# Smart Sensing in Internet of Things: Assignment 1

Rishika Varma K CS18B045

March 2021

## 1 Application survey

### 1.1 Blood Pressure Calculation

Unlike a lot of other apps this app does not use the camera and flash to measure the pulse. It requires a finger to be placed the place indicated on the screen and hold it in place for a certain amount of time (latency is of the order of 10 seconds) after which it gives the systolic and diastolic readings as well as the pulse rate. I was interested in this because of the different approach of sensing and I think that possibly the method of sensing is completely different in this case. When multiple readings are taken it remains mostly consistent. It also remains consistent when different fingers are used to take readings. I was also interested by the fact that the smoothing of the information is done in a way that the diastolic peak is not smoothed and the readings of systolic and diastolic blood pressure (which I expect is probably calculated from the amplitude) is given.

### 1.2 Cardiac Diagnosis

This app along with giving the heart rate also shows the graph that is obtained from the smoothed model of the data obtained from the video input . Using this data, it gives feedback on whether the user has any chance of common heart diseases based on some thresholds for amplitude, frequency, irregularity in frequency, etc. It is not very accurate and requires multiple tests on our side to come to a definite conclusion. This also sense via the flash and camera but it appears to be less accurate than many of its counter parts. I was interested by this app because of the different way in which it used the data instead of just giving the heartrate.

### 1.3 FibriCheck

This app interested me because it has been certified by the FDA and so I was intrigued to see how it works .It is an app that is mainly oriented towards helping diagnose heart disease. It gives the same level of data as an ECG and also takes additional information regarding whether the user was sitting, standing or lying down, and if the user has any previous history of heart or other related diseases that could cause heart disease etc in the form of a questionnaire. I believe this information will be used to give values of weights for creating a model for the diagnosis. This way the app makes custom models for diagnosis depending on the person which will clearly increase accuracy. The app senses using the flash and camera and it senses for 1 minute before processing the data for the heart rate and abnormalities in terms of frequency, amplitude (which indicates the pressure value) or any other abnormalities in heart rhythm. The heart rate is also fairly consistent over different readings using different fingers and in different lighting. Although it does have longer latency than most apps which are of the order of seconds whereas this app takes more than a minute to do so I feel it is justified by the level of detail and accuracy provided.

### 1.4 Selfie Heart Rate Monitor - Facebeat

This app if 1 of the few apps that has a different form of sensing and this is why I was interested. Along with sensing by placing the finger on the camera another option offered is through by the face. By aligning our face as indicated using the front camera of the mobile, the app senses data and gives output. The output is given continuously by taking cumulative data. The app is also quite accurate in terms of the values even if there is movement in the finger or face. The problem that I see is that of ambient light as the app does not use flash for getting heartrate from the face. In case the face is not perfectly aligned and there is higher amount of noise from the surroundings then due to frequency in the ambient light it might give faulty values.

## 2 Explanation of code experiment

To post-process the videos in my dataset and obtain perform experiments on how the outcome which in this case is the calculated value of heart rate changes with change in resolution and frame rate, I have written a python script. This script has 3 parts.

The first part analyses each frame of the video and calculates the heartrate every 5 seconds cumulatively (i.e, after the 10th second the algorithm calculates the heartrate using all frames in the 10 seconds) using 2 methods. From each frame we obtain the mean value of red and using the trend of this value from all the frames we find the heart rate. The first method used is by fourier transforming the data and finding the frequency above 1 that has maximum weight. The second method used is by smoothing the data appropriately so that the peaks remaining are given approximately by 1 peak for every pulse, counting the number of peaks and scaling it into bpm. A final value of heartrate given by analysing the entire data is given at the end.

The second part of the code is for observing how the calculated value varies with sample rate (in this case frame rate of the video). To decrease frame rate I selected 1 in f frames so that it decreases by a factor of f. By varying f value the variation in heart rate with sample rate can be observed. In the code I plotted the results for 10 different values of frame rate.

