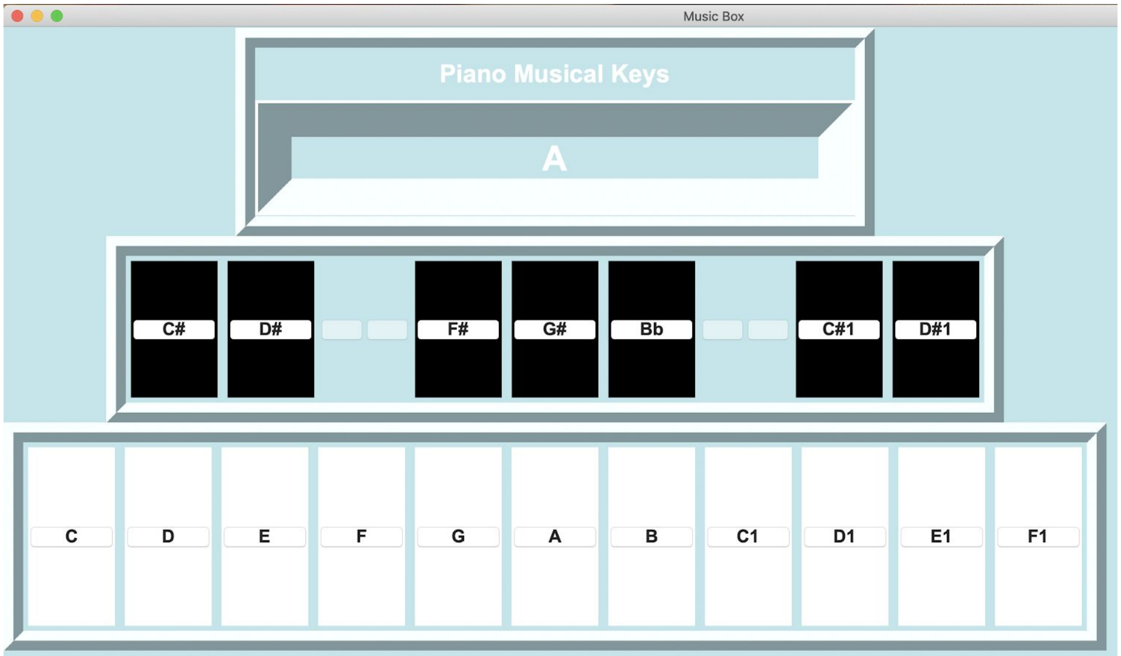
C3 is connected in series with a 10k variable resistor between two gain setting pins of the op amp. The gain would be 20 without the capacitor. However, with the 10 μF capacitor, the gain reaches it maximum value which is 200. The variable resistor adjusts the gain between 20 and 200, and C7 is used for additional decoupling of the power supply from the chip.

Ceramic capacitors perform better at high frequency since they are not constructed with an internal coil, which means the parasitic inductance is small compared with electrolytic capacitors. In contrast, electrolytic capacitors have an internal coil, so they would be used for low frequencies. All the values of the capacitors are chosen based on the frequencies that need to be rejected.

Graphical User Interface

Other than the audio feedback, visual feedback is used as well to make the product more user-friendly. The visual interface on the laptop shows the user the current note. This corresponds to the position of the user's head, thus helping the user adjust their head to the correct position.

The visual interface is written from scratch in python script using Tkinter[8], a well-documented python library intended for GUI design. The interface emulates the design of piano keys and the key that is being played turns grey when it is being played by the user.



(Figure 13: visual interface)

In order to display the note being played on the GUI, the arduino sends the string representing the note to the laptop wirelessly via bluetooth.

