A Study of PPG principals and use of a camera to capture reflected light for heart rate detection

Submitted by: Sara Kinghan

ID: 17964398

Supervisor: Dr Jeff Kilby

School of Engineering, Computer and Mathematical Science

A project report presented to the Auckland University of Technology in partial fulfilment of the requirement of the degree of Bachelor of Engineering

2020

Statement of Originality

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another (except where explicitly referenced and acknowledged), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Date: Friday 30th October

Name: Sara Kinghan

Acknowledgements

I would like to thank my supervisor Dr Jeff Kilby for proposing this project. I am grateful to have had the opportunity to explore this topic.

I would also like to express gratitude to my colleague who completed this project with me, Jasmine Schwieters.

Abstract

Non-contact methods of determining pulse rate are of interest for medical use. We used Eulerian video magnification method to reveal the colour change in the face due to blood flow from heartbeats. Then we explored different methods to calculate a pulse rate using the colour change as the signal. The two group members were the subjects for the videos which were used. A pulse oximeter was used to obtain a ground truth pulse rate for the videos. We achieved a method of extracting the green colour information from a video and processing it into a signal from which a pulse rate could be calculated. Three different methods of calculations were used. In addition, we displayed heart rate variability which is also of interest in medical and scientific areas. Our results show that it is possible to get a basic bpm reading using a camera and MATLAB code. This project was very much first steps into a large subject area leaving many options open of further work and development.

Acronyms

EVM	Eulerian video magnification
rPPG	Remote photoplethysmography
PPG	Photoplethysmography
iPPG	Imaging photoplethysmography
HR	Heart rate
PR	Pulse rate
EKG	Electrocardiography
BPM	Beats per minute
HRV	Heart rate variability
ANS	Autonomic nervous system
PG	Plethysmography

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background

Wu et al. wrote a paper called “*Eulerian Video Magnification for Revealing Subtle Changes in the World*”. The focus of that project was to reveal the variations in videos that are impossible to see with the human eye. The method they developed is known as Eulerian video magnification (EVM). It reads a video and applies spatial decomposition and temporal filtering to every pixel in each frame. Then the result is amplified so that the human eye can see the colour changes that are occurring. (Wu et al., 2012) The goal of their paper was to show that spatial and temporal processes could be used on videos to reveal the changes that are occurring in the world but are not possible to see with the naked human eye. One such change is the blood that flows through the body at each heartbeat changing the colour that is reflected from the face. From this change in colour, a pulse rate can then be calculated.

1.2 Project Aim

The paper by Wu et al. was the inspiration and starting point for this project. Their project revealed the change in colour reflected from the face from each pump of blood from the heartbeat. They achieved this using MATLAB (The Mathworks, Natick, USA).

The members of this project group were to attempt to achieve this and then if a signal from the change in colour could be generated, to use it to determine a pulse rate.

The overall goal of this project was to use the colour changes in a face to determine a pulse rate using MATLAB (The Mathworks, Natick, USA, version R2020b). This meant the project had two main aims. The first was to detect and reveal the colour changes in the face in order to obtain the required signal. To do this, the code from the project by Wu et al. could be used. They had made the source code publicly available. It was downloaded from (*Eulerian Video Magnification*, n.d.). Although it was not required that the method of EVM be used. Next was to obtain the colour change as a signal from which a pulse rate could be calculated. The end goal was to produce a pulse rate from a video.

1.3 Introduction

The applications of a project in the area of remote photoplethysmography (rPPG) can be useful in medical and scientific fields. It does not require the user wear any sensors in order for information to be obtained.(Zhang et al., 2020) Current pulse measuring devices can cause damage to the skin of premature new-borns and the elderly.(Bush & Mall, n.d.) Non-contact methods of measuring the pulse rate would also be of use for telemedicine and for the average person with underlying medical conditions that may wish to monitor their heart rate without expensive, large or invasive equipment.

For the group members of this project this was a completely new area, from using more capabilities of MATLAB, to processing images and signals.

Chapter 2 Literature Review

Determining a heart rate from detecting the changes in blood flow is known as photoplethysmography (PPG). It is the method that imaging photoplethysmography (iPPG) and remote photoplethysmography (rPPG) are based on. Therefore, a review of the developments and concepts of PPG would aid in the understanding of the principles of iPPG and of what this project was to achieve. (Sun & Thakor, 2016)

2.1 Heart rate

Heart rate (HR) is an important parameter in a clinical environment for assessing the function of the heart. It is useful as an indicator of health status and in the diagnostics and assessment of cardiovascular diseases and chronic diseases. (Zhang et al., 2020) The pulse rate (PR) is typically measured using electrocardiography (EKG). The EKG detects electrical activity of the heart. (Miljković & Trifunović, 2014)

The pulse rate is the frequency at which the heart beats or contracts. It is measured in number of beats that occur per minute (bpm). The pulse rate changes depending on the body's need for oxygen. Many factors can influence this including but not limited to physical activity, emotion, illness, stress, and drugs. (Feukeu & Winberg, 2019) The Mayo clinic states that a normal resting heart rate for adults falls within a range of 60 to 100 beats per minute. (*2 Easy, Accurate Ways to Measure Your Heart Rate*, n.d.)

2.1.1 Heart rate variability

When two beats have occurred the first bpm can be calculated from the time elapsed between them. When the pulse rate is plotted each time a new bpm is calculated, this is known as heart rate variability (HRV) since it shows the variations of the pulse rate during the cardiac cycle.(Zhang et al., 2020) HRV has a major part in the assessment of the neural control mechanism of heart rate. (Miljković & Trifunović, 2014) The electrical and contractile activity of the heart is mostly modulated by the autonomic nervous system (ANS). This regulation causes variability in the pulse rate when active and at rest. The variability is expected to be high when the subject is in a normal physiological state. Displaying heart rate variability provides insights into the connect between the sympathetic and parasympathetic nervous systems which aids in assessment of the autonomic nervous system. (Kranjec et al., 2014)

The variation between each heartbeat is not commonly shown. Using a heart monitor such as a pulse oximeter or a fitness watch, it is the wearers average pulse rate that is shown over a pre-set time period, such as 8 or 15 seconds. The variability of the heart between each beat it makes is not shown on these devices and likely only possible on an EKG machine. However, medically and scientifically, it is not just the average bpm that is useful information to have for assessment, but also the hearts variability.

2.2 Plethysmography

Plethysmography (from plethysmos meaning 'increase' in Greek) is the detection of the cardio-vascular pulse wave that travels through the body. Plethysmography (PG) methods are used to record the changes in the volume of blood in a tissue area that result from the pulse wave.(Goudarzi et al., 2020) It is this change in volume that is detected and used to determine a pulse rate. In PG the wave is found using the variations in impedance, strain or air pressure.(Verkruysse et al., 2008) When light is used the method is known as photoplethysmography. (Goudarzi et al., 2020)

2.3 Photoplethysmography

In 1938, Hertzman was the first to introduce the term photoplethysmography as a description of a non-invasive optical technique for detecting the changes of the volume of blood in blood vessels. (Kamshilin & Margaryants, 2017) Specifically it measures the light that is reflected from or transmitted through the body (McDuff et al., 2015) by using the principal that oxygenated haemoglobin absorbs more light than its surrounding tissue. (Verkruysse et al., 2008) Therefore, there is more light absorbed when blood is being pumped to the body than when the blood volume has decreased. (Sun & Thakor, 2016) More about wavelengths of light and their absorption into blood and reflection from tissue will be provided in its own section. It is from these concepts that the pulse oximeter was created.

2.4 Pulse Oximeter

PPG became a popular method for measuring the heart rate. It was less invasive, less expensive and less complex. (Feukeu & Winberg, 2019)

Pulse oximeters first became available commercially in 1983. (Kamshilin & Margaryants, 2017) They usually contain two lights, one red and another near infrared.   A pulse oximeter is designed to fit over a finger or toe, or in some cases an ear and send wavelengths of red and near infrared through the body.   A photodetector on the other side measures the wavelength and the interaction between blood and colour is used to determine a pulse rate. (Kamshilin & Margaryants, 2017) This is known as the transmitting mode.   Less light reaches the photodetector when blood volume increases because it is absorbed by the blood. Thus, an inverse relationship. When the lest amount light is detected, that is the heartbeat.

The wavelength of red light is around 650nm and it can penetrate approximately 4-5mm beneath the skins surface. The wavelength of near infrared light is around 750nm and it can penetrate approximately just over 5mm into the skin.   Due to the depth at which these two lights can penetrate tissue they are able to reach the blood that flows in the arteries of the body. (Rouast et al., 2018)

The effect of the wavelength will be expanded on in the next section. However, it is important to first explain the difference between the two modes of PPG systems, transmitting and reflecting.   This project will use footage from a video, therefore the reflected mode it the one that will be used.   It differs because it is the amount of light being reflected that is captured compared to the transmitting mode where it is the amount of light that is missing due to absorption.

2.5 Light waves and skin interaction

PPG is typically performed using red and/or near infrared light sources. This is due to the history of PPG and the principal that blood absorbs more light that its surrounding tissue.(Verkruysse et al., 2008)

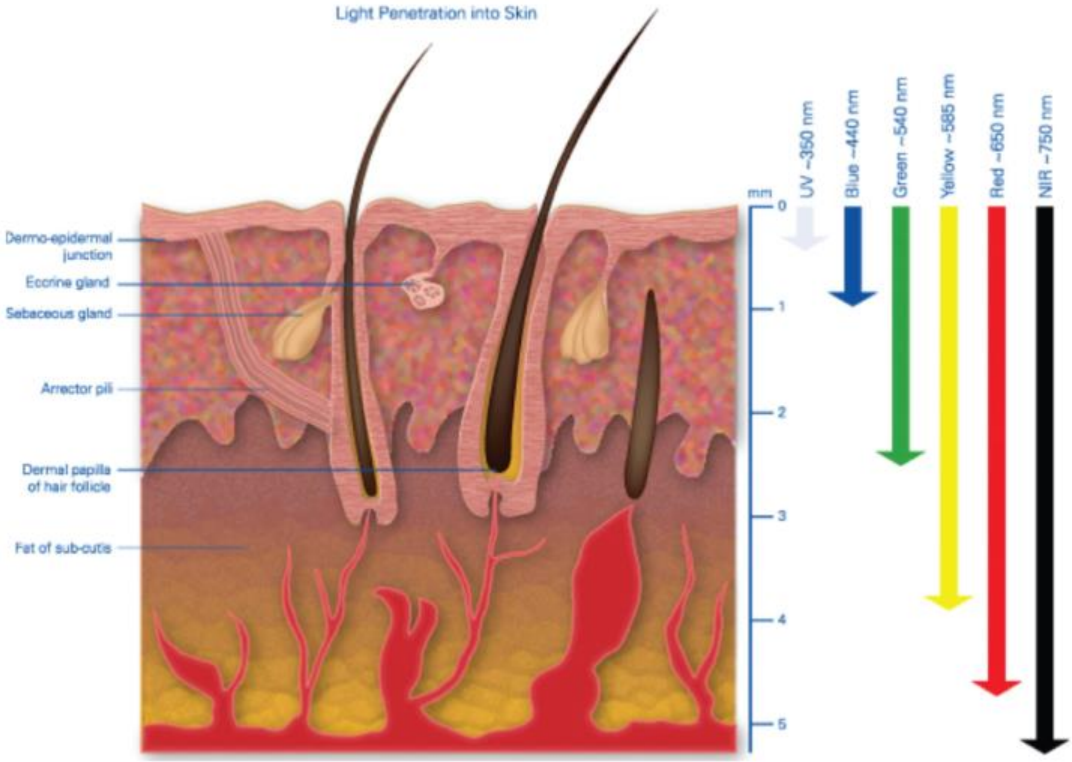

Figure 1: Image to show the depth of penetration of different wavelengths of colour. (Ash et al., 2017)

The wavelengths of red and near infrared can penetrate to the depth of the arteries which carry the blood as shown in Figure 1.