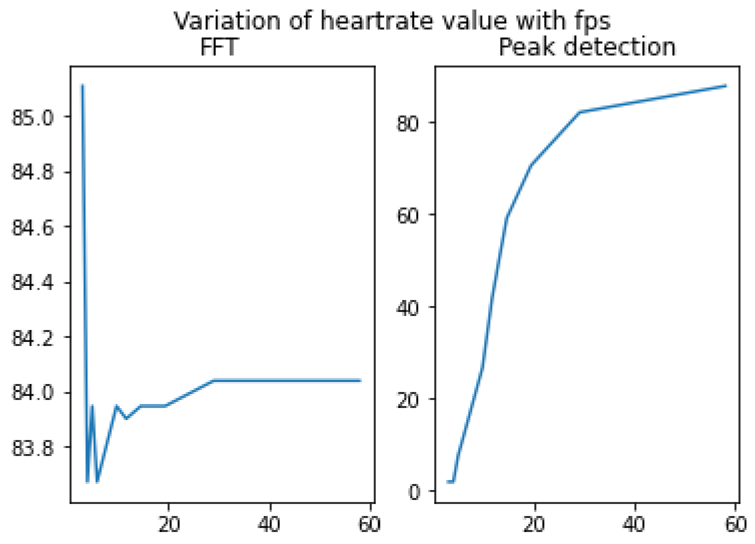
Similar to the second, the third part of the code is for observing variation of calculated value with resolution of the video. For this I used ffmpeg for decreasing resolution of the video to different values beforehand and used these videos to plot the value of heart rate for various resolutions. Here I plotted the results for 7 different values of resolution.

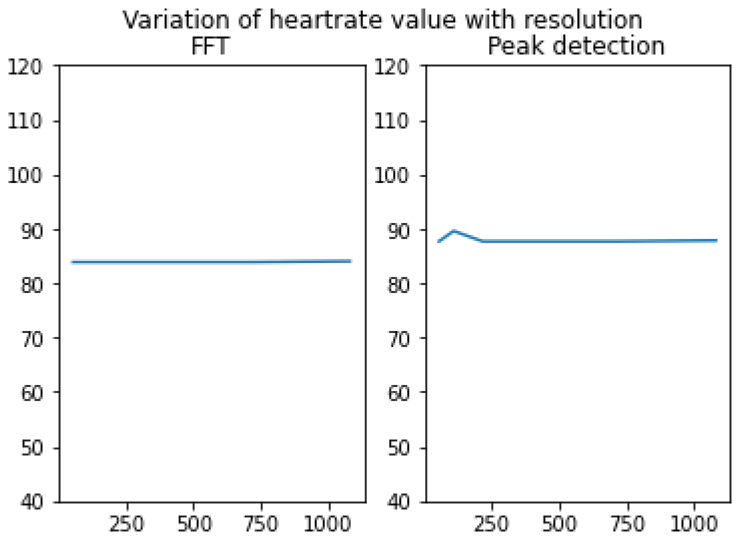
## 3 Data obtained from post-processing videos in the data set

### 3.1 Data sample 1

```
                        Analysis of Data1
Values given by FFT:


108
84
84
84
84
84


Values given by peak detection algorithm:


96
96
92
87
86
88
```

```
Final values:

84,          88
```