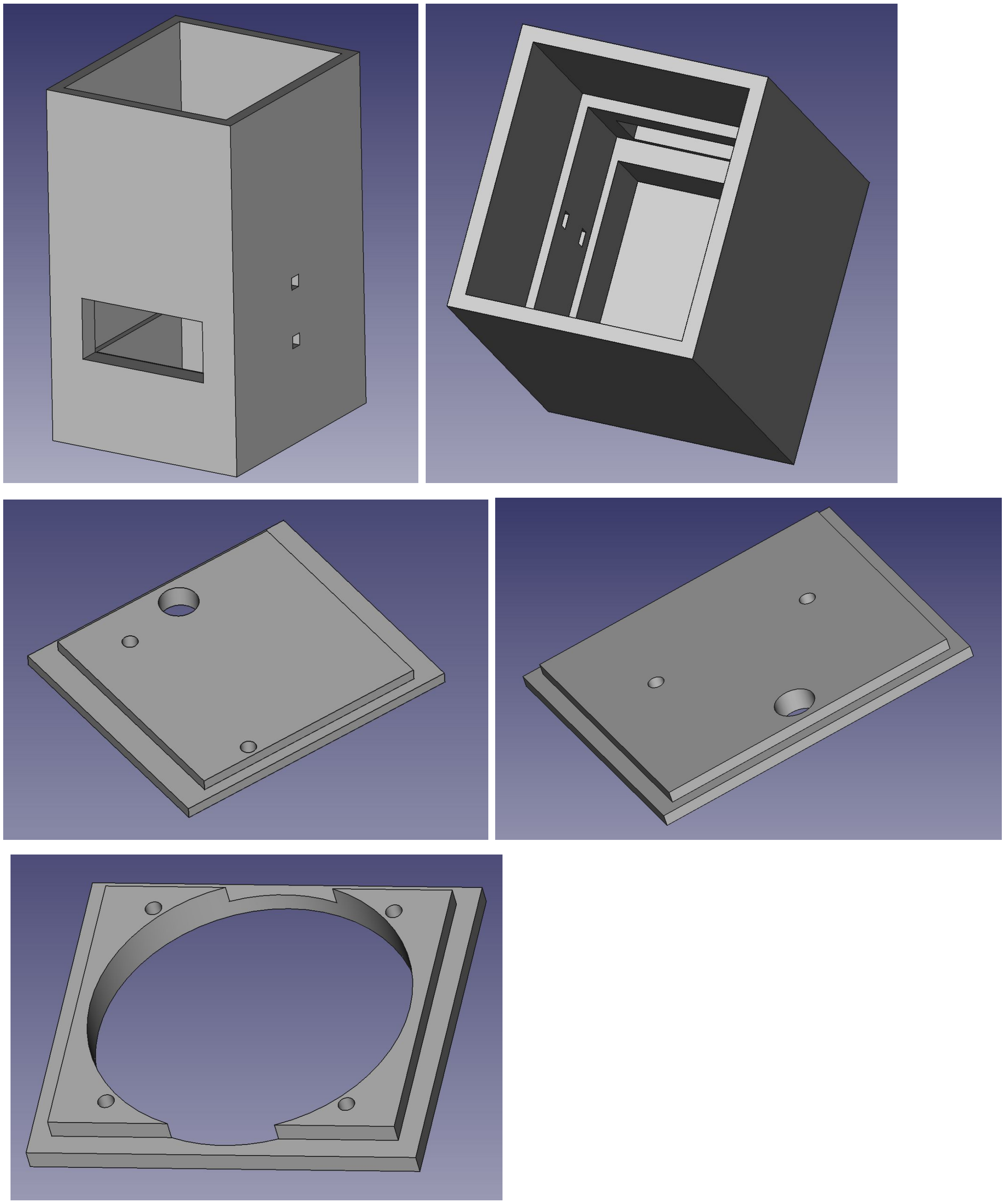
Bluetooth

The Adafruit Bluefruit LE UART friend bluetooth module[9] is used in the product. It has a built in library in python (Bluefruit LE Python Library[10]) which works only on linux and MAC. The python library is well-documented and provides a starting point for figuring out how to write code to connect and communicate with the correct bluetooth device. For the python script, an object-oriented approach was taken, since this allowed us to update the GUI (displaying notes played) and continuously check for incoming data from the arduino.

Speaker Box

As the prototype grew both in size and complexity during the course of the project, it became clear that mounting all of the necessary parts on the hat was infeasible. This was due to both the weight and size of the different parts. Thus, it was decided that having a box containing parts of the project that did not require mounting on the hat, would be a beneficial design improvement. The parts that needed to be on the hat were the IMU and mouthpiece. This left the batteries, arduino, speaker driver circuit and the speaker. Although the design process was complex and challenging, the end product came together nicely.