Verkruysse et al. study "*Remote plethysmographic imaging using ambient light*" was the first to use ambient light as the signal from which a pulse rate was obtained. Meaning they did not force the wavelengths of certain colours into the skin. This differs from the Wu et al. project as theirs was to use image processing methods to reveal changes in the world that our eyes cannot detect, and they don't calculate a heart rate. Verkruysse et al.'s project recorded video with a standard digital camera and examined each of the three colour channels, red, green and blue. Their conclusion was that green light is absorbed better in hemoglobin than red and green can travel deeper than blue light. They also found the red and blue channels to contain more noise. (Verkruysse et al., 2008) This prompted several other studies which explored this. The general finding was that the green colour channel gave the strongest PG signal.

Kamshilin & Margaryants in their paper "*Origin of photoplethysmographic waveform at green light*" challenged this on the basis that green light cannot possibly be being absorbed by the blood as it does not penetrate that deep. They claimed that green light could not be modulated by blood in the arteries due to its small penetration range. (Kamshilin & Margaryants, 2017) They explored what could be the reason behind other studies successes in using green light.

When blood is pumped around the body it causes the artery to expand as the blood flows through it. The expansion of the artery compresses the distance between the capillaries. (Kamshilin & Margaryants, 2017) In Figure 2, the image on the left shows the artery not yet with blood pumped through. The capillaries are situated as normal and green light from ambient light is entering the skin. Most of the green light will travel into the body and be absorbed. In Figure 2 on the right, the body has expanded the artery and closed the gaps between the capillaries. This prevents the green light from penetrating deeper and reflects it back. (Kamshilin & Margaryants, 2017)

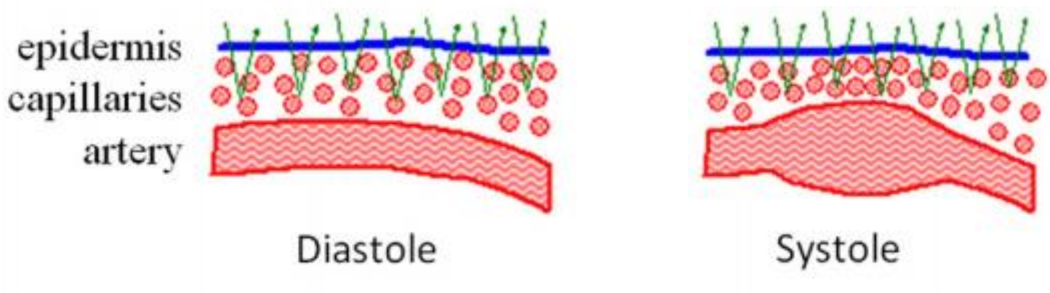

Figure 2: Image to show the movement of capillaries due to blood flow. It is this movement that affects how green light interacts with the tissue. (Kamshilin & Margaryants, 2017)

2.6 rPPG and iPPG

Imaging photoplethysmography (iPPG) and remote photoplethysmography (rPPG) are an extension of PPG methods. However, they are non-contact methods of measuring a pulse rate. Both terms are used interchangeably although iPPG is the use of images to extract a pulse rate signal whereas rPPG might not use images. As research and development in this area grows, more definition of the terms may occur. Verkruysse et al.'s study is referred to as the first case of iPPG. Though they never used that term in the paper. Wu et al.'s paper is not rPPG or iPPG as it does not determine the heart rate, only reveals colour change.

Chapter 3 Methods and Underlying theory

When the heart beats it sends blood around the body. The increase in blood flow from the heartbeat causes the capillaries closer together. The decrease in distance between capillaries means less colour makes its way through, and more is reflected back into the camera and recorded. The literature review has provided some background information into this.

The inspiration for this project was a study done by Wu et al. that made visible to human eyes the colour change that was occurring. In that project, image processing techniques were used to detect and reveal the colour change. However, their project only used the image processing to reveal the colour change and does not calculate a pulse rate from that.

Therefore, the goal of this project, was to use their code to detect the colour change and then find a way to use that signal to determine the pulse rate. This involved a few steps.

First, videos were needed in order to test the code. Second, to detect the colour change that occurs in the face and amplify it so it can be seen by human eyes. Third, use the occurrence of the colour change to calculate a pulse rate. Fourth, it was an aim to present the code in a user-friendly way.

3.1 Scope of project

From the literature it was clear that this is large area and one of much interest and research. For this project it was important to keep the focus on the goals and not be drawn into the many possible avenues that study in this area could take. In order to do this some parameters were decided on to minimise the scope of the project. Briefly, they were to focus on, obtaining the signal of colour change. Once obtained, getting one method of pulse rate calculation working. Limiting the heart rate to a resting heart rate only. And not aiming to compensate for any factors that cause noise or variance in the signal from within the code.

Amplifying the signal and revealing the colour change was not the aim. The signal was needed to determine a heart rate, but visually showing it in a video format was not the goal. There are a few ways to calculate a pulse rate which will be discussed later. The goal was to get one working. At a resting heart rate, the subject is calm, still as possible and breathing normally. Under an active heart rate, the subject will be breathing heavily, causing increased motion and possibly naturally redden the face. This introduces additional barriers therefore only a resting heart rate will be attempted.

Since video footage would be used there are factors that affect that signal. Capturing 30 frames per second (fps) means the camera is more sensitive to changes in the environment than human eyes are. The motion of the head, colours and lights in the background, interaction of lightwaves with different skin pigmentation and lighting on the face are just a few of the influencing factors.

Face detection was considered as a method of analysing only the region of interest. This would minimise the influence of any background colours and lighting.

None of the other factors would be compensated or adjusted for in the code. Therefore prior to recording the videos, parameters were determined to eliminate and reduce the effect of some factors. How this was considered is detailed below.

3.2 Video capture

Source videos were needed in order to test the code that was to be made. The two members of the project were the only subjects used. Both subjects are female between the ages of 20-35 with fair skin of a similar pigmentation. No adjustments in the code were made to compensate for skin pigmentation.

The camera used was a mirror-less DSLR, Olympus Pen Lite E-PL5. It has 16 megapixels and sensor type CMOS. It was fitted with a lens, Olympus M Zuiko Digital 14-150mm F4-5.6.

It was mounted on a tripod to eliminate handheld motion from video capture. All the videos were recorded at 30 fps and have a resolution of 1920x1080. The camera was positioned the required distance to have subjects face in the frame. This distance was about 1.5 meters but was not measured.

Filming took place in a large classroom. Natural light and artificial light were throughout the room. The subjects were not under any direct or forced illumination. Subjects sat on the floor with their back straight against a dark coloured wall. The dark colour was to provide a clear point of difference in colour between the area that was the subject's face and the background.

For each of the two subjects, 1 set of videos were to be taken of 4 videos each. They were allowed time to settle and reach rest so that a resting heart rate could be captured.

The videos were taken one after the other. Each video was more than 15 seconds but less than 18 seconds. During filming they were asked to remain as still as possible but to breath normally. It was decided to use at least 15 seconds of footage since a heart rate is measured in beats per minute. 15 seconds would contain a quarter of that time frame.

While videos were recording, the subjects wore the Rossmax SB100 Finger Pulse Oximeter. It was worn on the first finger of their dominant hand. This was the right hand for both subjects. Its values were to be the ground truth heart rate for this project. This pulse oximeter has a measurement deviation within $\pm 3\%$.

For each video segment, the values from the pulse oximeter were recorded by the other member. This was done by writing down the values that appeared on the pulse oximeter over the time frame that the camera was recording. The highest and lowest values were used to obtain a mean heart rate for the 15 second period over which the video footage is captured. This value was used to compare the results of the project code to.

However, this pulse oximeter measures a mean HR over a period of 8 seconds. The videos were recorded for at least 15 seconds and the intension was to analyse the signal over the entire time frame of the video. The amount of information gathered over a 15 seconds period will be greater than over 8 seconds and may result in variance. However, it was decided that the more information available for the code to calculate a heart rate would be a better start point. Implications of this decision will be discussed in the results.

3.3 Revealing colour change

The project by Wu et al. made the code available for public non-commercial use at (*Eulerian Video Magnification*, n.d.) For this project, this code was the start point.

The process of EVM was not discussed in the literature review and will be briefly explained here.

3.3.1 Eulerian video magnification method

First the video is decomposed into different spatial frequency bands.   This is done through use of a Gaussian pyramid.

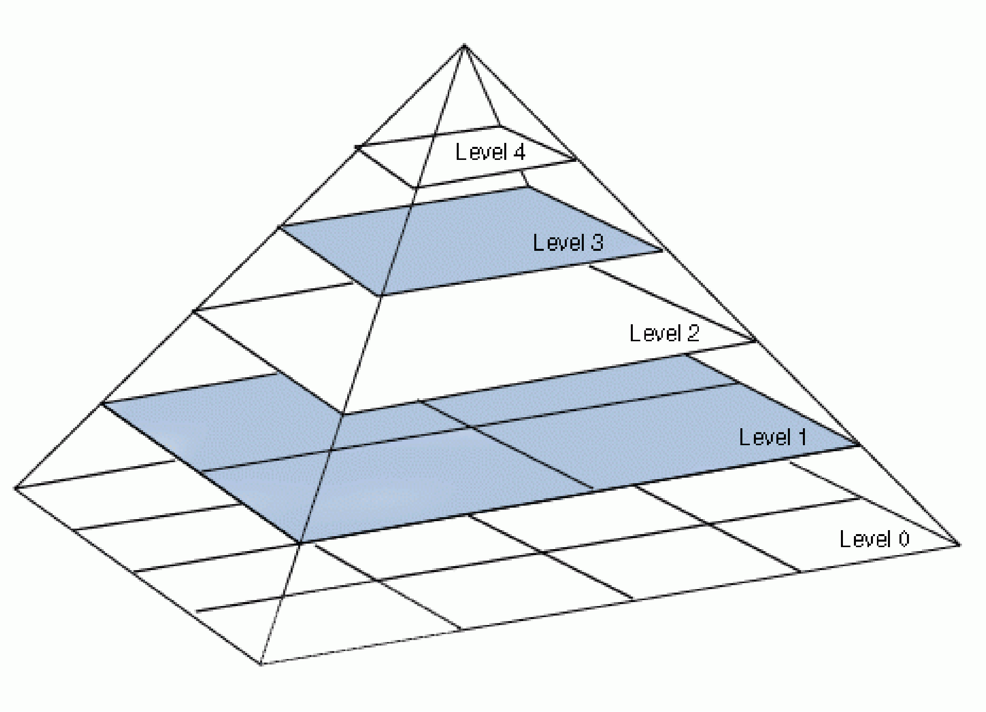

Figure 3: Gaussian pyramid levels from level 0 to 4. ('Gaussian and Laplacian Pyramids', 2020)

An image is used as an input and a gaussian filter is applied. This produces level 0 at the same size as the original image. However, with each level of the pyramid after the gaussian filter is applied then the resolution is down sampled by 2.('Gaussian and Laplacian Pyramids', 2020)   Thus, the pyramid structure as shown in Figure3.   If the starting resolution is 1920x1080 at level 0, a typical video recording resolution, at level 1 the image size will be 960x540. The filter is a blur effect in image processing.