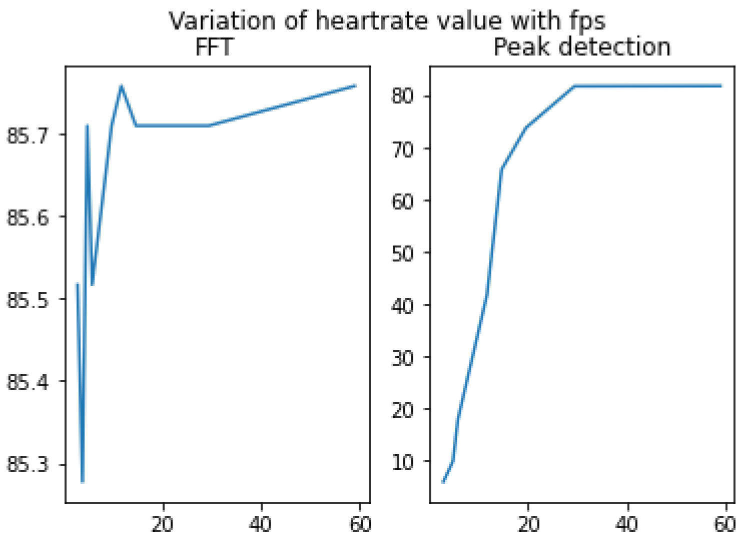


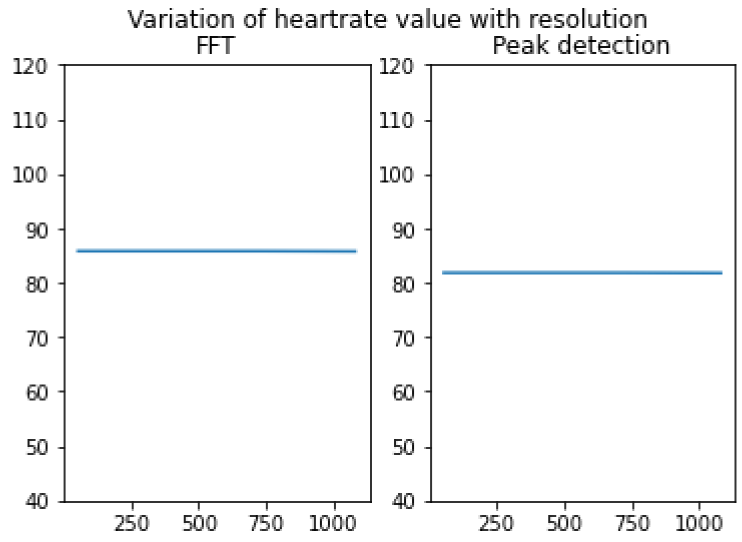


## 3.2 Data sample 2

```
                          Analysis of Data2
Values given by FFT:


72
```

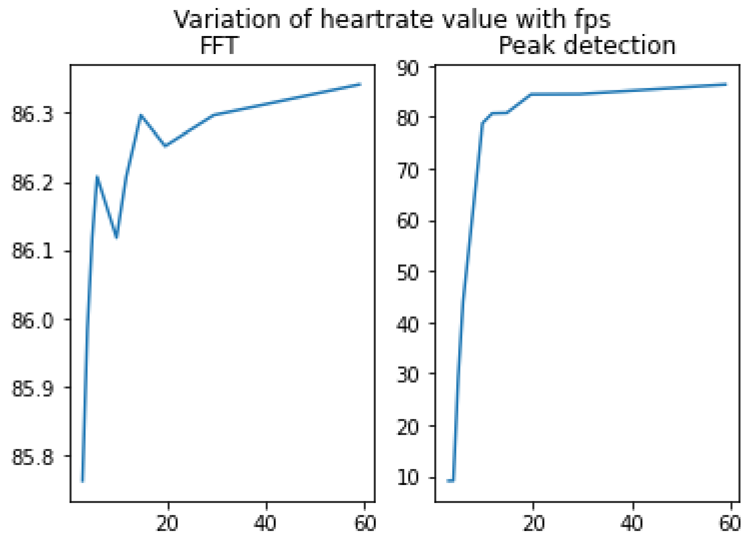
```
78
72
78
86
86


Values given by peak detection algorithm:


72
78
76
78
79
80


Final values:

86,          82
```

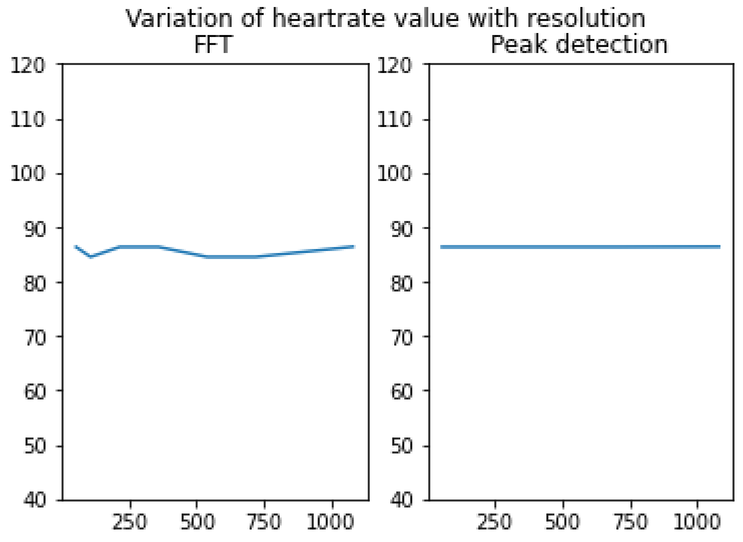




## 3.3 Data sample 3

```
                              Analysis of Data3
Values given by FFT:


84
84
84
84
86
86


Values given by peak detection algorithm:


96
90
88
87
86
86


Final values:

86,          86
```