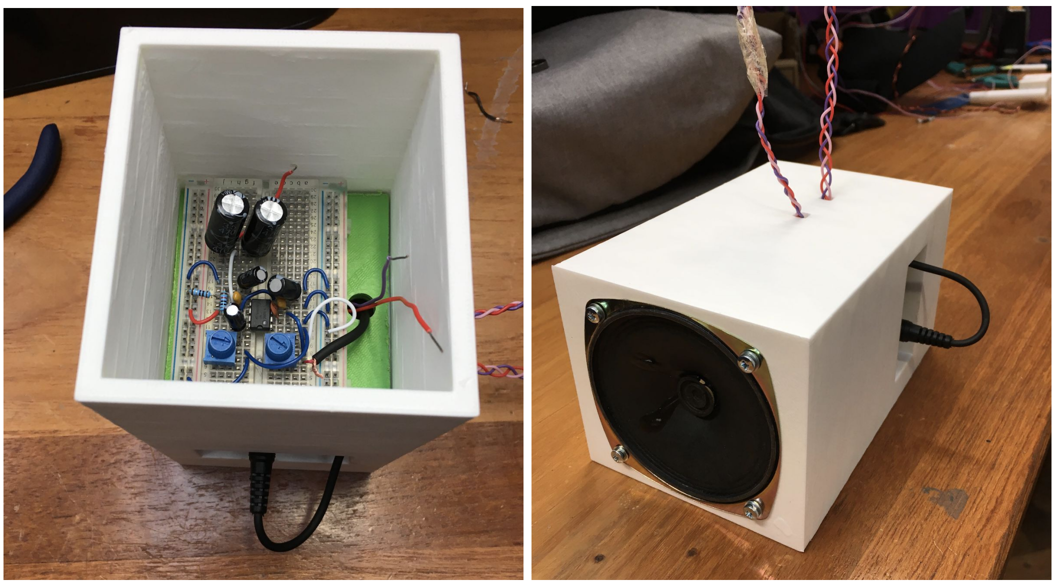
As throughout the project, it was important that the user did not have access to all the sub-systems, to ensure both the safety of the user and the negation of any damage. Thus a layering system was decided on that allowed all of the sub-systems to link to each other inside the box, while being fixed in place so that they weren't damaged during the movement of the system. The following designs were created to implement this:



(Figure 14: structural design for speaker box)

On the bottom layer of the box, the batteries are placed. They fit snugly so that no movement occurs when the box is moved around. The arduino was then bolted down on to the first partition and placed into the box. The large holes on each partition allow interconnections to be made. Furthermore, a large cutout was made in the side of the box to allow the user to access the arduino connections and turn the power on and off. The next partition has the driver circuit screwed on to it, this was placed above the arduino. Finally, the speaker is placed onto the top lid and bolted down. This is the only part that has any exposure to the user. However, this is needed to allow music to be heard. Finally, two holes are made in the side of the box to allow a lanyard to be linked through, in order for the user to have this around their neck. The box will rest on the user's legs and so that there will be no weight on the user's neck. The lanyard is simply there in case the box were to fall off and cause damage to the system. The reason we chose this instead of a shoulder mount or bag pack for example, to cater for all severities of disability including full arm amputees.

Below, the actual prototype design is shown, where the layering is visible and the fully enclosed sub-systems can be seen. The box provides protection and also gives a sleek design:



(Figure 15: 3D printed speaker box)

Prototype, simulation, demonstration

For the demonstration phase of the project, the subsystems were joined together to create a working prototype of the product. Thorough testing was conducted to check that the mappings for each note where in comfortable locations for the user to move their head into. Using the speaker, audio feedback was also supplied allowing us to check that the correct note was played. Once fine tuning had been completed, the prototype was complete. However, just like any other musical instrument, the user must practice before any meaningful music can be played.

(Figure 16: Product testing)

After practising for one afternoon, Hamish was able to play some basic music using the system. Songs including: ‘Happy Birthday’, ‘Twinkle Twinkle Little Star’ and ‘Mary Had a Little Lamb’ were demonstrated to both assessors and the public during the demonstration. A video of the system being used to play some basic songs during the testing phase are shown in the following video: https://youtu.be/UX_DPvC_nr0.

Although these are basic examples, even the idea that someone who may never have been able to play music before could play ‘Happy Birthday’ for a family member, illustrates the benefits that this product has on the target market.

Critical Analysis and Evaluation

Mouthpiece

When the mouthpiece subsystem was tested using the power supplies in the lab, the output voltage was very consistent for a given input. However, when the circuit was powered via the arduino, the output voltage was not stable and would fluctuate around 1.27V even when not blowing across the sensor. To compensate for this, a threshold level was set in the Arduino to ensure that notes would only be played when someone was blowing into the mouthpiece as opposed to fluctuations in the baseline voltage.

Amplifier and Speaker

The amplifier circuit is spatially inefficient. To save space, the circuit could be constructed on a veroboard instead of a breadboard. Another improvement would be the quality of the output music. Since there is a distortion of sound caused by the speaker, a better quality speaker is needed for high quality audio output.