Figure 4: Example images of the effect of using a Gaussian pyramid. Showing original to level 2. ('Gaussian and Laplacian Pyramids', 2020)

Figure 4 shows the effect the gaussian pyramid has on an image from its original capture to level two of the pyramid. At each level, the number of pixels has halved.

Essentially the original image is reduced or decomposed by a set number of levels. At each level, the image is blurred and then the resolution is halved. Visually, it looks out of focus or of poor quality. But this is a smoothing process for the information held in the image and the first stage in Eulerian video magnification. (Alcala et al., 2016)

Then temporal filtering is applied. This is done by applying a band pass filter of a set range to every pixel in each frame. Wu et al. used ideal bandpass filters for colour amplification, stating that the sharper cut-off frequencies from bandpass filters were preferable. The frequencies of interest might be 0.66 - 4Hz. This is a heart rate range of 40 to 240 bpm. After filtering, the signal is then multiplied by an amplification factor.

After this point, the resulting array is formatted into a video. It can be made into a video on its own, showing just the signal obtained, or it can be overlayed on the original so that it can be viewed as if those colour changes are normally visible to the human eye.

Colour spaces:

The project by Wu et al. uses NTSC. A typical camera will capture in the RGB colour space. Wu et al.'s code converts RGB to NTSC. NTSC stands for the National Television Systems Committee and it defines a colour space that is known as YIQ. This colour space separates the information from the image into two parts, colour and grayscale. This means that this colour space can be used to transmit colour and greyscale information. Making it suitable for displaying images on colour and black and white televisions. Y is Luma, or the brightness as is a value between 0 and 1. I is the amount of blue or orange tone in the image. Q is the amount of purple or green tones in the image. (*Understanding Color Spaces and Color Space Conversion* - *MATLAB & Simulink* - *MathWorks Australia*, n.d.) The project by Wu et al. was the only project found that uses this colour space.

Expected result:

First, the results from that project were looked at. Figure 5 shows a picture they provide of the output obtained using the code they have made open to the public.

The top four images are four frames from the original video to show that colour change is not visible. The bottom four images are the same four frames of the original video with the amplified pulse signal overlaid over the video.

It does not state it in the paper by Wu et al. however these frames likely are not immediately sequential. It is more likely that two frames that show the greatest amount and lest amount of colour were chosen to illustrate the method.

Figure 5: Results from the EVM method shown by Wu et al. (Wu et al., 2012)

3.3.2 Attempts

Attempt 1:

It was the goal of this project to use their code to obtain the signal of the change in colour and then attempt to calculate a heart rate from that. Therefore, the first attempt was to run the code from Wu et al. as it was obtained on the videos that had been recorded for this project.

The code has a few options for users to alter. Those are the level of pyramid, amount of magnification, and the frequency range for the band pass filter. Combinations of the following settings were tried. Pyramid level of 4 and 6, magnification of 50 and 100, frequency range of 0.8 to 1.5 and 0.8 to 3. Other than varying those user options, none of the code was altered.

Attempt 2:

A video is captured in the RGB colour space. The camera sensor picks up the values of R (red), G (green), and B (blue). In this attempt, the video colour space was not converted, but remained in the RGB colour space and the signal was to be made into its own video, not overlaid on the original video. This was so that signal could be more clearly observed.

The code below is the unaltered code from Wu et al. and is the part of the code where the colour space conversion occurs.

```
temp.cdata = read(vid, startIndex);
[rgbframe, ~] = frame2im(temp);
rgbframe = im2double(rgbframe);
frame = rgb2ntsc(rgbframe);
```

On the fourth line, it converts RGB to NTSC. Instead of converting the frame from the RGB colour space into the NTSC colour space, it was left as an RGB image. How the code was altered to achieve this is shown below.

```
temp.cdata = read(vid, i);
[rgbframe,~] = frame2im(temp);
frame = im2double(rgbframe);
```

Also, the code for amplifying the signal had to been altered. The original code using NTSC with a colour space of YIQ was amplified the following way. Each part of the YIQ colour space was amplified.

```
filtered_stack(:,:,:,1) = filtered_stack(:,:,:,1) .* alpha;
filtered_stack(:,:,:,2) = filtered_stack(:,:,:,2) .* alpha .* chromAttenuation;
filtered_stack(:,:,:,3) = filtered_stack(:,:,:,3) .* alpha .* chromAttenuation;
```

However, using RGB it was the intension to amplify only the green part. When trying to isolate one colour from RGB, MATLAB numbers them 1, 2, and 3 respectively. The code below amplifies only the green colour and leaves the red and blue unchanged in the array.

```
filtered_stack(:,:,:,2) = filtered_stack(:,:,:,2) .* alpha;
```

The code to create the output video combines the original video and the signal. At this stage in testing, it was just the signal that was wanted. The code from Wu et al. shown below is how the original video and the signal are combined. The original video is read, and each frame is converted from RGB to NTSC. Then the filtered stack which contains the signal has the pyramid collapsed and is added frame by frame to the original video. Then the combination of them is converted back into RGB in order to be formatted into a video.

```
temp.cdata = read(vid, i);
[rgbframe,~] = frame2im(temp);
rgbframe = im2double(rgbframe);
frame = rgb2ntsc(rgbframe);

filtered = squeeze(filtered_stack(k,:,:,:));
filtered = imresize(filtered,[vidHeight vidWidth]);
alter = filtered+frame;
frame = ntsc2rgb(alter);
```

This was not what was wanted. The code was altered to just run the signal into a video format. All of the above code was replaced with the code below.

```
 filtered = squeeze(filtered_stack(k,:,:,:));
 filtered = imresize(filtered,[vidHeight vidWidth]);

open(vidOut)
writeVideo(vidOut,im2uint8(filtered));
```

The code was then run, and an output video of the signal was produced.

Attempt 3:

From attempt 2 a video of the change in colour (signal) was obtained. Wu et al. showed the results by overlaying the signal back over the original video. This would show the face of the subject from the captured video and also the colour change amplified on top of each other. An example of their results can be seen in Figure 5. This enables human eyes to see the colour change that is occurring but which they are not normally able to see. This was attempted using the signal obtained by using the RGB colour space.

The code from Wu et al. shown below is a part of how the original video and the signal are combined. The original video is read, and each frame is converted from RGB to NTSC. Then the filtered stack which contains the signal has the pyramid collapsed and is added frame by frame to the original video. Then the combination of them is converted back into RGB in order to be formatted into a video.

```
temp.cdata = read(vid, i);
[rgbframe,~] = frame2im(temp);
rgbframe = im2double(rgbframe);
frame = rgb2ntsc(rgbframe);

filtered = squeeze(filtered_stack(k,:,:,:));
filtered = imresize(filtered,[vidHeight vidWidth]);
```

```
alter = filtered+frame;
frame = ntsc2rgb(alter);
```

As Attempt 2 was not conducted using the NTSC space, this part of the code needed to be altered. Shown below is how the code was altered. All that had to be changed was the frame was not converted to NTSC and then the rgbframe from the original video was added to the signal and there was no need to convert back to RGB as the array was already in that colour space.

```
temp.cdata = read(vid, i);
[rgbframe,~] = frame2im(temp);
rgbframe = im2double(rgbframe);
filtered = squeeze(filtered_stack(k,:,:,:));
filtered = imresize(filtered,[vidHeight vidWidth]);
orig = filtered+rgbframe;
```

Attempt 4:

The videos that were taken for this project had the subjects face fill as much of the frame as possible. However, there was still a larger amount of background than was wanted.

Code was found from (*Webcam Heartbeat Monitor with Matlab Code*, 2019) It was made freely available on YouTube. It used MATLAB commands to detect the face area and generate a box around the region of interest. The code below calls a command from MATLAB's computer vision toolbox, Cascade Object Detector and reads the third frame from the selected video. From that one frame it is able to generate a box around the face.

```
faceDetector = vision.CascadeObjectDetector();
videoFileReader = vision.VideoFileReader('P1010009.MOV');
videoFrame = read(v,3);
face_box = step(faceDetector, videoFrame);
```

This code was run on the original video and the dimensions of the face_box for that video were stored. Then code was made to crop the output video from attempt 2 so that only the area within the box was kept. The cropped frames were stored in a video formatted array and then written as a video.

```
    thisFrame = read(v, i);
    newframe = imcrop(thisFrame, face_box);
    videoFrames = cat(4,videoFrames, newframe);

    open(vidOut);
    writeVideo(vidOut,im2uint8(videoFrames));
    close(vidOut);
```

Attempt 5:

A video is comprised of many single frames. The videos used in this project were filmed at 30 fps. In one second, 30 frames are captured. When playing a video, in each second, 30 frames are displayed. A typical heart could beat once every second. Viewing the information obtained in a video format was still limiting for human eyes. Another way of viewing the information was needed to fully reveal what had been captured and amplified.

Since the information is within the video, it was a matter of considering ways to easily see each individual frame rather than 30 in one second. Code was developed to display an image that comprised of 60 frames in a grid of 5 by 6. 60 was chosen as it is likely to contain at least two heart beats. However, both the number of frames to be displayed and the grid size can be changed

Once the video of choice is declared, the code below will read that video for the chosen frame range. 20 to 80 was used here. Then a montage image of the frames is created in the size of 5 by 6. Each grid space is one frame from the frame range. The MATLAB command "montage" will generate the output image.

```
videoFrame = read(v,[20 80]);

montage(videoFrame, 'Size', [12 5]);
```

3.4 Pulse rate

3.4.1 Attempts

Attempt 1:

The code from the project by Wu et al. is for detecting and revealing the colour change that occurs. It does not use this to calculate the pulse rate. During the stages of processing using that code, the signal was in a 4 D format or 3 D format. The signal could be visually expressed as a video, but could it be plotted? Attempts were made to use the output video and the signal in the format they were created in. Also attempted was ways to change their format from 4D into 2D or 1D as those are more easily processed in MATLAB.

Attempt 2:

Further investigation was required. Possible options considered were to manually count the occurrences of the colour in the face from the video played back frame by frame, convert the 4D signal into a 2 or 1D format, start from beginning and find another way to detect the colour change that will produce an output in a more workable format.

The code found at (*Webcam Heartbeat Monitor with Matlab Code*, 2019) had the solution. It captured the colour information from an image differently. It runs over each frame, pixel by pixel and determines the mean value of the green colour for that frame and stores it in an numerical array. It does not perform any image processing. An array of numbers is much easier to work with.

```
frameSquareMeans = zeros(1, frameCount);
for i = 1:frameCount
    frame = read(v,i);
    frameSquareMeans(i) = mean(mean(frame(hRegion, wRegion, colour)));
end
```

frameSquareMeans is the array that holds the mean value of green for each frame. This is the start point of the signal. MATLAB then performs a FFT and the code applies a band pass filter with a range of 0.66 to 4 Hz. This is heart rate range of 40 to 240 bpm. After the band pass filter the signal is then inverse Fourier transformed and amplified. The code that was found does all these processes then overlays the signal on the original video to show the colour change it. It does not determine a heart rate.