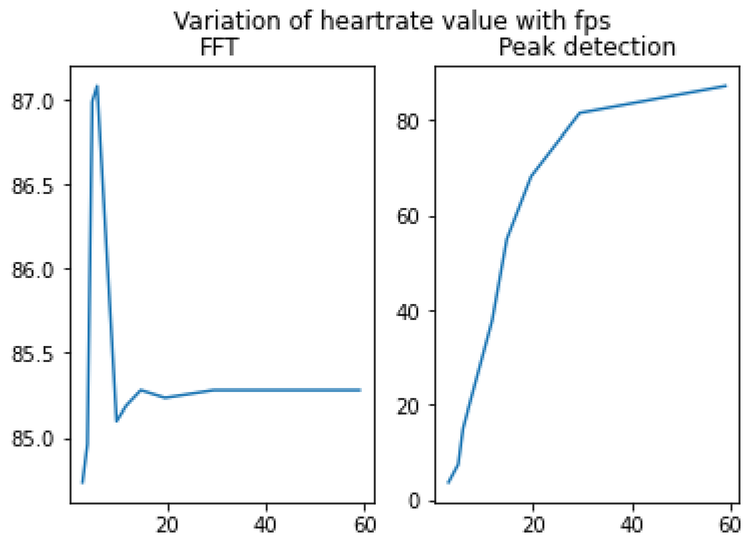
## 3.4 Data sample 4

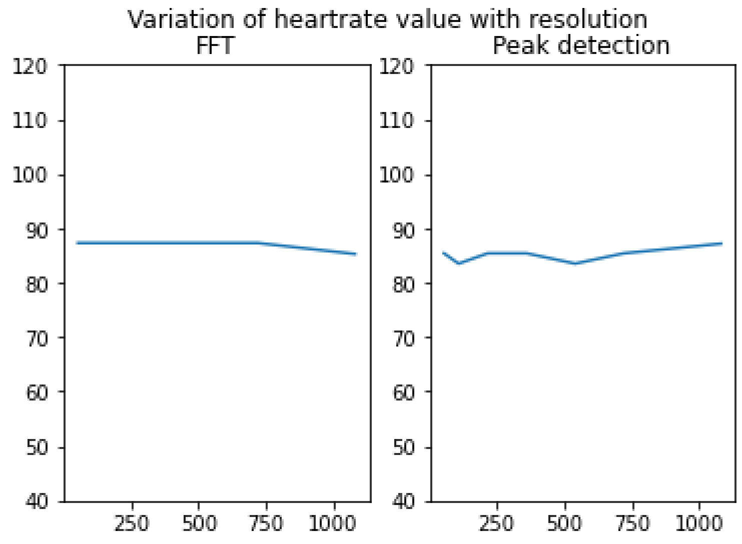
Analysis of Data4

Values given by FFT:

84
84
84
84
84
84

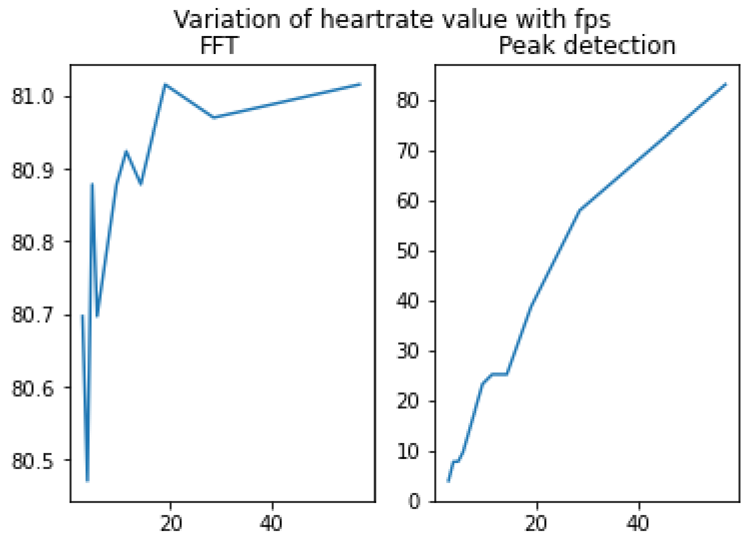
```
Values given by peak detection algorithm:


84
78
88
87
89
88


Final values:

85,          87
```

