Interim Report

by Nishat Tarannum

Submission date: 09-Jan-2019 12:55AM (UTC+0000)
Submission ID: 985296410
File name: Interim_Report_nt161.docx (819.91K)
Word count: 3125
Character count: 17628

UNIVERSITY OF
LEICESTER

~~Interim Report~~

Module code:	**EG3005**
Module name:	**Third Year Project**

9th January 2019

Author(s):	**Nishat Tarannum**
Student ID(s):	**169018358**
Degree:	**MEng Software and Electronic Engineering**
Tutor/Project supervisor:	**Mr. Andrew Norman**

SUPERVISOR'S COPY/EXAMINER'S COPY [For 3rd/4th year projects only - delete as appropriate]

Summary

The aim of this project is to make a portable, heart rate monitor which will be able to diagnose any abnormal heartbeat patterns such as arrhythmias. This device will enable people to know immediately if they are suffering from such heart conditions without going to a doctor. The device shall be compact and easy to use to allow people of most ages to use it comfortably. This report covers the work done, decisions taken, and the various challenges faced so far. It further details any future plans and improvements.

Contents

1 Introduction and Background

The purpose of this project is to make a device which is able to detect any irregularities in the heart beat pattern. Users will be able to diagnose any issues immediately (such as arrhythmias) without consulting a doctor or having to do an electrocardiogram (ECG). The device will be portable, it will be implemented using a Raspberry Pi, an Arduino and a photoplethysmographic (PPG) sensor which will be the substitute for ECG electrodes. This will increase usability as people of most ages will be comfortable in using the device with minimal technical knowledge.

ECG electrodes were not considered because for an ECG three points of different voltage are required to produce the ECG trace. Furthermore, a conducting gel has to be applied to the body to reduce resistance between skin and electrodes. This means that overall this method is less user friendly and requires other components whereas the PPG method does not need anything other than the sensor.

1.1 Types of Irregular Heart Rhythms

An arrhythmia is defined as a breakage of the rhythm (of the heart beat) either in timing or shape (of the ECG trace). If the shape of the ECG trace is different to a normal ECG trace, it means that the electrical activation didn't occur in the sequence it should have. The mechanical action will also be out of sequence thus the shape of the trace will be different. This is a cardiac arrhythmia.

One type of arrhythmia is called atrial fibrillation. This causes the heart rhythm to be irregular, in which case sometimes the electrical impulses are propagated to the ventricles and the ventricles beat. Other times the atrial contraction which are p-waves in the ECG trace, is not seen because there is a limited time in which the waves can be captured. Consequently, the beat to beat timing interval in atrial fibrillation is chaotic, with no regular pattern in timing.

Figure 1 - ECG Trace of Person with Atrial Fibrillation [1]

1.2 Heart Rate Variability

A healthy heart beat contains irregularities. Heart rate variability (HRV) means the variation in time between consecutive heartbeats. It is universally accepted as a non-invasive marker of autonomic nervous system (ANS) activity. Variability is different to the heart rate, as it increases when doing relaxing activities and decreases when in stress. Thus, HRV is generally greater when the heart rate is low and smaller when the heart rate is high. The relationship between HRV and heart rate is inversely proportional [4]. Variations are detected in an ECG trace by measuring the intervals between heartbeats are called R-R intervals. The R peak is the large spike at the beginning of a QRS complex as shown below in Figure 2:

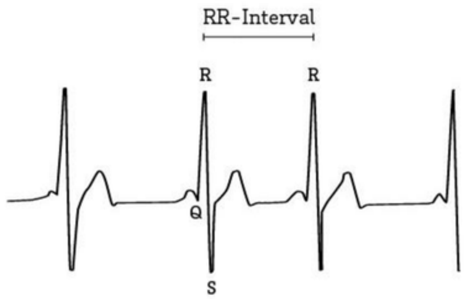

Figure 2 - QRS complex of ECG Trace [5]

To calculate the amount of variation in the intervals between heartbeats rMSSD (Root Mean Square of the Successive Differences) ~~can be used which~~ is the most commonly used HRV formula [5]. Patients with a history of atrial fibrillation record episodes HRV is significantly reduced under rMSSD (<30ms) [7].