GUI

For the bluetooth connection between the system and the computer, Linux was used in the first instance. However, there were various errors with the dbus file within the python library PyBluez. It was not feasible as it was only successfully connected a few times, while MAC was more reliable. The GUI and bluetooth script were written separately and when trying to combine the two codes there were threading errors (parallel programming). In order to solve this, we took an object-oriented approach. This way, the bluetooth will be run first and call continuously to the GUI object to update the interface.

Speaker Box

The speaker box was able to hold all the different components securely. There was also enough space for all the connections between the levels. However, in the original design, the batteries were placed at the bottom level. This means that it is quite tricky to access the batteries and could prove challenging to replace them. All the wires connecting each layer were contained within the box except for the AUX cable which stuck out the side before going back into the box. In future iterations, an access point could be placed on the bottom of the speaker box to allow easy access to the batteries as well as using a more flexible AUX cable so that it won't stick out of the box.

Batteries

The original plan was to power the Musicap with 4 AA batteries. However when the Musicap was tested with the batteries, there was only enough power to turn on the Arduino and not to power the other sub-systems, so a computer had to be used to power the Musicap. In future designs, other power solutions will need to be explored to make the Musicap fully portable.

Project management

We met as a group once or twice a week depending on the workload. We used a Whatsapp group chat to organise these meetings. The Gantt chart gave us a schedule to stick to. The project was split into each submodule for subgroups to work on, this division of labour allowed more efficient work and the meetings allowed us to collaborate so that the modules would still be compatible. We shared a google drive folder to bring together all of our work and it allowed us to work on the same documents at the same time.

Project plan

The main plan for the project is shown on the Gantt Chart created for the preliminary report. This can be found in the Appendix 4.

However, we tracked the actual time that was spent on each activity every week, thus the Gantt Chart in Appendix 5 shows the actual time allocated to each activity. In the end the prototype was completed on time. This was made possible by regular meetings and good planning.

Tracking progress

After every meeting we would track our minutes and write down what we need to do and what we had learnt in the meeting. This was very useful when we were meeting without the supervisor for advice. See example of one of our meetings minutes in appendix 2. It shows how we allocate work and track the specifics of each sub-system. The minutes are shared on the google drive so that everyone has access to them. If anyone wanted to know the current progress of a module outside of a meeting this was the main way it happened.

Future Work

Although the prototype was successful in proving that the project concept was viable and worthwhile, there are of course future improvements that could be made to improve the system. Some such improvements that the group want to implement in the future are:

- Add more octaves of notes, allowing the user to have the ability to play more complex music than that played during the demonstration. However, this would also require the user to be skilled with the system which would require many hours of practice. Furthermore, if a way of playing chords could be developed, this would add further complexity to the device that users could exploit.
- Complete multiple rounds of testing with disabled users to get feedback on the system. This would allow us to make improvements that we had not thought of and make sure the most important elements for the target market are given the highest priority.
- Make the sensors wireless. This would mean no wires would have to go to the cap, allowing a more aesthetically pleasing and user-friendly design. This would also mean that batteries will need to be introduced for these sensors.
- Develop an easier system for the user to change the instrument sounds that are produced. Having to alter the code can lead to mistakes and is not user friendly. If an easier method was developed such as a simple switch, this would be a much better solution. The flexibility of being able to choose the sounds that are produced would also lead to the user gaining an advantage over conventional musicians.
- A manufacturing process would have to be explored. In order to produce many thousands of units, a better method of manufacturing the product must be developed (see below).

Industrial Design, Ergonomics and Manufacturing Considerations

To produce this product on a mass scale we would need to produce our own chips and parts in order to reduce costs. For example rather than using a whole arduino to do one job, we could create a simple chip that only has the sole purpose of processing IMU values into midi notes.

This would vastly reduce the cost of the product and make the processing component much smaller allowing it to be more compact.
There would be many considerations in the manufacturing. The assembly would be crucial. Weight is a large consideration because if the design is too heavy it could end up straining the user's neck when the device is used for long periods of time. We used 3D printing for a lot of our parts in the demo. In further manufacturing we would use other materials. A more sturdy plastic to protect the electronics could be used. It would also look better and cleaner. This could be achieved with more advanced 3D printing or outsourcing to a company that could produce the boxes to our dimensions. It would be cheaper in the long run for us to manufacture our own plastic casings.

In terms of the ergonomics of the product we would have to conduct further testing with disabled users. By finding out what positions are comfortable for people after long periods of time we could hone the placements of the different notes in 3D space. We would also be able to learn which disabilities require further support or development, before being able to use the Musicap.