Out of interest, the code was run to see what video output was produced from placing that signal back over the original video. As it has taken a mean value of the green for each frame, the signal is just the value of the amount of green. When applied over the original video, it does not map to the face, it fills a box area with the amount of green for that frame.

Attempt 3:

A signal had been obtained that was in a format that could be used. Using the MATLAB plot command enabled the signal to be seen. That then lead to research into how to determine a pulse rate from a signal with a pulsating nature caused from heart beats.

MATLAB has a command called findpeaks. The code line below stores the number of each peak to the variable pks and the frame number at which the peak occurred to locs for the data held in the variable signal.

```
[pks, locs] = findpeaks(signal);
```

Each peak was to be a heartbeat. Next the count method of determining a pulse rate was used. N stores the value of the total number of frames divided by freq which is the frame rate (30 fps). N is the total time of the video determined by number of frames. Lower case n is the total number of peaks from the signal. The pulse rate calculation for the entire video is the number of peaks that occur divided by the time frame over which they occur, then multiplied by 60 for the result in minutes.

```
N = frames_num / freq;
BPM = (n / N) * 60
```

This method of determining the heart rate is the count method. It is the same calculation as placing fingers onto a wrist and counting the number of beats over a time frame, 15 seconds and then multiplying that result by 4 to get the average bpm.

Attempt 4:

Not every peak that occurs is a heartbeat. Looking at the plotted output of the signal it was noticeable that some of the peaks that were being counted fulfilled the criteria of a peak (was higher than the value prior and after it) however was not a heartbeat. In MATLAB parameters can be set for the findpeaks command.

```
findpeaks(signal, 'MinPeakHeight', TimeAveragedSquares, 'MinPeakDistance', 
10,  'MinPeakProminence', 10);
```

MinPeakHeight sets a minimum value over which a peak must occur. MinPeakDistance sets the minimum distance that must occur between each peak. For this signal, it is the number of frames. MinPeakProminence sets the minimum height a peak must have increased from in order to be a peak. These parameters aided in defining what was to be determined a peak from the signal. Trial and error and common sense were used to test suitable values for these parameters.

Attempt 5:

Pulse oximeters calculate the heart rate by determining the amount of time that has passed from the current beat to the previous beat. As each new beat occurs the heart rate for that beat is calculated and stored. The mean of each stored heart rate is determined and displayed. This operation is performed over a rolling time period of 8 seconds. Heart rate data from more than 8 seconds is discarded so that the mean is only calculated from 8 seconds worth of data.

This attempt was to try this method of pulse rate calculation.

```
peakBPM = (freq / (locs(pk) - locs(pk-1)) ) * 60;
```

The code line above calculates a pulse rate by dividing the freq (fps is 30) by the distance the previous peak is from the current peak. Locs is the frame number at which a peak occurs and pk is the number of each peak. Pk is coded to start from 2. The code runs in a loop, increasing the number of pk each time until the last peak.

`locs(pk)` is the frame number at which that number of peak occurs. At pk value of 5, it will hold the frame number at which the fifth peak occurs.

`(locs(pk) - locs(pk-1)` This part calculates the distance in number of frames the current peak is from the previous peak.

The number of frames per second is divided by the number of frames that occur between two peaks and then multiplied by 60 to return a bpm. The bpm calculated is from those two peaks only and is stored in the array created below. Batt is just the name given to the array to store the bpm values. As the value of pk increases, each new bpm is calculated. Mean(batt) takes the mean of the values stored in the array batt. HR is used to display the mean bpm as it changes with each new pulse rate calculated and it is stored in another array called BPM.

The final value in the BPM array is the mean of all values that were calculated from the distance between the peaks.

```
batt = [batt, peakBPM];
HR = mean(batt);
BPM = [BPM, HR];
```

Attempt 6:

The count method and peak to peak method are both calculated in the time domain. From the research however it is also possible to calculate the heart rate from the frequency domain.

MATLAB performed the FFT on the signal and then plotted it. The heart rate can be calculated by finding the frequency at which the data spikes at and multiplying that frequency by 60.

3.4.2 Plotting

Once the signal was obtained it could be plotted with the MATLAB plot command. However, for this project, since the data is gained frame by frame, it was then attempted to plot it frame by frame.

This involved getting a loop that ran a variable (i) from 1 to the total number of frames. Then plotted the signal data frame by frame, for each value of (i).

In addition, it was attempted to have each peak marked or indicated as it was plotted to make it clear to other viewers. This was a matter of coding a marker to be placed at the value of the peak each time (i) equalled the frame number at which a peak occurred.

Running the code in MATLAB the user is able to see the heart rate values that are being calculated. However, for the ease of other users it was then attempted to display values on the plots generated. The two values to be displayed were the bpm from each peak as it was calculated and the cumulative mean of the bpm values.

3.5 User interface

The code files can be opened and run in MATLAB. However, that is not intuitive for those unfamiliar with the code and process of analysis. Comments were used throughout the code to break into in sections, title the section and explain parts and parameters. This was more for the group members. Several MATLAB toolboxes had been used for this project. If another were to run the code, those toolboxes would be required. MATLAB has an App Designer component in which an app can be built. The app would hold all the commands and toolbox functions necessary to run the code.

The intension was to design an interface that clearly shows the options. A button that will open a folder option where the user can choose which video to run. The confirmed heart rate for the video would be displayed. Also displayed were the results from the methods of pulse rate calculation. So that users can easily see all at once the results from each method.

The app reads and stores data differently to a code file. The code needed to be adapted for the app's code environment. All code that had been worked on for this project was fully available to the other group member. The area of work of the other group member was adapting the code to operate in the app and code the functionality of the app.

Chapter 4 Results and Discussions

4.1 Revealing a colour change

Attempt 1:

The first attempt was to run the code from Wu et al. unaltered on videos that had be recorded of the group members. It was expected that the results would be similar to that achieved by Wu et al. However, this was not the case. The code had a few parameters that could be adjusted by the user, level of pyramid (tried 4 and 6), amount of magnification (tried 50 and 100), and the frequency range for the band pass filter (tried 0.8 to 1.5 and 0.8 to 3). Despite trying several combinations for the settings, the results that were occurring were not similar to the Wu et al project results. Figure 6 shows a snapshot from the video output of the signal without the original video added in. The signal is not capturing the changes in colour in the face.

Figure 6: A snapshot of the video output of the signal generated from Wu et al.'s code using our video.

This snapshot is one of the occurrences of the greatest amount of colour in the video. Edges of the face would slowly get brighter then dim to black. Only some of the edges of the face and the eyes are shown which is a lot less colour that the results achieved by Wu et al show. It was decided that this attempt at using the code unaltered was unsuccessful.

We were not the only ones to not have success in running the code by Wu et al. Another group, Alcala et al. also tried to use the code for their project. In their report they stated that they were not able to reveal the colour changes in the face in the time frame of the project. It was producing an output but it was not similar to the output examples shown in Wu et al.'s paper. (Alcala et al., 2016) They did not suggest a reason for this. It is not known why we and other groups were unable to achieve the desired result using the Wu et al. code unaltered. However, the literature provided another path to try.

Attempt 2:

Given the literature on using the green colour wave in iPPG, we decided to alter the code from running in the YIQ colour space, to the RGB colour space and amplify the green colour.

This produced a better result. Figure 7 shows a snapshot from the video output of the signal when the code was run in the RGB colour space.

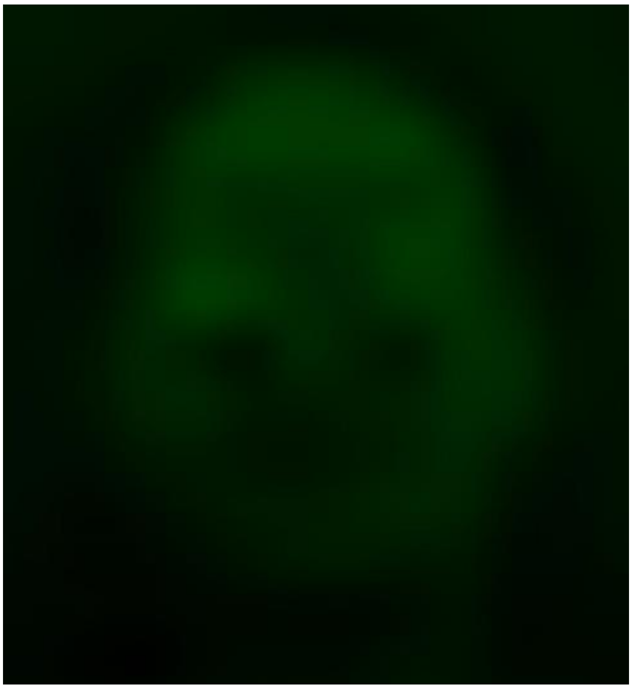

Figure 7: Image of the resulting signal from altering the code for RGB.

The results shown by Wu et al. display the signal over the original video. We had no guide as to what the signal on its own might look like. Our video of the signal shows that by using MATLAB we were able to detect and amplify the changes in green light that are reflected from the face. On a heart beat the blood from the arteries pushes and contracts the skin causing more light to be reflected. There will still be a small amount of green light reflected on the lull between heart beats. However, the video output signal was obtained by multiplying the signal by 100. In doing this, the green that is reflected from a heartbeat is just now visible. When the green is at the its peak in colour means that the heart has pumped blood through the body forcing the capillaries to move closer together and more green light to be reflected back. On the video, it appears as flashes of green from blackness.

Attempt 3:

With a signal now found and a video output of it obtained, the next attempt was to overlay the signal over the original video. This is how Wu et al. displayed their results and that was the reason why we decided to try this too. That produced some unexpected results. It was expected that the face would become greener from having the signal over top. That did happen but the colours went from green to purple and back and forth with each heartbeat. It is likely that this is due to the RGB colour space and some part of the signal data is causing the other end of the colour spectrum to appear on the lulls between beats. We were unable to discover the cause of this or fix it. However, we had achieved obtaining a signal of the colour change which was a part of our project goals.

Attempt 4:

Watching the output video of the signal we noticed that a lot of the background was black, and we also were only interested in the face area. Code was developed to crop the video to the face area. In MATLAB you can run the face detection continuously over an entire video. That required a lot of processing power from the computer and caused some issues. To deal with this, we referred to our project goals and the scope of the project. The videos we had taken had the subjects stationary and as still as possible. While their heads likely moved a little, the few millimetres were an acceptable loss. We used the face detection code on the original video to define the dimension of the face box. Then the output signal video was cropped to the dimensions of the face box. As shown in Figure 8 the face box is quite generous in size. Even if the head were to move a few millimetres, the larger face area was still included.

Figure 8: Frame 6 of Video 09 with face detection box.