New studies also showed that low-resting short-term heart rate variability was associated with elevated incidence of atrial fibrillation. Higher risk ~~for higher risk for~~ atrial fibrillation was linked to lower overall heart rate variability. A baseline heart rate of < 60 beats per minute was associated with an increased risk for atrial fibrillation [6]. Thus, these measures could be used to find arrhythmias.

1.3 Importance of Filtering Noise in Signals

Filtering is a process that removes unwanted components from a signal. This is especially important in these types of applications where any noise or interference could lead to incorrect peak detections and thus can influence when measuring the heart rate. Different types of filters could be used for heart rate processing ~~however,~~ the most common filter used is the Butterworth filter as it has ~~maximum flat frequency response.~~ Furthermore, low pass or bandpass filters would be best for this type of application as for an ECG trace values beyond 100Hz will probably be unnecessary for detecting arrhythmias. In a low pass filter, high frequencies are attenuated. For a band pass filter on the other hand, frequencies in a specific frequency band are passed. The cut-off frequency is also an important factor for filtering it is the frequency beyond signals will not pass through it [8].

2 Method and Decisions

After the components had arrived, the first thing done was setting up the Raspberry Pi. The operating system (Rasbian) was downloaded onto the SD card and installed onto the Pi as well as other software which would be required for the project, such as the Arduino IDE etc. Next, the Arduino was set up and the photoplethysmographic (PPG) pulse sensor was connected. This sensor has to either be clipped to the ear lobe or wrapped around a finger using the Velcro strap provided.

The pulse sensor uses photoplethysmography to measure the heart rate and this is displayed on the Arduino serial monitor using some code which is provided in an Arduino library specifically for the pulse sensor called PulseSensorPlayground. This library contains basic programs which can plot the heart rate trace and display the heart rate.

Photoplethysmography (PPG) works through the use of low-intensity infrared green light at high precision. Various tissues in the body absorb light when light penetrates through the tissues. It is more strongly absorbed by the blood compared to other tissues such as bone etc. This variation in absorption means that PPG sensors are able to detect changes in blood flow, by using the changes in light intensity. The amount of blood flow through the blood vessels is proportional to the PPG voltage signal. PPG can even be used to detect small changes in the volume of blood, this means it is able to provide a higher resolution (for more information please check [2]).

Purely for testing the functionality and quality of the pulse sensor, the provided library code was used to display the heart rate trace on the Arduino serial monitor. The sensor was quite sensitive to movement which could be seen from the trace, as trace was a bit erratic when the heart rate was measured while moving the finger. Furthermore, there was some noise visible in the signal, shown in Figure 3, in the results section below. Thus, it was decided to filter the signal before doing using the data to find any irregularities in the heart rhythm.

Before filtering, the original data was written to a text file using some Python code shown in Figure 6. This was done so that testing for heart rhythm problems could be done more conveniently as this file could be used as required instead of having to acquire new data when it was needed. This code wrote values to a text file sent from the Arduino (which was reading the sensor data); the Arduino was sampling the sensor values at a rate of 100 Hz. To keep the sampling frequency constant, timer interrupt coding was used on the Arduino, the code is shown in Figure 5.

The Python code in Figure 6, listens to the port that the Arduino is sending information to and it writes each value read by the Arduino to the text file; each value is written on a new line until the program is terminated or there is an error. The Arduino timer interrupt code in Figure 5, basically sets up the timer, the 64-bit prescaler and finally the match compare register so that every 100 Hz there is an interrupt. The baud rate for this is set to the maximum value of 115200 to ensure that serial communication is faster than the data sampling rate. The code reads the value of the pulse sensor and assigns the value to a variable, this occurs 100 times each second.