Conclusion

Overall the project has been a success. The cost is relatively low and it achieves its purpose. If we went forward with the project as a genuine business venture we would have some amendments to add as discussed in the future work section. Each module of the project (mouthpiece, IMU and information processing) worked well in isolation and when brought together, the system effectively solves the problem proposed.

Bibliography

[1]Paralysis, L. (2019). Stats about paralysis - Living with Paralysis. [online] Reeve Foundation. Available at: https://www.christopherreeve.org/living-with-paralysis/stats-about-paralysis [Accessed 21 Mar. 2019].

[2]“OHMI Trust: Enabling Music Making for the Physically Disabled,” The OHMI Trust. [Online]. Available: https://www.ohmi.org.uk/. [Accessed: 21-Mar-2019].

[3]Learn.adafruit.com. (2019). Overview | Adafruit BNO055 Absolute Orientation Sensor | Adafruit Learning System. [online] Available at: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview [Accessed 21 Mar. 2019].

[4]Basics, C. (2019). Build a Great Sounding Audio Amplifier (with Bass Boost) from the LM386. [online] Circuit Basics. Available at: http://www.circuitbasics.com/build-a-great-sounding-audio-amplifier-with-bass-boost-from-the-lm386/ [Accessed 21 Mar. 2019].

[5]Connectivity, T. (2016). LMM-H03 Mass Air FLow Sensor. [online] TE Connectivity. Available at: https://www.te.com/commerce/DocumentDelivery/DDEController?Action=showdoc&DocId=Data+Sheet%7FLMM-H03%7FA%7Fpdf%7FEnglish%7FENG_DS_LMM-H03_A.pdf%7FCAT-MAF0001 [Accessed 21 Mar. 2019].

[6]Instruments, T. (2006). DUAL HIGH-SPEED LOW-NOISE OPERATIONAL AMPLIFIER. [online] Texas Instruments. Available at: https://www.ti.com/lit/ds/symlink/mc33078.pdf [Accessed 21 Mar. 2019].

[7]Instruments, T. (2004). LM386 LOW VOLTAGE AUDIO POWER AMPLIFIER. [Online] Texas Instruments. Available at: http://www.ti.com/lit/ds/symlink/lm386.pdf [Accessed 21 Mar. 2019]

[8] “Graphical User Interfaces with Tk¶,” Graphical User Interfaces with Tk - Python 3.7.3rc1 documentation. [Online]. Available: https://docs.python.org/3/library/tk.html. [Accessed: 21-Mar-2019].

[9]Cdn-learn.adafruit.com. (2019). [online] Available at: https://cdn-learn.adafruit.com/downloads/pdf/introducing-the-adafruit-bluefruit-le-uart-friend.pdf [Accessed 21 Mar. 2019].

[10]Learn.adafruit.com. (2019). Library Usage | Bluefruit LE Python Library | Adafruit Learning System. [online] Available at: https://learn.adafruit.com/bluefruit-le-python-library/usage [Accessed 21 Mar. 2019].

Appendix

1. Cost Analysis

Item		Indv cost	Quantity	Subtotal cost
Sensors	Flow Sensors Bi-directional AFW	9.69	1	9.69
	Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout - BNO055	26.67	1	26.67
Music production	Audio IC Development Tools Music Maker Shield for Arduino w/3W Amp	26.67	1	26.67
	lm386n-4 audio op amp	2.01	1	2.01
	Adafruit Accessories Speaker 3"	1.49	1	1.49
	Audio Jack	7.07	1	7.07
Micro-controller	Arduino Uno	16	1	16
Wireless communication	Adafruit Bluefruit LE bluetooth development module	12.92	1	12.92
Others	Jumper Wires Qty 6, Eight Pin STK Hdrs for Arduino	4.72	1	4.72
	Veroboards (medium)	1.76	2	3.52
	Veroboard (small)	0.22	1	0.22
	Sprung Test Points	0.991	10	9.91
	Cap	2.5	1	2.5
TOTAL				120.89

2. **Example of meeting minute**

22 Jan 19, 12-12.30pm

- Music maker, breath sensor spoilt
- Music maker too soft, buy speaker, might consider making our own op amp amplifier. Battery issue, will it be too large? Are we expecting speaker to be placed on head as well?
- Gyroscope working fine and values for different quantities are available and sensitive to movement. But haven't figured out what exactly the quantities mean. Will test it with the music maker to decide what quantities to use.
- Breath sensor is fragile, need to think about how to protect it. Hamish has some ideas on making 3D printed shell to secure stuff in place, considering this serrator/contact/connector spring probe to fix the fragile sensor onto the shell.
- Placed order for music maker, breath sensor, one speaker, one set of arduino header.
- Shiyu and Kai Wen to help with 3D Printing for breath sensor, meet Hamish on Friday to talk about how to design the 3D Printing stuff