This is how we were able to crop the signal video. This would not work on a moving object since the box is pre-defined and does not change for the duration of that video. MATLAB object detection is capable of running on moving objects. However, if live object detection and image processing is to be run on live footage, the computer will need to be able to handle the requirements of performing that. Ours could not. Since we had defined the scope of our project, we were still achieving what we had set out to do.

Attempt 5:

While watching the output video of the signal it occurred to us that we were still unable to see all the information from the signal. I wanted to be able to see a series of frames. Scrolling along the video player timer bar was showing me 30 frames in one second. It was possible to code the video to run in MATLAB playing back frame by frame and the display time could be varied. However, I felt this was not the best way to display the information. Code was developed to display a number frames in a montage.   A snapshot of the resulting image is shown in Figure 9.

Figure 9: Montage of Output signal video from video 09. Showing frames 35 to 50.

Figure 9 is only showing 20 frames. Two full 60 frame montage images are in Appendix A.

This code enabled us to see a range of frames at once.   Rather than 30 in one second or one after the other. This way the signal can be more clearly expressed. It was interesting to see how many frames a heart rate was detected over and also there was often a peak green just before or after the main heartbeat. This relates to physical motions of a heart as it completes a cardiac cycle.

4.2 Pulse rate

Attempt 1:

It was initially believed that the code from the project by Wu et al would produce a signal of the variation of colour in the face. From this it would then be a matter of determining when the greatest amount of colour occurred and when the least amount occurred and presenting that as beats per minute.

However, the EVM method uses image processing techniques to make things that are not visible to the human eye, visible. In order of it to achieve this, the data it is processing stays in 4D or 3D arrays.

We were unable to find a way to further process these arrays. We could not get the output signal to plot. Visually we could see the signal as a video, but we could not find a way to process it further to determine a heart rate. As this was a new area for us, we were worried about destroying the signal. Wu et al. had created the arrays in a specific way. They had defined what was stored in each of the 4 dimensions. To move from 4D to 2D would mean losing some information. We were unsure about doing this.

Attempt 2:

After some research another code was found. It does not perform EVM nor any image processing. But it claimed to be able to detect the colour change. We tested it out. The code did not calculate a heart rate, but it did create a 1D signal of the colour change.

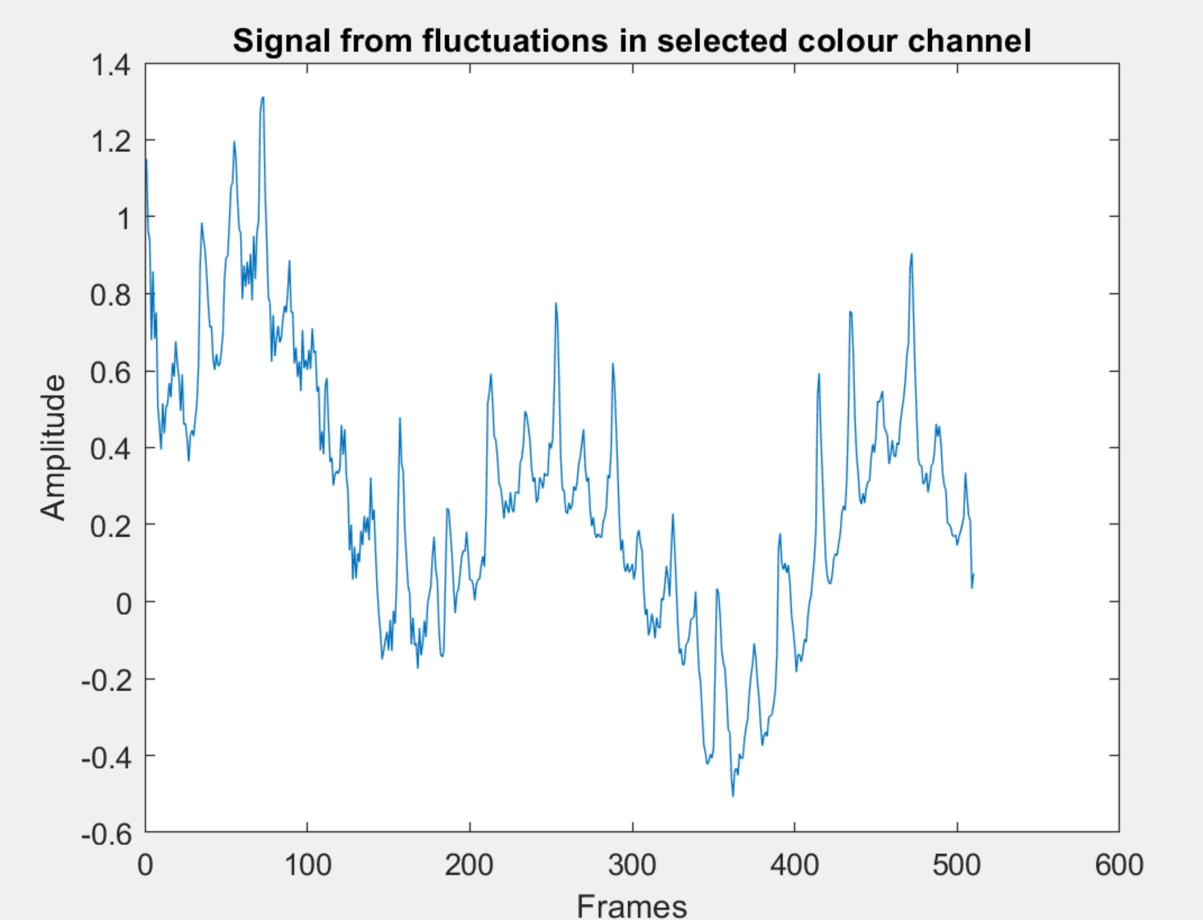

Figure 10: A plot of the data obtained from the mean value of green from each frame.

Figure 10 shows the signal from one of the videos. This signal is created from the mean green light amount in each frame. The code also provided the MATLAB commands for the other processing required, the band pass filter. The camera is capturing a lot more information than we realise. The band pass filter narrows the area of interest to events that occurs within its range. For this project, it was the heartbeat, 0.66 Hz to 4Hz. In order to have the filter applied, the signal needs to be in the frequency domain.

Figure 11 shows the plot of the signal after it has been FFT in MATLAB. The next step in the code is to apply the band pass filter. On Figure 11, it is the small peak area between 0 and 5 Hz that we are interested in.

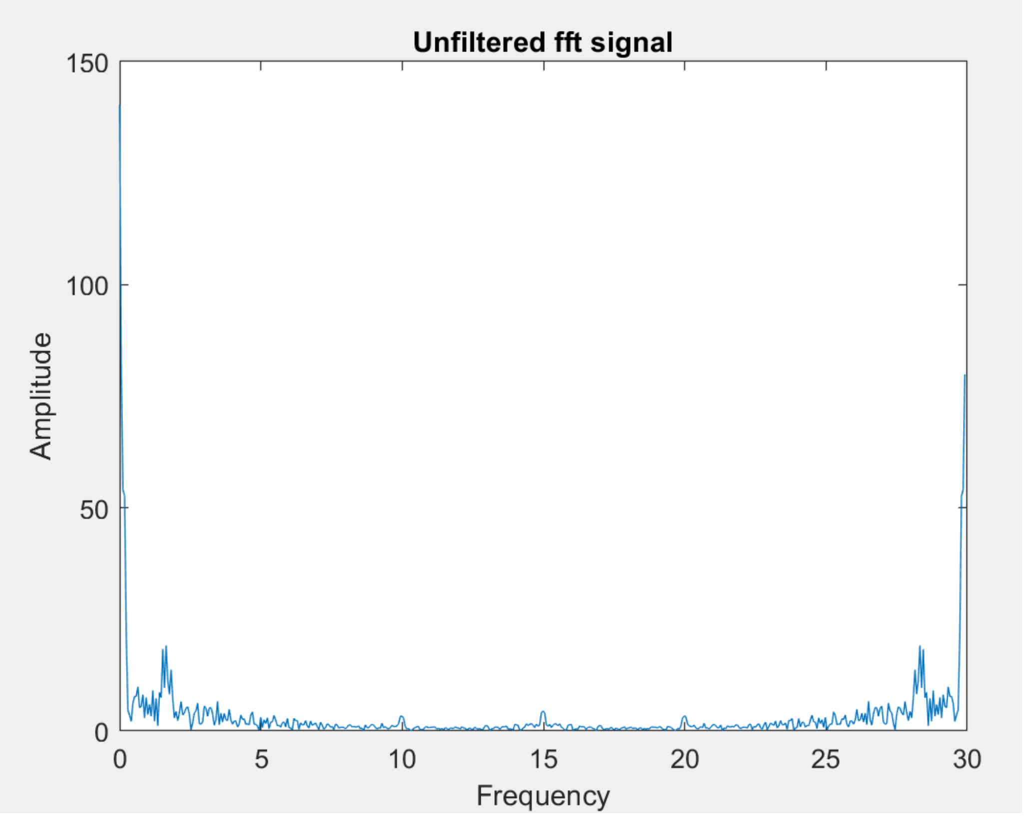

Figure 11: The signal plotted after it has been FFT

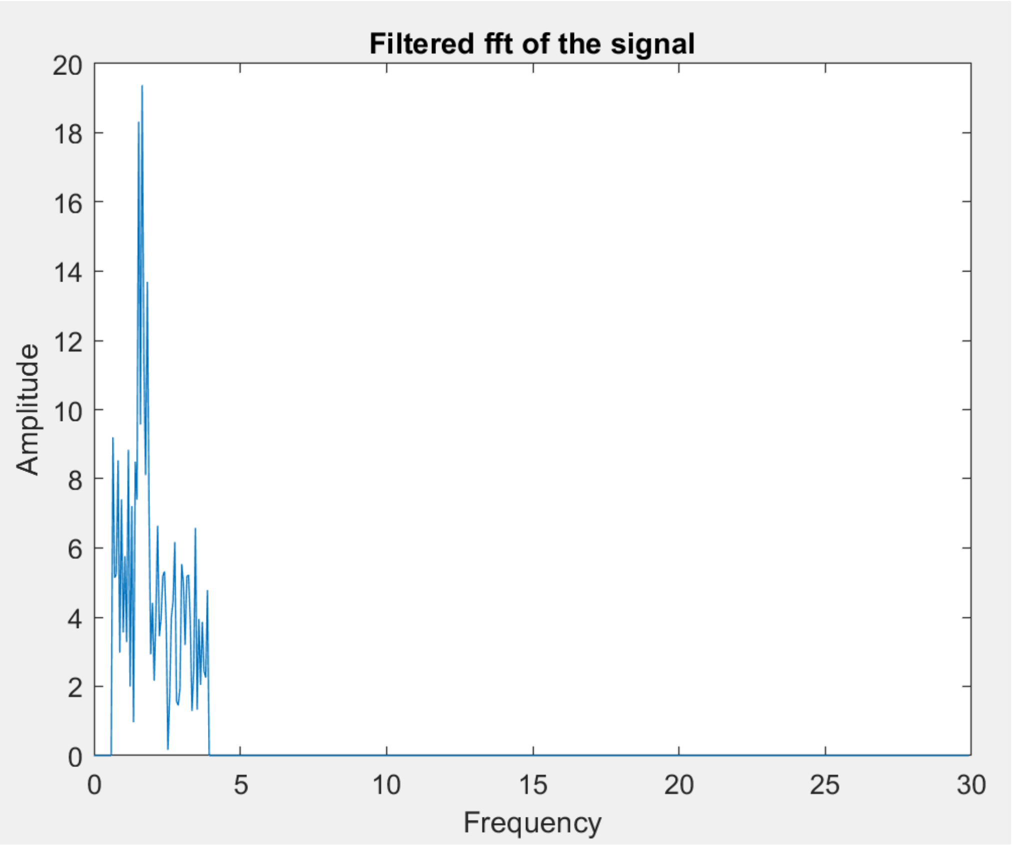

Figure 12: Signal plotted after FFT and after the application of the bandpass filter (0.66 - 4)

Figure 12 shows the plot of the signal in the frequency domain after the band pass filter has been applied. By applying a band pass filter, we are removing the frequencies that are not of interest to us. After this, the inverse FFT is performed and the plot of the signal can be seen in Figure 13.