Once the data was written to the text file, it was then pre-processed by filtering. First, the text file including the data was converted into an array in Python, so that filtering could be done using various Python libraries such as SciPy and MatPlotLib was used for plotting the signal. A Butterworth filter was used after researching through many different filters. This was mainly done to remove noise and eventually to do a Fourier transform to separate the heart beat frequencies. This would allow for detecting peaks and eventually the heart rate. A low pass Butterworth filter (response is maximally flat which means there are no ripples) was used to remove noise and to smoothen the signal. The higher the order of the filter the closer it is to an ideal filter that has unity gain in the passband [3]. A sixth order filter was used, and the cut-off frequency was set to 0.5 (as this is a digital filter ω_n had to be between 0 and 1). These values reduced the noise and smoothen the signal to the degree required for Fourier transforms as shown in Figure 4 and code is shown in Figure 7.

Eventually, a Fourier transform was done however the results were not satisfactory. There was a large peak at 0 and no other prominent frequencies could be seen from the transform. So that method was abandoned. The Fourier transform didn't work because an ECG signal is a periodic signal however it is not a signal that can be reconstructed via sine and cosine waves. Hence, that method was unsuccessful for splitting the waveform into different frequencies.

As an alternative, peaks were identified to work out the heart rate and this was done by coding a different function shown in Figure 8. This code takes the sampling frequency, the array of the filtered signal and a threshold value to above which peaks will be identified. E.g. the threshold value could be 530, this is dependent on the values captured in the filtered data from the sensor and the threshold should be a suitable value. This value can be chosen by plotting the signal and identifying above which value do most peaks occur. The function itself looks at each value in the filtered signal individually and checks if the current value is greater than the previous value, the next value and also greater than the threshold. If it is true, then a peak is identified, and it is added to a variable called beat_count. A few calculations are done to eventually get the beats per minute and it is rounded to return a whole number from the function.

[illegible]ntly, a toolkit for processing heart related data was found and some initial processing was done using the functions in that toolkit, purely for testing purposes and to find out if the functions in that toolkit could assist in the project. The library was called HeartPy was installed. The code used some measures that related to heart rate variability such as rMSSD, and also used a different peak detection and rejection mechanism to calculate the heart rate. To detect peaks, it used a fitting function in which the standard deviation between successive differences was reduced. This method looks for the optimal threshold of the data by looking at the regularity of the heart rate. Some incorrect peaks which were detected may still remain due to various factors, these peaks were rejected based on a thresholded value for the RR-intervals.

The code in Figure 9, takes a set of data and it processes the data and returns a measures dictionary containing results. So, the function named process, in the code, takes the following arguments: an array of data, the frequency at which the data is sampled, and it also shows the time for processing the data. The next part prints the values of some values of the measures dictionary such as heart rate in beats per minute, sdsd (standard deviation between successive differences) and rMSSD. ~~However, nothing more was done with this information as the semester had finished.~~

[illegible]s noticed that the previous function in Figure 8 gave a heart rate of 66 beats per minute and the value of the bpm from the measures dictionary was 68 beats per minute (Figure 10), which showed that the temporary function is promising and can be further developed into something similar to the HeartPy function which uses a fitting function to choose the best threshold.