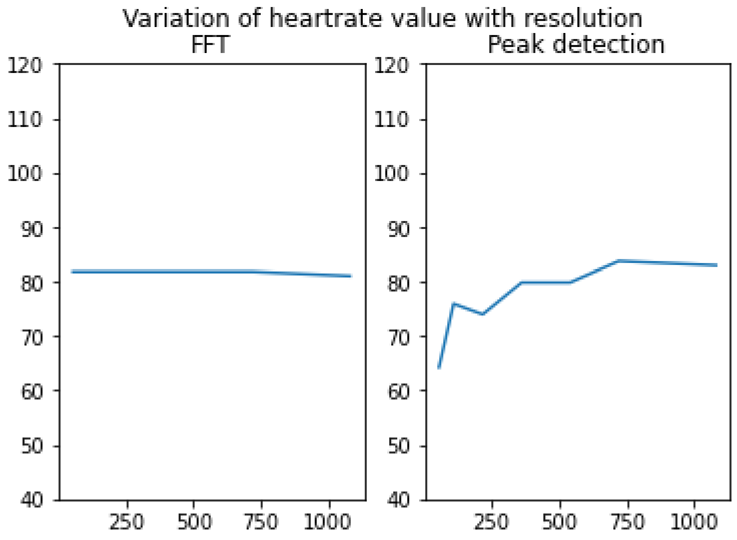




## 3.5 Data sample 5

```
                    Analysis of Data5
Values given by FFT:


84
72
80
81
79
80


Values given by peak detection algorithm:


60
72
80
78
82
80


Final values:

81,          83
```





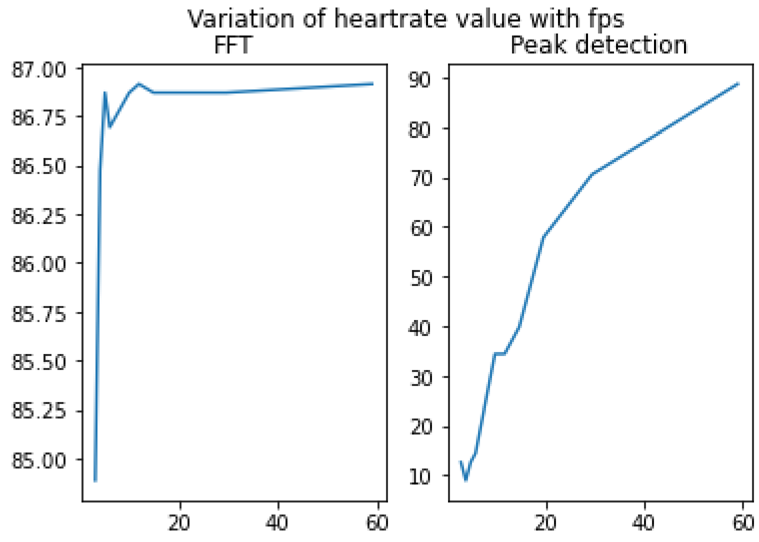
## 3.6 Data sample 6

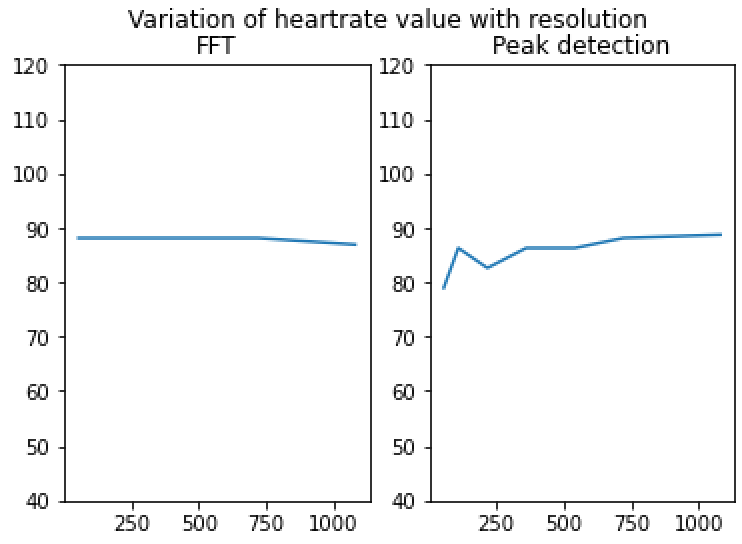
Analysis of Data6

Values given by FFT:

84
90
92
87
86
88

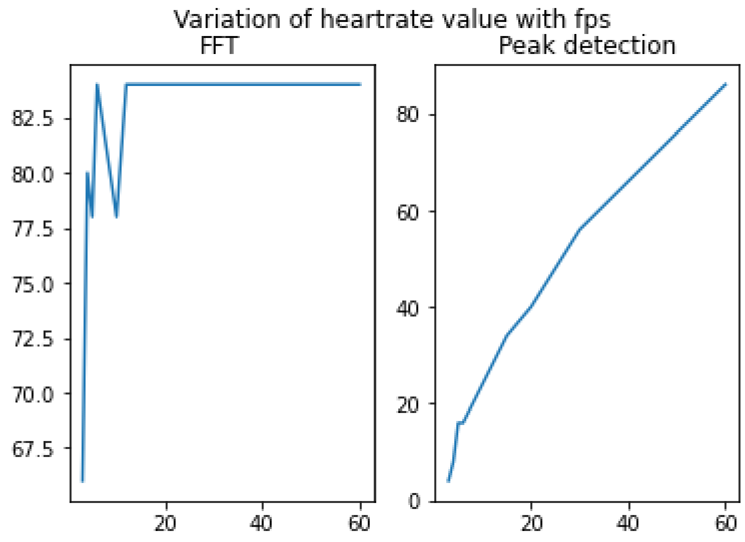
```
Values given by peak detection algorithm:


96
90
84
87
89
90


Final values:

87,          89
```

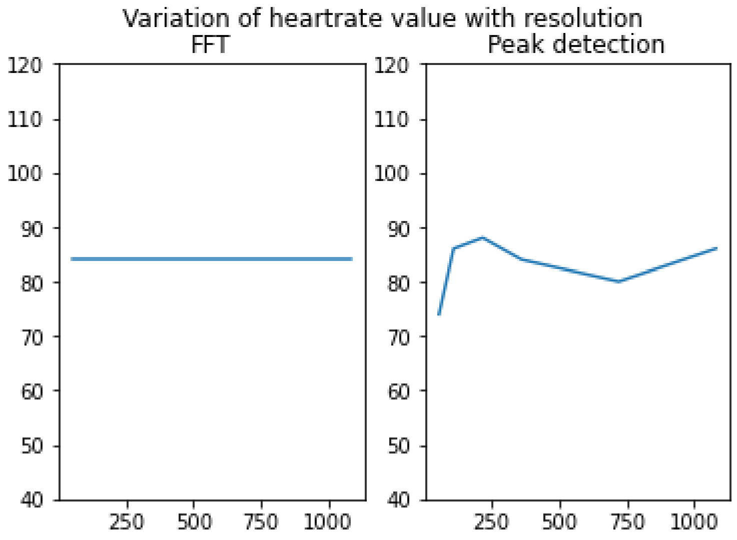




## 3.7 Data sample 7

```
                         Analysis of Data7
Values given by FFT:


72
102
72
81
82
84


Values given by peak detection algorithm:


72
84
88
78
79
86


Final values:

84,         86
```