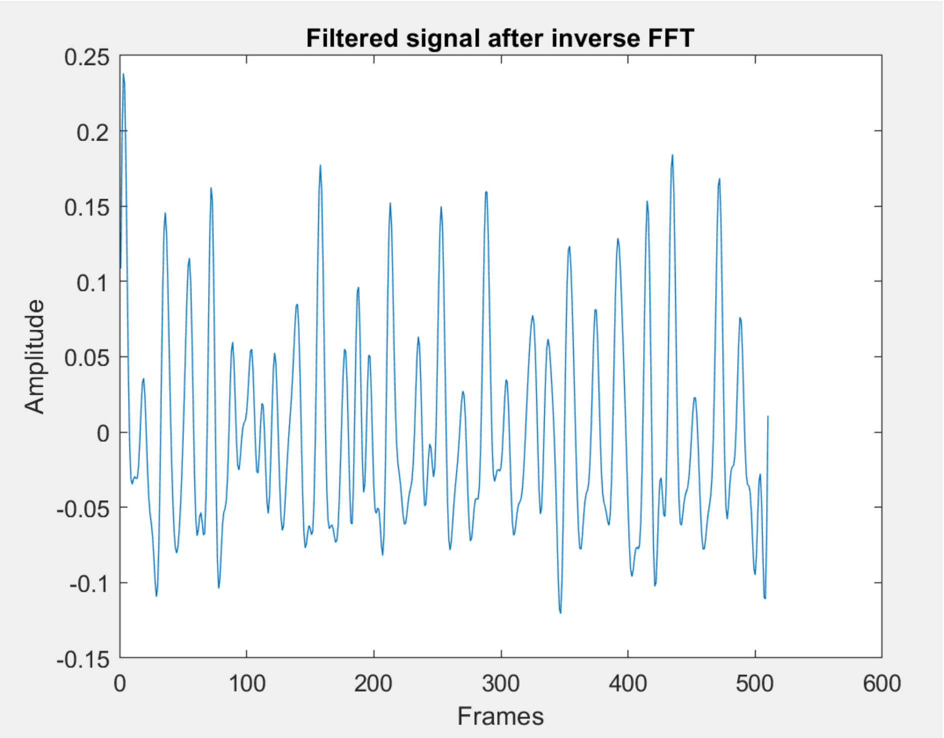

Figure 13: IFFT applied to the filtered signal then plotted

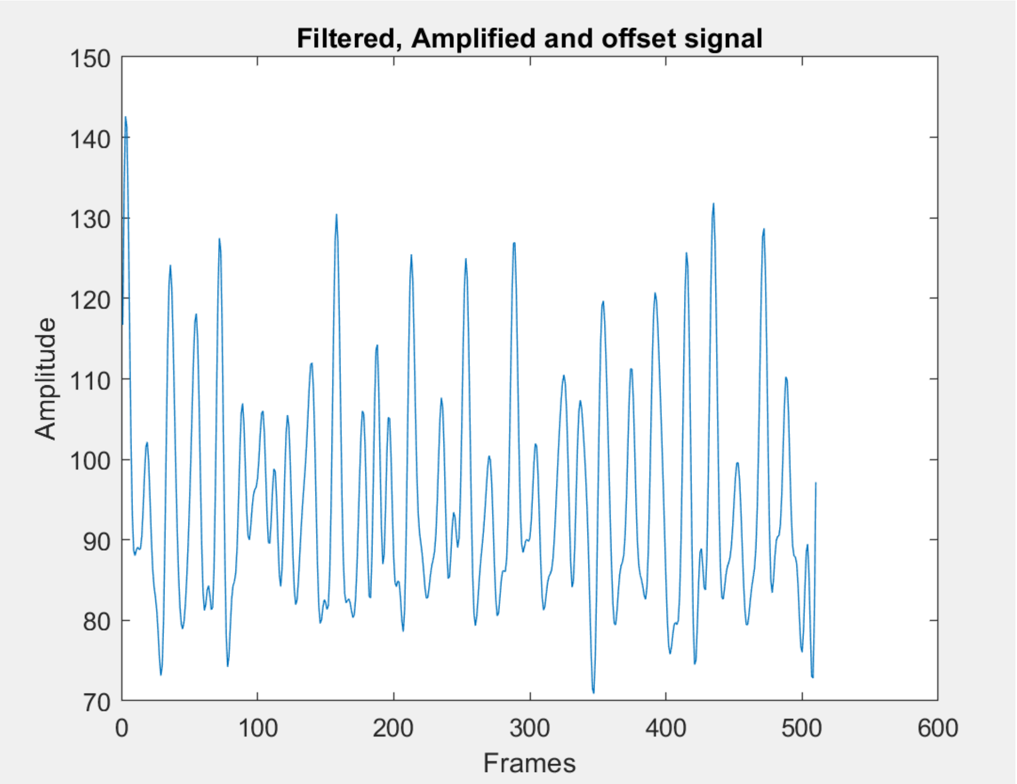

Figure 14: Signal has been filtered, amplified and offset and plotted

Figure 14 show the final signal.   It has been filtered and amplified.   We had obtained a signal that shows the changes in green light reflected from a face.

Attempt 3 – 6:

Now that we had a signal, it was to be used to determine a pulse rate.   With a signal in 1D, MATLAB could run commands on it, so performing calculations and plotting was achievable.

Attempt 3 was about using the signal to return a bpm value. Attempt 4 was developing peak detection methods to refine the signal. Attempt 5 was calculating a heart rate from our signal using the same method as a pulse oximeter.   Attempt 6 was to use the frequency domain to determine the pulse rate.

First, we did use the count method on the signal just to see if we could get a result. Though, before testing all the pulse rate calculation methods on the signals, we needed some additional parameters to define what peak event was from a heartbeat and what was not.

A peak is a data point where the value preceding it and following it are both less than it.   Not all peaks by this definition were the result of a heartbeat.   Likely many were due to noise in the signal.   Therefore, it was necessary to develop peak detection methods. The three parameters used were MinPeakHeight, MinPeakDistance, and MinPeakProminence.   All of these were tested on trial and error for the best fit for our videos. However, there was common sense and theory behind what was tried.   MinPeakHeight is minimum threshold level that a data point must be over in order to be a peak. This eliminated the peaks shown in Figure 15 below from being included as a heart rate peak. They are fluctuations in the data gathered but not a heartbeat.

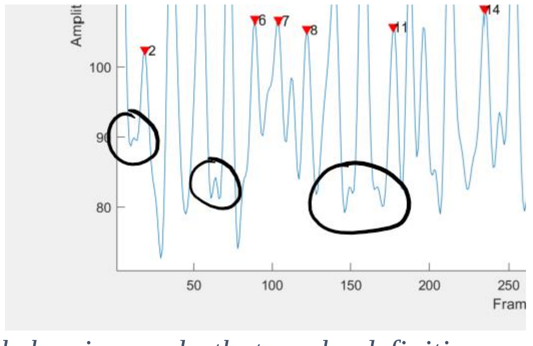

Figure 15: Image of signal showing peaks that are by definition a peak, but not a heartbeat peak. These peaks will be reduced by a minimum peak height requirement.

MinPeakDistance sets a minimum distance each peak must occur from another.   In the time domain the x axis is number of frames. The average adult has a heartbeat once every second, which from the video is 30 frames.   This parameter was set at 10 and in doing so eliminated the inclusion of peaks that were too close as seen in Figure 16. The variation dip between the peaks is very small on the original signal.   That variation is likely due to noise. We know that peak likely occurred, and noise may have affected it somehow. As for which of the peaks should have be included, that was not explored in this project.

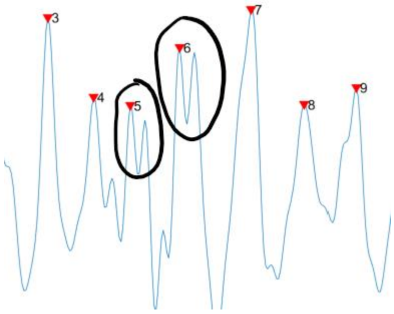

Figure 16: Image of signal showing peaks that are by definition a peak, but not a heartbeat peak. These peaks will be reduced by a minimum peak distance requirement.

MinPeakProminence sets the minimum height a peak must increase from in order to be a peak.   This was to eliminate the inclusion of the peaks circled below in Figure 17.   This was set at 10.

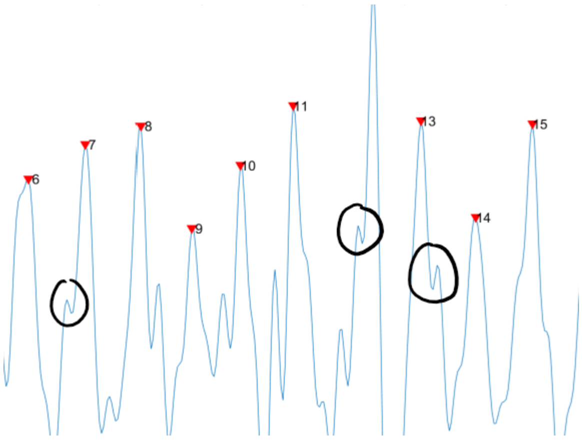

Figure 17: Image of signal showing peaks that are by definition a peak, but not a heartbeat peak. These peaks will be reduced by a minimum peak prominence requirement.

The signal we have obtained has noise. There may be additional filming parameters or code that can further reduce the noise and smooth the signal. However, these peak parameters did give one simple way to apply restrictions on the peak detection. The parameters were tested on all our videos and the values for a best overall fit were chosen for our pulse rate calculation methods to then be run.

Below are two tables. Table 1 shows the results for the videos of subject 1. Table 2 shows the results for the videos of subject 2.

Each table displays 5 columns of values. The first is the range of the values that was recorded from the pulse oximeter during the filming of that video. The second is the average of the pulse oximeters range. The values in three, four and five are found from the signal we obtained. Different methods of determining a pulse rate were used. The third uses the count method, the fourth is from the frequency domain and the fifth is from peak to previous peak calculation.