3 Results

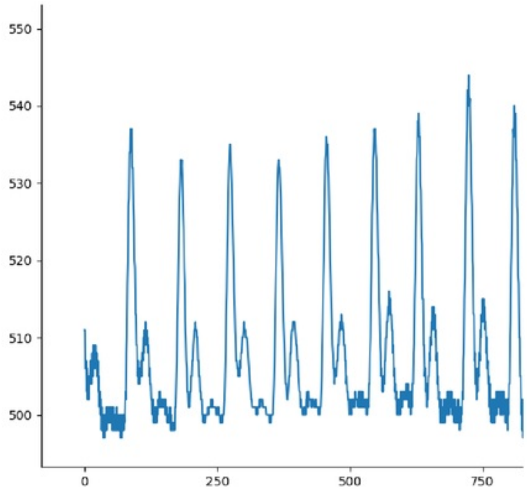

Figure 3 - Original signal showing noise

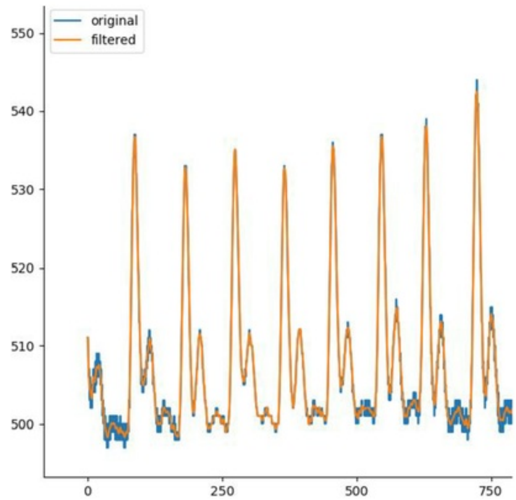

Figure 4 - Filtered and unfiltered signal comparison

```
//  Variables
int PulseSensorPurplePin = 0;  // Pulse Sensor PURPLE WIRE connected to ANALOG PIN 0

int Signal;                // holds the incoming raw data.
                          //Signal value can range from 0-1024

// The SetUp Function:
void setup()
{
  Serial.begin(115200);         // Set's up Serial Communication at certain speed faster
                                //than the rate at which data is being sampled.

  cli();//stop interrupts

  //set timer1 interrupt at 100Hz
  TCCR1A = 0;// set entire TCCR2A register to 0
  TCCR1B = 0;// same for TCCR2B
  TCNT1  = 0;//initialize counter value to 0
  // set compare match register for 100Hz increments
  OCR1A = 2499;// = (16*10^6) / (100*64) - 1 (must be <65536)
  // turn on CTC mode
  TCCR1B |= (1 << WGM12);
  // Set CS01 and CS00 bits for 64 prescaler
  TCCR1B |= (1 << CS11) | (1 << CS10);
  // enable timer compare interrupt
  TIMSK1 |= (1 << OCIE0A);

  sei();
}
// The Main Loop Function
void loop()
{
  //Do nothing!
}

ISR(TIMER1_COMPA_vect)
{
  Signal = analogRead(PulseSensorPurplePin);  // Read the PulseSensor's value.
                                              // Assign this value to the "Signal" variable.
  Serial.println(Signal);                    // Send the Signal value to Serial Plotter.
}
```

Figure 5 - Timer Interrupt Arduino Code

```python
#!/usr/bin/python3

import serial

port = "COM5"

with open("test.txt", "w") as outfile:
    try:
        with serial.Serial(port, 9600, timeout=1) as ser:
            n = 0
            while ser.is_open:
                # Convert to UTF-8 from byte string, and strip return char(s)
                line = ser.readline().decode('UTF-8').rstrip()
                outfile.write("{}\n".format(line))
                print(n, line)
                n = n + 1
    except serial.serialutil.SerialException as e:
        print("Serial communications error:")
        print(e)
```

Figure 6 - Python code for writing data sent from Arduino to text file

```python
def butterworth_filter(orig_sig):
    b, a = signal.butter(6, 0.5, 'low')
    output_signal = signal.filtfilt(b, a, orig_sig)
    return output_signal
```

Figure 7 – Python Butterworth Filter Code

```python
def bpm_from_peaks(freq, sig, threshold):
    beat_count = 0
    for i in range(1, len(sig) - 2):
        if sig[i] > sig[i - 1] and sig[i] > sig[i + 1] and sig[i] > threshold:
            beat_count = beat_count + 1
    fs = freq
    n = len(sig)
    duration_in_seconds = n/fs
    duration_in_minutes = duration_in_seconds/60
    bmp = beat_count/duration_in_minutes
    return round(bmp)
```

