Mansoura university

Faculty of Engineering

Electronics and

Communications Department

Heart rate measurement using webcam

A graduation project is submitted to the electronics and communication Engineering Department in partial fulfillment of the requirements for the degree of Bachelor in electronics and communication Engineering

Supervised by:

Assist. Prof. Ahmed Abd El-Rahman Elnakib

Heart rate measurement using webcam

A graduation project proposed by:

1) Amany Mouawd Abd El –Aziz.
2) Aml Mahmoud Abd El-Maboud.
3) Eman Hosny Mahmoud.
4) Eman Salah Mohamed Helm.
5) Eman Mahmoud El-sayed.
6) Eman Nagy Abo El-soaud.
7) Taghrieed Gad-Allah Refit.

Table of Contents

LIST OF ABBREVIATIONS

ECG	Electrocardiogram
bpm	beat per minute
IIR	Infinite Impulse Response
FIR	Finite Impulse Response
DFT	Discrete Fourier Transform
FFT	The Fast Fourier Transform
CRF	Complexity Reduction Factor
GUIs	Graphical User Interfaces or UIs
GUIDE	GUI Development Environment

List of Figures

Chapter (2)

Software Method

2-1 Introduction

The human visual system has limited spatiotemporal sensitivity, but many signals that fall below this capacity can be informative. For example, human skin color varies slightly with blood circulation. This variation, while invisible to the naked eye, can be exploited to extract pulse rate. **Fig (2.1)** illustrates these variations.

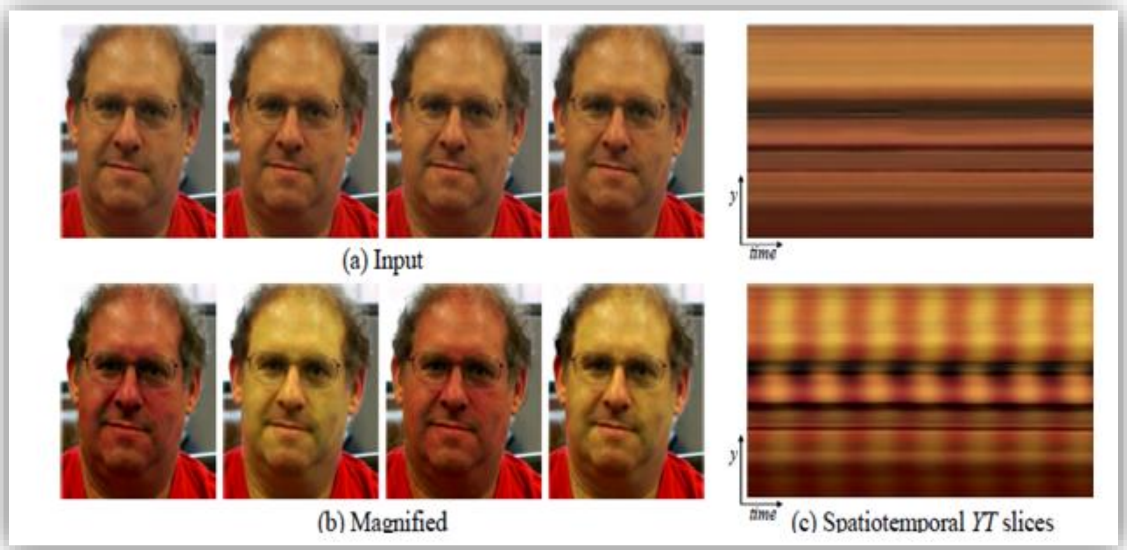


Fig.(2.1): An example of using our Eulerian Video Magnification framework for `visualizing the human pulse. (a) Four frames from the original video sequence (face). (b) The same four frames with the subject’s pulse signal amplified. (c) A vertical scan line from the input (top) and output (bottom) videos plotted over time shows how our method amplifies the periodic color variation . In the input sequence the signal is imperceptible, but in the magnified sequence the variation is clear. The complete sequence is available in the supplemental video.