3. **Code snippets**

- Arduino code for mapping breath to three settings of volume:

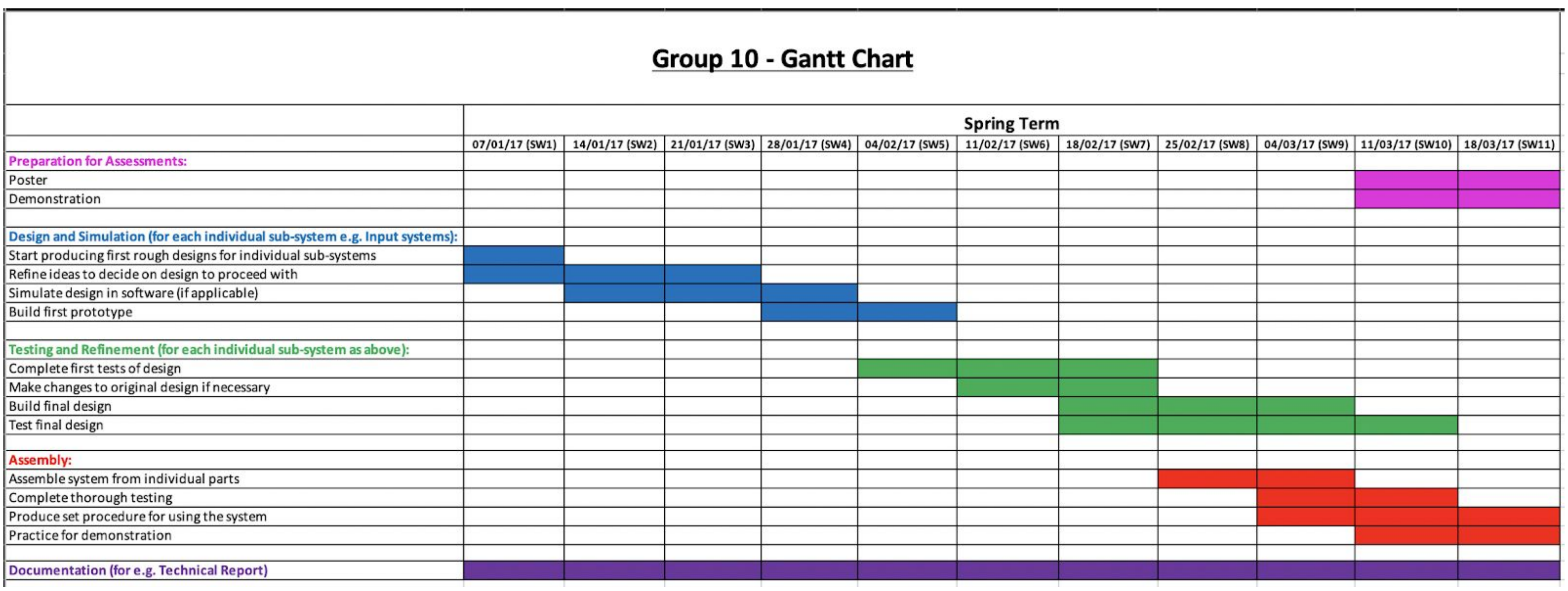
```
int maptovolume(double breath) {
  if (breath>=275 && breath <285) {
    return 80;
  }
  else if (breath>=285 && breath<295) {
    return 100;
  }
  else if (breath >=295) {
    return 127;
  }
}
```

- Arduino code that ensures that the correct note is played continuously while breath is detected

```
//notes mapping
double orientx = event.orientation.x;
double orienty = event.orientation.y;
double orientz = event.orientation.z;
midinote = maptonote(orientx, orienty, orientz);

if (breath > 275) {
   midiNoteOn(0, midinote, maptovolume(breath));
   while (breath > 275) {
      midiNoteOff(0, 0, 0);
      breath = analogRead(analogPin);
   }
}
else if (breath > 260 && breath < 275) {
  midiNoteOff(0, midinote, 0);
  Serial.println("no volume");
}
```

4. Gantt Chart- Predicted



5. Gantt Chart- Actual