Figure 8 - Python Code that Measures the Heart Rate

```python
data = hp.get_data('test_new_filtered.csv') #data.csv is sampled at 100Hz
working_data, measures = hp.process(data, 100.0, report_time=True)
hp.plotter(working_data, measures)
print('bmp = ', measures['bpm'])
print('standard deviation = ', measures['sdsd'])
print('rms of sd = ', measures['rmssd'])
```

Figure 9 - HeartPy example code

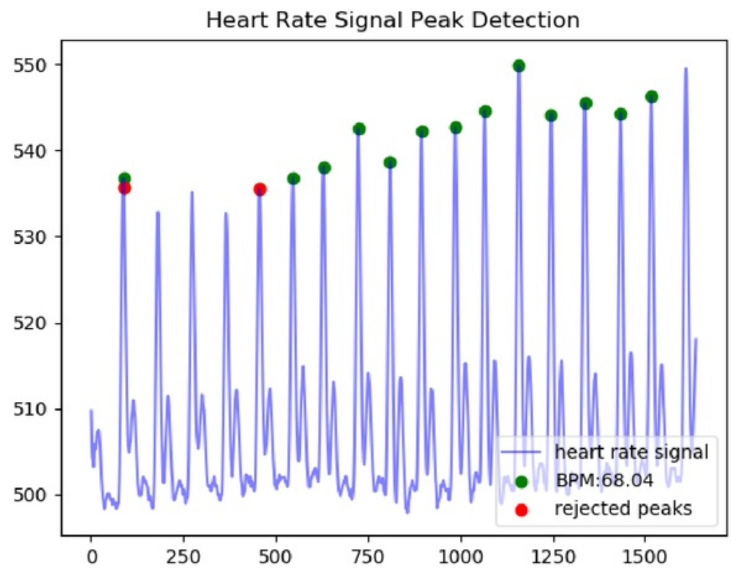

Figure 10 - HeartPy plot showing detected and rejected peaks and heart rate

4 Future Plans

Currently, what has been done hasn't totally been to plan according to the Gantt chart in the proposal. g up the Raspberry Pi and waiting for components to arrive etc, these things have led to being slightly behind schedule as well as some time was spent on Fourier transforms.

The original Gantt chart is shown below in Figure 11:

Gantt chart

Project Plan for Heart Rhythm Abnormality Monitor

#	Activity	Start	End
1	Learn Python Basics	11	13
2	Order Components	11	13
3	Arduino Basics & research Python libraries & SQL	11	14
4	Raspberry Pi Setup & Arduino Communication	13	14
5	Coding and getting the pulse sensor to work	14	17
6	Writing the interim report	15	21
7	Coding: Converting pulse sensor readings to graphs	17	19
8	Coding: Using data to find variance of heart rhythm	19	27
9	Coding: looking at ways to differentiate arrhythmias	28	31
10	Coding: arrhythmia statistic system	32	34
11	Final report	27	36

week > 11 12 13 14 15 16 17 18 19 20 21 27 28 29 30 31 32 33 34 35 36

Figure 11 – Initial Gantt Chart

Currently, everything has been done until activity 8. The method for looking at heart variance will most probably be root mean square of the standard deviation between two points as that is the most popular method, as found from research. This will probably have to be developed and researched further to make the method more reliable by finding what out what values of rMSSD prove that a person has arrhythmia or a risk to developing arrhythmia. It can be seen that there is evidence to suggest that low heart rate variability is linked to arrhythmia such as atrial fibrillation.

At this moment in time, activity 3 hasn't been needed because MySQL has not been incorporated into the project as of yet. It is still something to be considered, if the data from the pulse sensor was able to be logged to a database and the user would be able to track their heart rate, how frequent their episodes of arrhythmia were or if they have a specific time of the day when their arrhythmia increases etc. So that is something which may be added to the project after a better peak detection algorithm is made which is able to fit the optimal threshold without the user having to do it manually.