2.2 Method of work

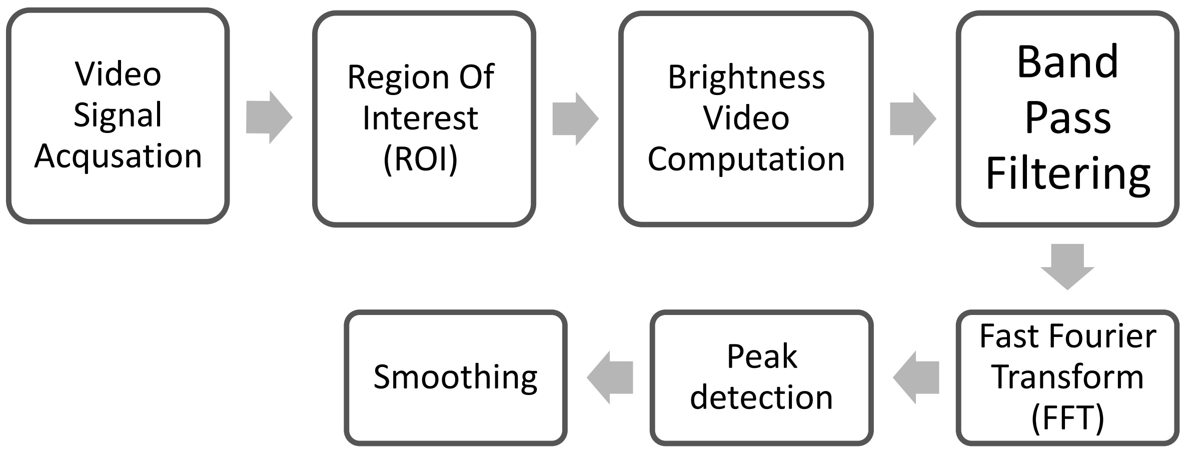


Fig.(2.2): System diagram of method

2.2.1 video signal acquisition and ROI

The first thing to take into account when it comes to sampling a signal is bandwidth. The normal human heartbeat is between 60 and 200 beats per minute (bpm), depending on age, fitness condition and the physical activity that the subject is doing. In our experiment, we have generously assumed that the target signal can be found between 40 and 230 bpm, i.e. 0.667 and 3.833 Hz. According to Nyquist, the sampling frequency should be at least twice the highest value (7.667 Hz) to catch the whole range of heartbeat frequencies without aliasing. The videos are recorded at 24 or 30 frames per second depending on the shot.

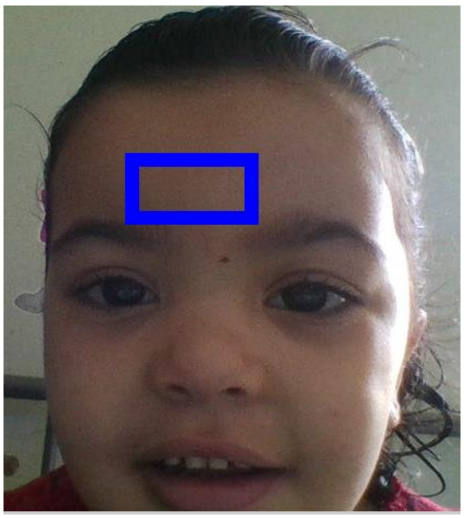


Fig.(2.3): Video Signal Acquisition and ROI.

2.2.2Brighteness signal computation

The signal we want to process is the brightness of the skin over time. We cannot ensure that all pixels in the image will contain the brightness variation that we are looking for and we want the rest of the processing pipeline to stay computationally light. Therefore, we chose to combine all pixels into a single average brightness value per frame. On the other hand, as I am thinking about implementing the algorithm in real time, we have skipped the common image brightness computation -combining the red, green and blue planes- in favor of a simple average of all the pixels in the red plane. This is computationally much cheaper and gives very similar results because almost all the image energy is in the red plane. So, the *n*- the sample of the red brightness function can be expressed as:

$$\boldsymbol{b}[\boldsymbol{n}] = \frac{1}{W.H}\sum_{X}^{H}\sum_{V}^{W} v[n, x, y, 1] \qquad Equ.(2.1)$$

Where:
W : width of the image in pixels

H : height of the image in pixels
v [n, x, y, 1] : light level of the red plane (index 1) at [x, y] coordinates of frame n in the video signal.