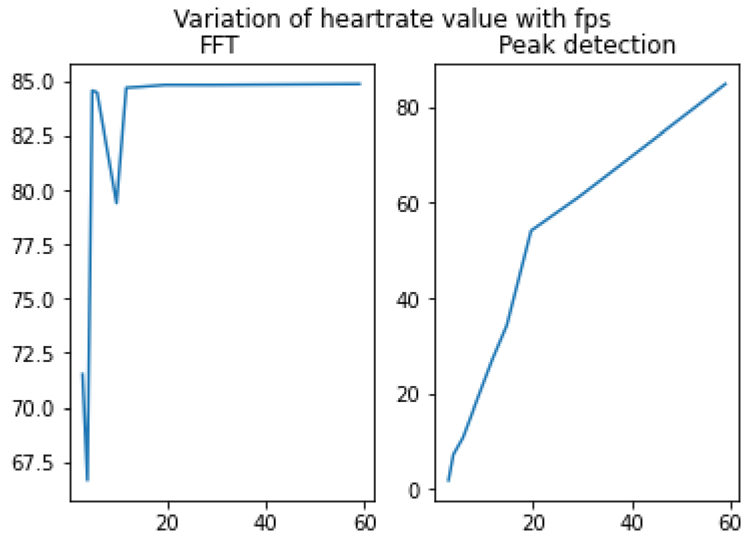


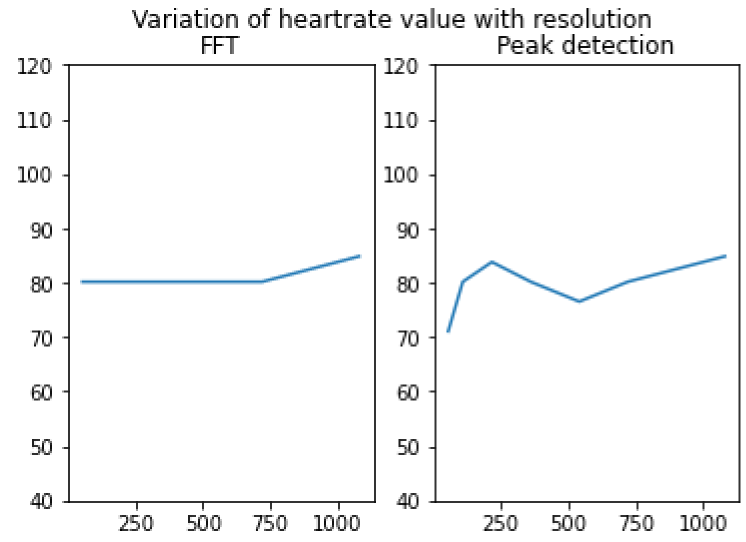


## 3.8 Data sample 8

```
                    Analysis of Data8
Values given by FFT:


84
84
80
75
79
84
```

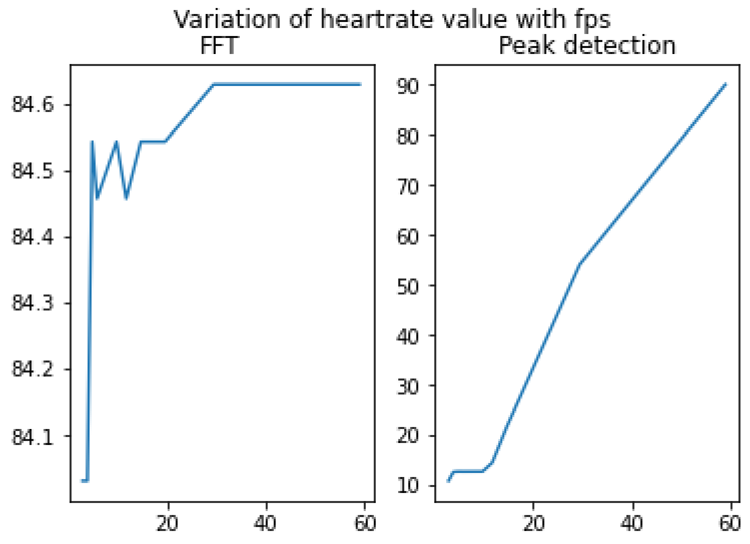
```
Values given by peak detection algorithm:


120
96
92
90
86
84


Final values:

85,          85
```

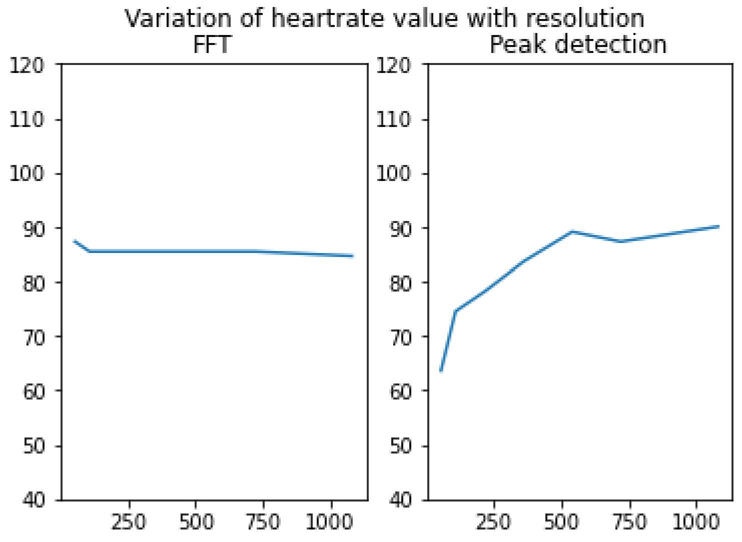




## 3.9 Data sample 9

```
                    Analysis of Data9
Values given by FFT:


84
84
84
84
84
84


Values given by peak detection algorithm:


96
102
100
90
86
90


Final values:

85,          90
```