Furthermore, if these issues are resolved then the MySQL database could be connected to a server and users would be able to access their data online via an app or a website similar to the Fitbit. And they would be able to track their data anywhere and anytime.

So, the next steps are to: first resolve how to find if a person has arrhythmia maybe not only use rMSSD, there could be other measures that may help in proving someone has arrhythmia and also create a better peak detection algorithm.

References

[1] Healio.com. (2018). Atrial Fibrillation ECG Review - Criteria and Examples | LearntheHeart.com. [ONLINE] Available at: https://www.healio.com/cardiology/learn-the-heart/ecg-review/ecg-topic-reviews-and-criteria/atrial-fibrillation-review# [Accessed 8 Oct. 2018].

[2] Soulfit. 2018. How Does PPG Technology Works?. [ONLINE] Available at: https://soulfit.io/blog/how-does-ppg-technology-works/ [Accessed 7 January 2019].

[3] StackExchange. 2016. Higher Order Butterworth Filters. [ONLINE] Available at: https://dsp.stackexchange.com/questions/34127/higher-order-butterworth-filters. [Accessed 8 January 2019].

[4] Firstbeat. 2019. Heart Rate Variability. [ONLINE] Available at: https://www.firstbeat.com/en/science-and-physiology/heart-rate-variability/. [Accessed 8 January 2019].

[5] Ouraring. 2017. What Is Heart Rate Variability And What Can You Learn From It. [ONLINE] Available at: https://ouraring.com/heart-rate-variability-basics/. [Accessed 8 January 2019].

[6] Healio. 2017. Low heart rate variability linked to higher AF risk. [ONLINE] Available at: https://www.healio.com/cardiology/arrhythmia-disorders/news/online/%7B44ee2ed9-8cb2-4ca2-a8e5-1307e94e9d01%7D/low-heart-rate-variability-linked-to-higher-af-risk. [Accessed 8 January 2019].

[7] Barauskiene, V., Rumbinaite, E., Karuzas, A., Martinkute, E. and Puodziukynas, A. (2016). Importance of Heart Rate Variability in Patients with Atrial Fibrillation. Journal of Cardiology & Clinical Research, p.1.

[8] Wikipedia. 2018. Filter (signal processing). [ONLINE] Available at: https://en.wikipedia.org/wiki/Filter_(signal_processing). [Accessed 8 January 2019].

Student Self-reflection on performance

All students must complete the following sections for every piece of work they submit using this template. The aim of this is to help you use feedback more effectively to improve your marks and your skills as a professional engineer. This section is not formally marked, but your tutor may use it when discussing your work with you.

Describe how you have used AT LEAST ONE of the following sources of information to improve this piece of work: 1.) (PREFERRED) Feedback from previous assignment(s). This can be from the same module or from a previous module or previous year of study (e.g. comments from 1st year lab formal reports should be used to help improve your 2nd year lab formal reports). 2.) The marking criteria or rubric provided for this assignment. 3.) The Department Technical Writing Handbook for Students.
Feedback from first year lab reports and second year design reports have helped in writing this report. Furthermore, the technical writing handbook also supported in referencing.
Are there any aspects of this work that you would specifically like the marker to comment/or advise on? For example: "I wasn't sure if my figure formatting looked professional and would appreciate feedback on this aspect"
I wasn't sure if I have written enough about my project as a whole and if more background information was required, I would like some feedback on this. Thank you.

Interim Report

GRADEMARK REPORT

FINAL GRADE

15/20

GENERAL COMMENTS

Instructor

PAGE 1

Comment 1

General feedback:

This is well-written on the whole, you do need to provide simpler explanation of some concepts, and think about the order in which you are introducing concepts. There is slightly too much about the mechanics (e.g. the Python code) and not quite enough about the theory, but on the whole this is a very good report.

Comment 2

I would have liked to see the title of the project here too.