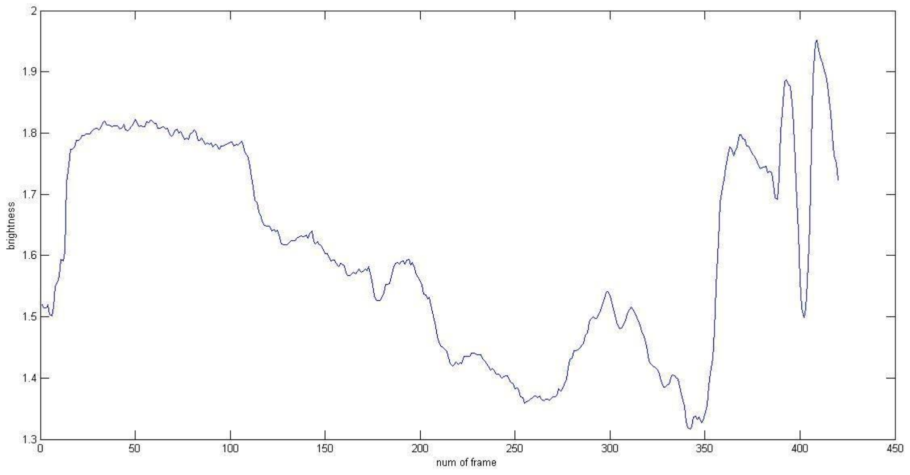


Fig. (2.4): Brightness Video Computation

2.2.3 Band pass filtering

After the signal acquisition, a band-pass filter attenuates frequencies outside the interest band. This reduces the noise in later processing steps (peak fine-tuning) and makes the resulting heart rate signal smoother. For

our case, a second-order Butterworth filter is designed. The cutoff frequencies have been set to contain our band of interest: 40-230 bpm .
Notice how an initial piece of 1 seconds is cut off the filtered signal. It is an approximation of the time it takes for the filter to completely remove the constant signal offset (Fig2.3). If this initial piece is not removed, we might get bad readings of the heart beat during the signal stabilization time.

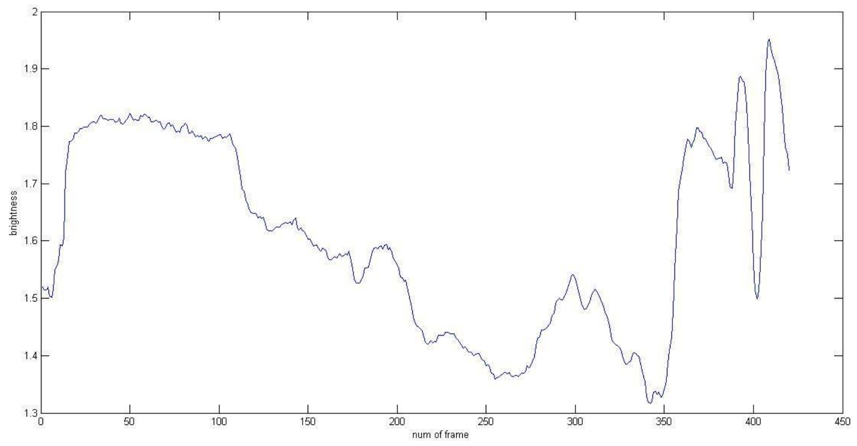


Fig.(2.5): Band Pass filtered signal.

<u>Other filter types can be tested, and We chose Butterworth as:</u>

1. It is an IIR filter and the order required for a given bandwidth is much lower than with a FIR filter. Lower order usually means less computations.
2. It has flat pass-band and stop-bands compared to other IIR structures that show ripples. This avoids favoring certain frequencies over others in the valid range.

2.2.4 Sliding window:

In order to give a continuous estimation of the heart rate, the FFT and the following two steps (peak detection and smoothing) are repeated every 0.5 seconds. This computation is always performed over a window containing the last 6 seconds of signal samples. Virtually, the window moves over the signal and this is why it is called "sliding" or "moving" window. The 6-second length is not arbitrary. The window length directly affects frequency resolution and, thus, the accuracy of our estimation. The FFT of a signal sampled *N* times at a sampling frequency F_s is *N* bins long. All the bins together cover a bandwidth of F_s . So, the frequency difference between two consecutive bins is F_s / *N*. This is the frequency resolution F_r. As the sampling frequency can be written as the number of window samples divided by the total time it took to sample them (the window time duration), we can say that:

$$\boldsymbol{Fr = \frac{Fs}{N} = \frac{\frac{N}{Tw}}{N} = \frac{1}{Tw}} \qquad Equ. (2.2)$$

Where:
F_r : frequency resolution
F_s: sampling frequency
N : number of window samples
T_w : window time duration