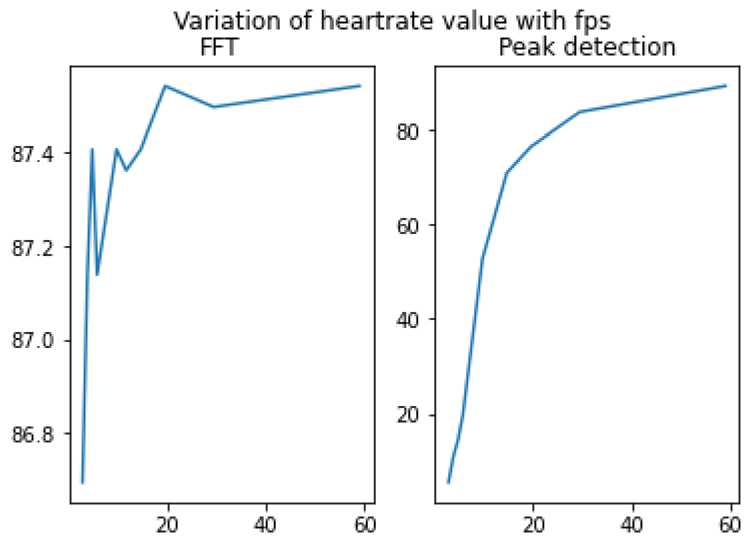


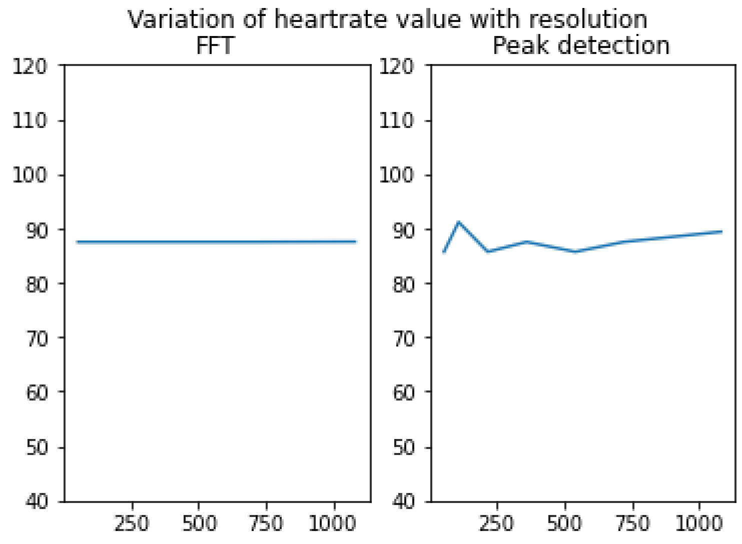


## 3.10 Data sample 10

```
                Analysis of Data10
Values given by FFT:


84
90
88
87
86
86
```

```
Values given by peak detection algorithm:


84
84
88
87
86
90


Final values:

88,         89
```





## 4 Conclusions

In terms of fps, from observing the plots above, we can come down to around 10 fps.Above that it is mostly constant in majority of the cases. This is because, when calculating frequency of some data according to the nyquist theorem, the sample rate must be more than twice that of maximum frequency of the data. The heartrate value in Hz is around 1.3-1.6 Hz. Thus our sample rate must be greater than atleast 3.2 Hz. To be on the safer side it is better to have around atleast 10 Hz. For peak detection algorithm a higher fps is preferred as with lesser fps when there is higher noise accuracy is lesser than in FFT.

The resolution of the video is much more forgiving and remains constant most of the time. Even when resolution is lowered by a factor of 10 there is not a lot of difference in the result. The corners of a video usually contribute to noise as the finger curves and so it can be cropped to increase snr.

Thus even if resolution of the sensor is significantly low it can be compensated by having a good sample rate i.e. video of high fps. However if the fps is significantly low we cannot get a good result even if the resolution is very high.

## 5 References

https://www.geeksforgeeks.org/extract-images-from-video-in-python/
https://www.geeksforgeeks.org/python-pil-getpixel-method/
https://stackoverflow.com/questions/43111029/how-to-find-the-average-colour-of-an-image-in-python-with-opencv
https://realpython.com/python-scipy-fft/

https://stackoverflow.com/questions/23308578/how-to-find-numpy-argmax-on-part-of-list-and-save-index
https://stackoverflow.com/questions/25735153/plotting-a-fast-fourier-transform-in-python/25735274
https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/
https://swharden.com/blog/2020-09-23-signal-filtering-in-python/