Table 1: Table of subject one's videos with the ground truth pulse oximeter reading and results from three methods of pulse rate calculations

Subject 1: JS Resting	**Pulse Oximeter range**	**Pulse Oximeter average**	**Pulse rate Count method**	**Pulse rate Frequency domain**	**Pulse rate Peak to peak**
Video 01	65 - 68	66.5	65.74	65.74	67.57
Video 02	60 - 63	61.5	63.69	63.69	64.93
Video 03	80 - 108	94	76.29	87.19	83.91
Video 04	72 - 80	76	67.43	48.70	71.71

Table 2: Table of subject two's videos with the ground truth pulse oximeter reading and results from three methods of pulse rate calculations

Subject 2 : SK **Resting**	**Pulse Oximeter range**	**Pulse Oximeter average**	**Pulse rate Count method**	**Pulse rate Frequency domain**	**Pulse rate Peak to peak**
Video 09	98 - 110	104	98.72	98.72	103.08
Video 10	99 - 104	101.5	94.45	98.08	95.75
Video 11	99 - 102	100.5	92.48	99.33	97.01
Video 13	100 - 105	102.5	94.45	105.35	99.58

It needs to be noted that the pulse oximeter has a deviation of $\pm3\%$ and it calculates its average value over a time period of 8 seconds. For our results, they were calculated over the time period of the videos which was between 15 and 18 seconds. This may cause variation as our values hold double the amount of data for calculation than the pulse oximeter. The effect of a heart rate spike will be removed from the data of the pulse oximeter after 8 seconds, but in our code, it is still affecting the value.

Video 01 and 02 are the subjects natural resting heart rate of around 60 to 65 bpm. Then their heart rate increases to 20 bpm over their normal resting pulse rate value. Our code was more accurate on the videos with the subjects true resting pulse rate values (Video 01 and 02) than on the other two videos (Video 03 and 04).

The natural resting heart rate for subject one was 60-65 bpm and for subject two was around 100 bpm. The code was accurate on the videos that captured each subject’s resting heart rate, but less accurate as the heart rate value changed from their normal resting range.

The result from Video 04 in the frequency domain is very low.

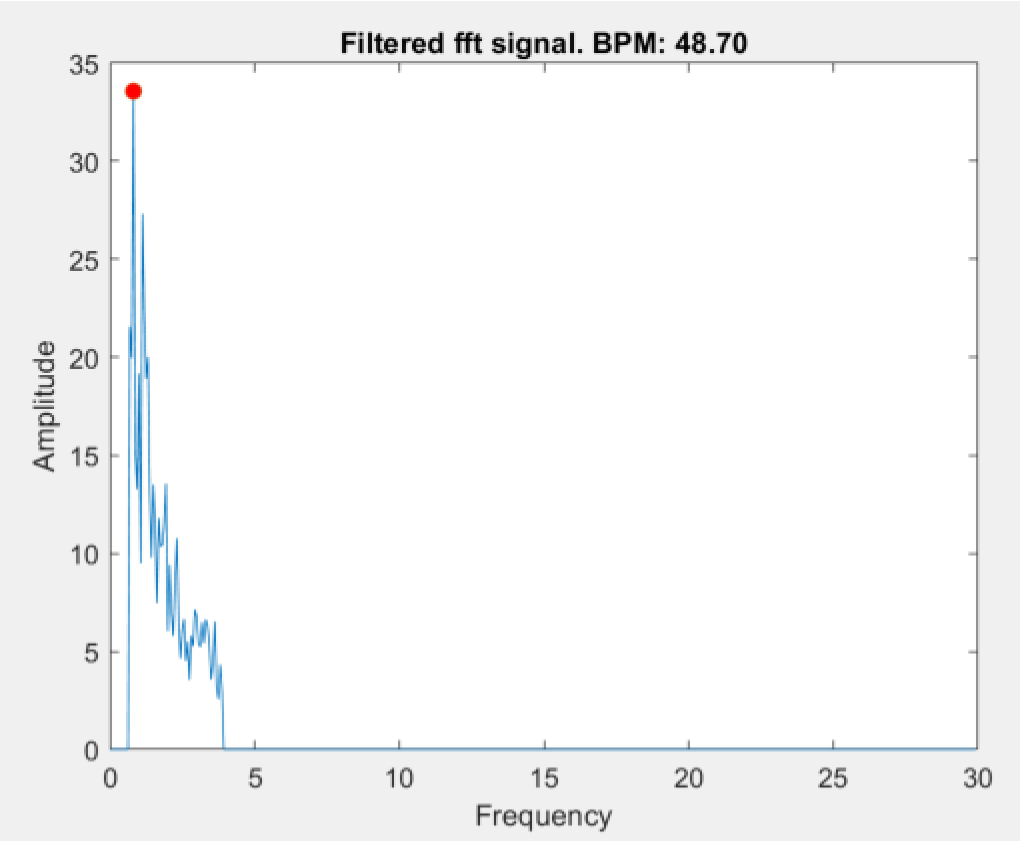

Figure 18: Plot of bpm detection in the frequency domain to show the effect that the slightest variance has on the calculated bpm

Detecting a pulse rate from the frequency domain can be difficult with additional noise in the signal. Ideally the highest peak is chosen and then the frequency at which that peak occurs is multiplied by 60 to give a bpm. This is shown in Figure 18. If that value is off even slightly, you are getting a very different value as an output.

It is interesting that the different methods to calculate a heart rate can produce quite different results. Taken Video 13 as an example.

Table 3: Table of bpm values for Video 13 to show the variance in pulse rate calculations

Subject 2 : SK Resting	**Pulse Oximeter range**	**Pulse Oximeter average**	**Pulse rate Count method**	**Pulse rate Frequency domain**	**Pulse rate Peak to peak**
Video 13	100 - 105	102.5	94.45	105.35	99.58

The pulse rate calculated from the count method was a bit too low and from the frequency domain it was a bit too high to the pulse oximeter average.

However, different methods of calculating pulse rate is not the subject of this project. They just provided us with addition methods to try in our goal using the change in colour to calculate a pulse rate.

It is also worth noting that this project was an attempt to get a pulse rate output. There were many parameters that could have been factored in but just getting from where we started to this point is an achievement. Therefore, there are noise factors in the signal that will affect the results. Noise from lighting, skin pigmentation, and our coding of the filters.

4.2.1 Plotting

We wanted to be able to show the data frame by frame. We developed a loop to plot the signal by each frame. Then we worked on getting the peaks to be indicated by a marker, so that it was clear which peaks were from heart beats. The result of those attempts is shown in the top plot of Figure 19.

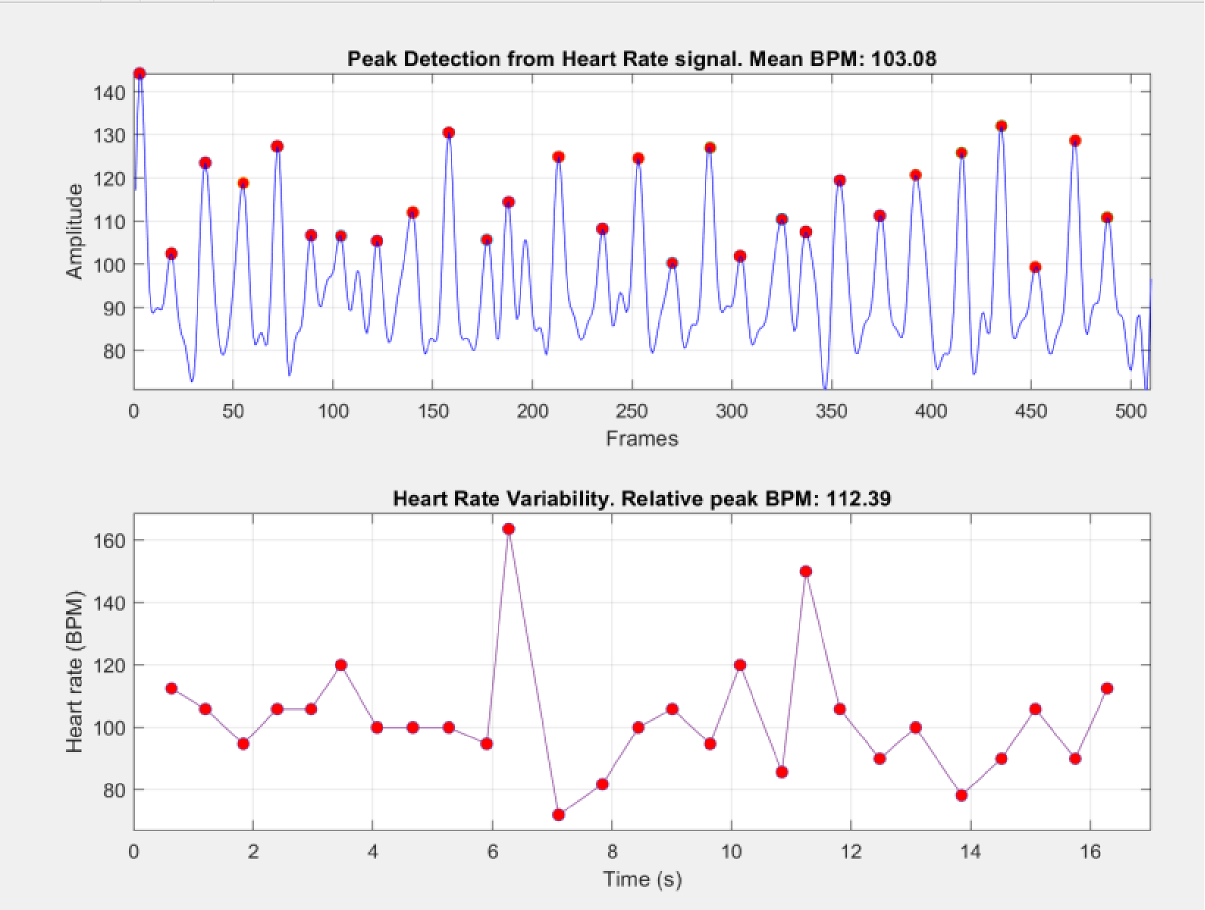

Figure 19: Top plot is the final signal plotted with markers to show the detected peaks. Bottom plot shows the HRV as it is a plot of each bpm as it is calculated from each new peak.

Given that the peak to peak method of calculating a heartbeat is calculated between two peaks, we then though we would try and plot the value of the pulse rate as it is calculated from the signal. The result of this is the bottom plot in Figure 19.

As the top plot runs, once the second peak is plotted, it calculates the peak to peak pulse rate and that bpm value is plotted on the bottom plot. At the third peak, the bpm calculated from the distance between the 3 and 2 peaks, and the bpm is plotted on the bottom plot. This is what is known as heart rate variability as discussed in the literature review.

4.3 App

The code that had been developed and used was to be placed into app for easier use by other users.