Therefore, the higher the window duration, the better the frequency resolution. The accuracy will be better, too, as it is half the resolution in this case .However, increasing the window duration decreases the time accuracy. Think about the trivial case in which the whole signal length is picked as the window length. If a peak is detected in the FFT, it is impossible to tell when that tone started within the signal or how long it lasted. Whatever number we give will be a maximum of a window length away from the real value. Another problem of a long window is that it will force the user to wait for an equally long period to get a first reading after starting up the measurements. In summary, with a 6-second window, we get a tolerable 6-second startup delay that gives a fair time accuracy of 6 seconds and a fair frequency accuracy of 5 bpm (half the FFT resolution). All in all, it looks like a good trade-off.

*Why do we move the window in 0.5-second steps?

Computing an estimate every 0.5 seconds does not improve the time accuracy of the output, but it increases the time resolution of the reading. It produces more heart rate output samples per second that will be later smoothed to provide a more continuous and frequent reading. With this, we are incrementing the time resolution of the reading but not its accuracy, which stays limited by the time resolution of the FFT.

2.2.4.1 Fast Fourier transform

The Discrete Fourier Transform (DFT) is used to translate the signal from the time domain to the frequency domain. The Fast Fourier Transform (FFT) algorithm was used to save processing time when computing the DFT. While the computational complexity of the DFT is O(N^2) for a set of N points, the FFT gets the same results with O($N \cdot \log_2(N)$), which means a huge speed-up when N is high. The FFT of a real signal is a complex signal in which each complex sample represents the magnitude and the phase of the corresponding frequency. In our case, the phase is not needed. The FFT magnitude is easily computed.

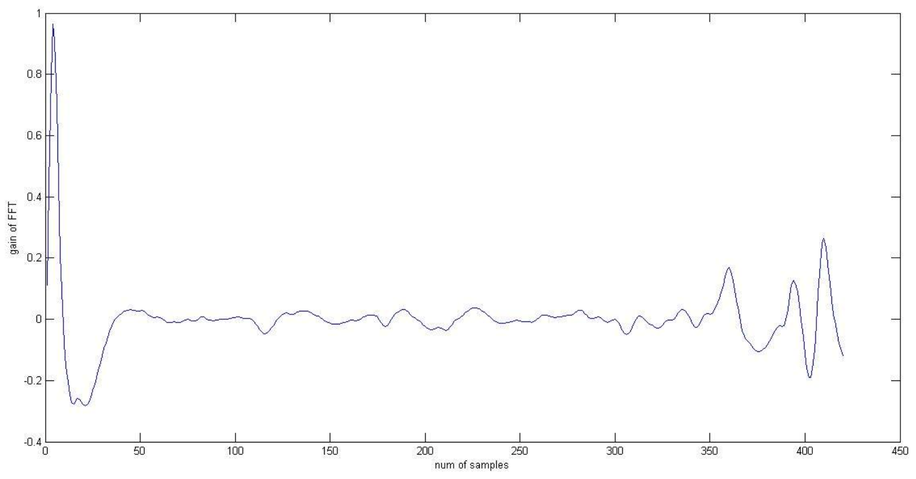


Fig.(2.6): output of Fast Fourier Transform (FFT).

2.2.4.2 Peak detection

Once the FFT is computed for the current sliding window contents, magnitude peaks in the interest band are spotted thanks to the find peaks function in Matlab . A sample is taken as a peak if it is either larger than its two neighbors or equal to infinity. Among the resulting peaks, the highest peak position is sought with the max function.

Finally, it is translated to the corresponding frequency in the FFT vector.