PAGE 2

PAGE 3

PAGE 4

Comment 3

Would have liked to see a diagram of a normal ECG trace here, and some explanation of how the different parts of it relate to physical activity in the heart (e.g. the big "R" peak is the ventricular contraction).

Comment 4

What's a p-wave? You haven't said.

Comment 5

What would an invasive marker be? No need for "non-invasive" unless you're going to explain it. I can see this phrase used in several places online - if it's something you have taken from elsewhere, give a citation (and if you're saying something is "generally accepted as ...", you definitely need to give me a reference to prove that it IS generally accepted).

Comment 6

This is what you have said in the previous two sentences too. All you need to do is say "The relationship between HRV and heart rate is inversely proportional [4]; HRV increases when the heart rate itself is low, and decreases when the heart rate is high."

Comment 7

Should have been earlier in the report!

Strikethrough.

Comment 9

I can't work out what this sentence is supposed to mean.

Comment 10

Higher risk of atrial fibrillation...

I think (from the short report on healio.com) the picture is probably more complex than this - without reading the two primary sources, there are other measures (e.g. frequency ratios) which indicate increased risk. There's also the point that you are trying to detect actual AF, not an increased risk of developing AF later in life.

Strikethrough.

Comment 12

reduce the accuracy of heart rate measurements.

Comment 13

Full stop, new sentence. I over-use "however" myself when writing, however it is not usually necessary.

Strikethrough.

Comment 15

Why is this a good thing? Is it a good thing?

Comment 16

What components? There needs to be a brief description here of how you decided which components to use, a diagram of how the system will fit together.

Comment 17

I have a bit of a bee in my bonnet about this sort of thing in reports (yours is not too bad, I have seen MSc reports with several pages devoted to downloading and installing Raspbian, something which any competent person can do in a few minutes). All you need to say is that for processing data you are going to use a Raspberry Pi Model 2B+, with the Raspbian operating system installed. Data collection will use an Arduino Mini (or Micro, I forget which). Explain why you're using the Arduino (built-in A/D).

Comment 18

Say exactly which pulse sensor you are using.

Comment 19

The baud rate is for serial communication, not for the interrupt. This section isn't particularly clear - you are using the timer interrupt on the Arduino to ensure samples are recorded at precise intervals. The data is then sent by serial communication to the Raspberry Pi, which is running a Python program to store the data in a file.

Comment 20

Can a Butterworth filter do Fourier transforms?

Comment 21

Ripples in what? I think you need a simple description of what a Butterworth filter does.

Comment 22

Figures should be close to the text describing them. I shouldn't have to scroll up and down to find the figures you are describing.

Comment 23

It was never going to work. Sometimes, you do things (on bad advice from a project supervisor for example) which don't work. You don't need to mention them in the report.

Comment 24

This is essentially using code from Paul van Gent, isn't it? I think you need to describe how the algorithm works, I don't need to see the short bits of code where you are using the function.

Comment 25

You need to reference this - it's important.

Comment 26

You need a better description of how this algorithm works.

Strikethrough.

Comment 28

I'd have liked to see more description of the different algorithms used for peak detection and why you think they give different results.

Comment 29

As previously noted, all these figures should be in the previous section.

Comment 30

Not necessary to include all this. There should be a reference to the source of the information about how to use timer interrupts on the Arduino.

Comment 31

This code is trivial, and I think I wrote all of it! No need to include it.

Comment 32

This doesn't tell me anything about how a Butterworth filter works, or how the Python code to implement the filter works. You don't need to include code in the report - the algorithms used are much more important.

PAGE 10

Comment 33

This is a little too informal.

Comment 34

I think you also need to mention the Physionet database which you are eventually going to have to use as a source of abnormal heart rate data.

PAGE 11

Comment 35

This is a short online report on two papers in J. Am. Coll. Cardiol. - you need to cite those two primary sources, not the secondary source.

PAGE 12

PAGE 13