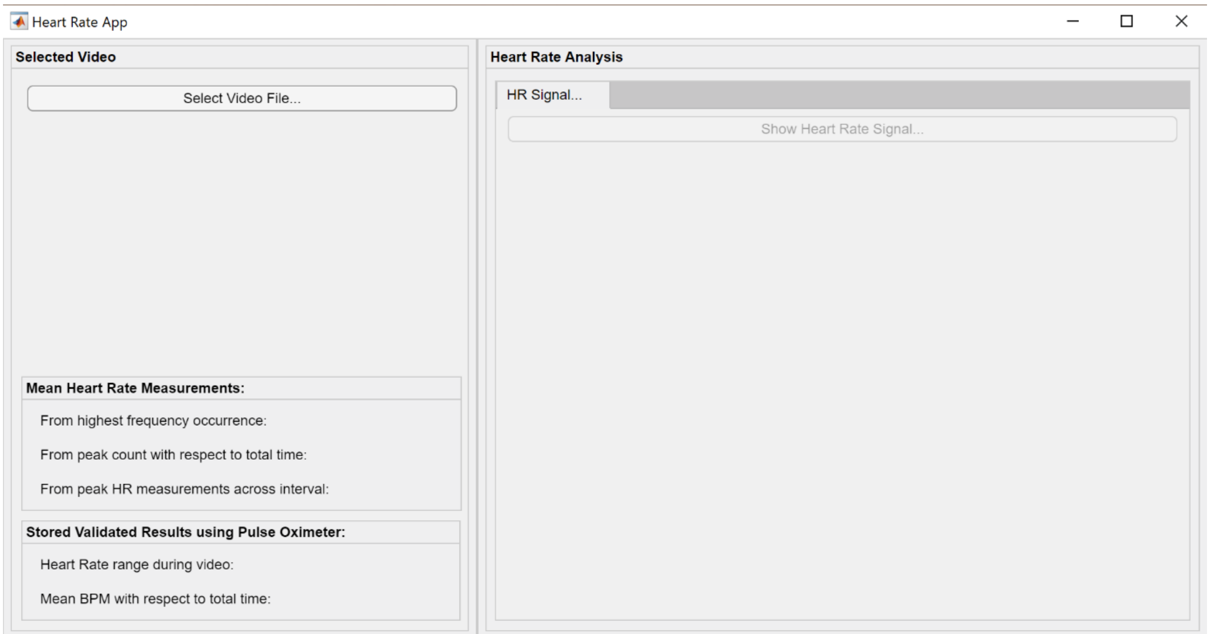

Figure 20: Final decided layout of interface for app

It went through several design versions. Figure 20 shows the final layout that was decided on. All decisions in layout were made by both members, but the coding for the functionality of the app was the sole work of the other member. Figure 21 shows the final app product. When you press "Select video file" a pop up opens where you can use any of our recorded videos. When you select one, the first frame of the that video appears in the window. Beneath the image is the pulse rate results. The bottom box displays the range from the pulse oximeter and the average bpm that we used as our ground truth. The box above that displays the results from our calculations from each of the three methods we used.

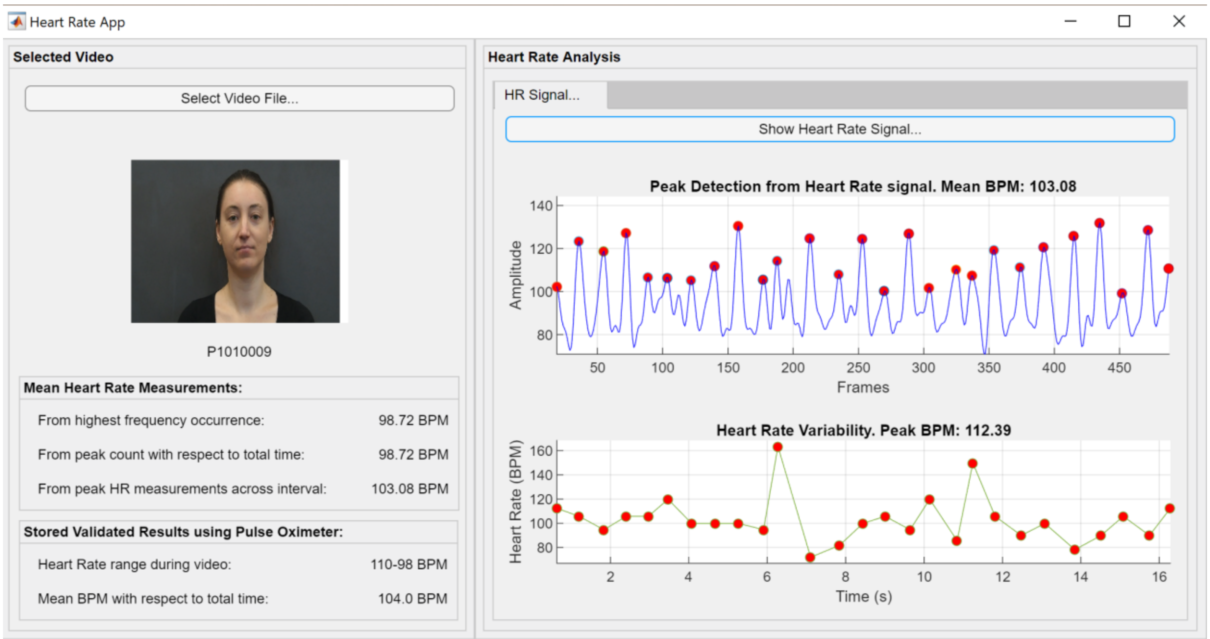

Figure 21: App working with video selected, HR values displayed and the plots of the signal and HRV

On the right side of the app, once a video has been selected, you can press “Show heart rate signal” and the plots that were created will run frame by frame through the signal.

All code that I had developed for obtaining the green colour signal, processing it and the methods of pulse rate calculation and plots were used by the other group member to be incorporated into the app. Some changes had to be made for it to run in the architecture of the app code.

Chapter 5 Conclusions and Future Work

5.1 Conclusions

The code from Wu et al. was very difficult to understand and work with. But in attempts and research to find solutions a lot of understanding was gained. In the end the code from Wu et al. was only used to visually show the colour change that is occurring but is not detectable by human eyes. Another code was found that detected that colour change but stored that information in a different format. Once a signal of the colour change was obtained then determining a pulse rate could be attempted.

Code was created that determined a heart rate using three different methods, the count method, in the frequency domain and the peak to previous peak method (as used by the pulse oximeter). Code to produce a double plot that plots the signal obtained with peak markers in the top plot and in the bottom plot is heart rate variability from peak to peak bpm calculation was also created. This was achieved from videos captured on an entry level DLSR camera and code commands from MATLAB. For a resting heart rate, the code was accurate.

5.2 Recommendations for Future Work

This is a large area and future work could be undertaken in many directions. Our project only works on a video of a stationary resting subject. That has a very minimal application. Future work could attempt to expand on the capability of this project. For smaller steps from this project it would have been interesting to have taken our pulse from an EKG to see if the peaks detected using our method match up with the pulses shown on an EKG. However, timing with this would have to be carefully considered. The time stamps for the start of filming and start of data from EKG would have to be set to the same point and the camera does take 30 frames per second. Another step up from this project would be to get it running from a live webcam feed. This is where the capabilities of OpenCV may be of use. There is a software called OpenCV (Computer Vision) that has amazing capabilities in the computer vision field. For face detection and motion adjustment OpenCV might be a more suitable software for that than MATLAB. However, for the mathematical processes of the gaussian pyramids and FFT, MATLAB may be the preferred software.

References

2 easy, accurate ways to measure your heart rate. (n.d.). Mayo Clinic. Retrieved 23 October 2020, from https://www.mayoclinic.org/healthy-lifestyle/fitness/expert-answers/heart-rate/faq-20057979

Alcala, K., Baldauf, K., & Sinapov, J. (2016). *Eulerian Video Magnification*. 10.

Ash, C., Dubec, M., Donne, K., & Bashford, T. (2017). Effect of wavelength and beam width on penetration in light-tissue interaction using computational methods. *Lasers in Medical Science, 32*(8), 1909–1918. https://doi.org/10.1007/s10103-017-2317-4

Bush, I., & Mall, S. (n.d.). *Measuring Heart Rate from Video*. 8.

Eulerian Video Magnification. (n.d.). Retrieved 23 October 2020, from https://people.csail.mit.edu/mrub/evm/

Feukeu, E. A., & Winberg, S. (2019). Photoplethysmography: Light Emitter Diode Wavelength Derivation from the Absorption Spectra of Haemoglobin. *2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 1–6. https://doi.org/10.1109/IMITEC45504.2019.9015874

Gaussian and Laplacian Pyramids. (2020, January 5). *COMPUTERVISION POINTS*. https://computervisionpoints.com/gaussian-and-laplacian-pyramids/

Goudarzi, R. H., Somayyeh Mousavi, S., & Charmi, M. (2020). Using imaging Photoplethysmography (iPPG) Signal for Blood Pressure Estimation. *2020 International Conference on Machine Vision and Image Processing (MVIP)*, 1–6. https://doi.org/10.1109/MVIP49855.2020.9116902

Kamshilin, A. A., & Margaryants, N. B. (2017). Origin of Photoplethysmographic Waveform at Green Light. *Physics Procedia*, *86*, 72–80. https://doi.org/10.1016/j.phpro.2017.01.024

Kranjec, J., Beguš, S., Geršak, G., & Drnovšek, J. (2014). Non-contact heart rate and heart rate variability measurements: A review. *Biomedical Signal Processing and Control*, *13*, 102–112. https://doi.org/10.1016/j.bspc.2014.03.004

McDuff, D. J., Estepp, J. R., Piasecki, A. M., & Blackford, E. B. (2015). A survey of remote optical photoplethysmographic imaging methods. *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 6398–6404. https://doi.org/10.1109/EMBC.2015.7319857

Miljković, N., & Trifunović, D. (2014). Pulse rate assessment: Eulerian Video Magnification vs. electrocardiography recordings. *12th Symposium on Neural Network Applications in Electrical Engineering (NEUREL)*, 17–20. https://doi.org/10.1109/NEUREL.2014.7011447

Rouast, P. V., Adam, M. T. P., Chiong, R., Cornforth, D., & Lux, E. (2018). Remote heart rate measurement using low-cost RGB face video: A technical literature review. *Frontiers of Computer Science*, *12*(5), 858–872. https://doi.org/10.1007/s11704-016-6243-6

Sun, Y., & Thakor, N. (2016). Photoplethysmography Revisited: From Contact to Noncontact, From Point to Imaging. *IEEE Transactions on Biomedical Engineering*, *63*(3), 463–477. https://doi.org/10.1109/TBME.2015.2476337

Understanding Color Spaces and Color Space Conversion—MATLAB & Simulink—MathWorks Australia. (n.d.). Retrieved 23 October 2020, from https://au.mathworks.com/help/images/understanding-color-spaces-and-color-space-conversion.html

Verkruysse, W., Svaasand, L. O., & Nelson, J. S. (2008). Remote plethysmographic imaging using ambient light. *Optics Express*, *16*(26), 21434–21445. https://doi.org/10.1364/OE.16.021434

Webcam Heartbeat Monitor with Matlab Code. (2019, January 13). https://www.youtube.com/watch?v=xtzVJo5SBvw

Wu, H.-Y., Rubinstein, M., Shih, E., Guttag, J., Durand, F., & Freeman, W. T. (2012). *Eulerian Video Magnification for Revealing Subtle Changes in the World*. 8.

Zhang, Y., Dong, Z., Zhang, K., Shu, S., Lu, F., & Chen, J. (2020). Illumination variation-resistant video-based heart rate monitoring using LAB color space. *Optics and Lasers in Engineering*, *136*, 106328. https://doi.org/10.1016/j.optlaseng.2020.106328

Appendix A:

Montage of frames 30 – 90 from the signal produced from Video 09 using an altered version of Wu et al.'s code.

Montage of frames 30 – 90 from the signal produced from Video 01 using an altered version of Wu et al.'s code.