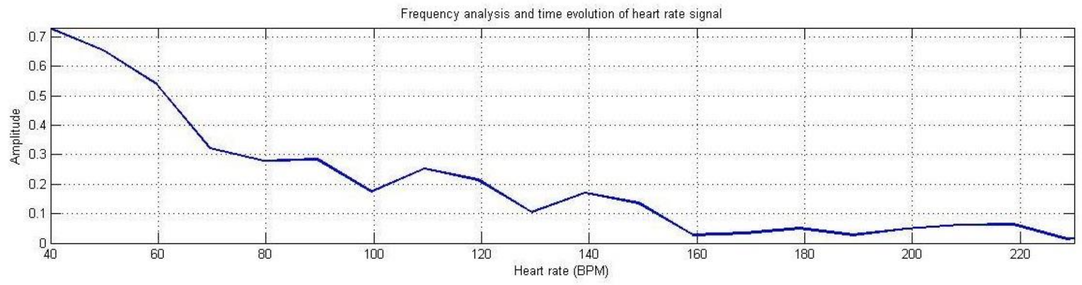


Fig. (2.7): The peak detection

2.2.4.3 Smoothing

At this stage, an approximate location for the most powerful tone in the frequency band has been found, but the possible outcomes are a discrete set in 10 bpm increments because of the frequency resolution produced by the 6-second window. We would like that the heart rate readings look more continuous, with 1 bpm frequency resolution instead. To achieve this, the signal window is correlated with a series of tones in phase and quadrature around the FFT peak in 1 bpm increments. The tones lie in the uncertainty interval around the peak caused by the FFT frequency resolution. The result of each signal-tone correlation is a complex number representing a phase-magnitude pair. The frequency that corresponds to the highest magnitude is taken as the smoothed heart rate:

$$\mathrm{C}k \sum_{n=0}^{N-1} b_n e^{j2\pi n}\left(f_p - 0.5F_r + kF'_I\right)/F_s \quad ,$$

$$\mathbf{U}\left\{0, \dots , \left[\frac{F_r}{F'_r}\right]\right\}, k \in$$

$$\mathbf{HR} = f_p - 0.5F_r + F'_I \underset{\mathrm{k}\in\mathrm{U}}{\arg\max} |c_k| \qquad Equ. (2.3)$$

Where:
HR : Smoothed heart rate
b_k : k-th sample of brightness signal
N : Window length in samples
f_p : FFT peak frequency

F_r : FFT frequency resolution
F_r' : Smoothing frequency resolution

At first, we came up with this method intuitively, thinking of the correlation as a measure of similarity. our goal was to find the tone most similar to the signal in a frequency range around the FFT peak, regardless of the phase, by measuring the similarity (correlation) between the signal and a series of reference complex tones. In fact, it is what the FFT does, but using only orthonormal tones, whose period is an integer number of samples. Since the FFT frequencies are orthonormal, they constitute a base in which all signals can be expressed by linear combination. Correlating against other intermediate frequencies does not give more information because they themselves are formed by linearly combining the orthonormal ones. This is the reason why the result of this method is just smoothed FFT data, instead of higher accuracy extra information.

Later, we realized that this method is equivalent to computing the FFT on a zero-padded version of the signal, which is a commonly used technique to smooth the FFT data, and picking only a frequency range. If the original 6-second signal in the window is zero-padded up to 60 seconds, the tone frequencies used in this method are found among the orthonormal frequencies of the 60-second FFT. Then, the method is equivalent to applying the DFT definition on the zero-padded window for certain frequencies only. The advantage of this "zero-padding partial DFT" over the "zero-padding FFT" is speed. The FFT computes the result as a whole and cannot be divided to get sub results in a continuous frequency range; therefore, its complexity stays at O($N_z \cdot \log_2(N_z)$), where N_z is the length of the zero-padded window. On the other hand, the complexity of the proposed method is O($S \cdot N$), where N is the window length and S is the number of test tones. So, the complexity reduction factor (CRF) is:

$$\left.\begin{array}{l} Fr = Fs/N \\ Fr' = Fs/Nz \\ Fr = Fs/N \end{array}\right] \longrightarrow \quad S = Nz/N \longrightarrow S.N = Nz$$

$$CRF = Nz.log_2(Nz)/S.N=$$

$$Nzlog_2(Nz)\,/Nz{=}log_2(Nz) = log_2 FsFr' \qquad Equ.(2.4)$$

CRF : Complexity Reduction Factor of the partial DFT method over the FFT on the zero-padded signal
F_s : Sampling frequency
F_r : Frequency resolution before smoothing
F_r' : Desired frequency resolution after smoothing
N : Window length.
N_z : Zero-padded window length
S : Number of test tones

For a given sampling frequency, the higher the new frequency resolution, the more beneficial will be using the "zero-padding partial DFT" instead of the "zero-padding FFT". In our case, $F_r' = 1/60$ Hz and $F_s = 24$ Hz in the worst case. So, the *CRF* is approximately 10.5, which means that -roughly speaking- the time taken by the proposed method will be something in the order of a 10% of the time taken by the FFT on the zero-padded window, and the final heart rate as shown in (**Fig. 2.6**).

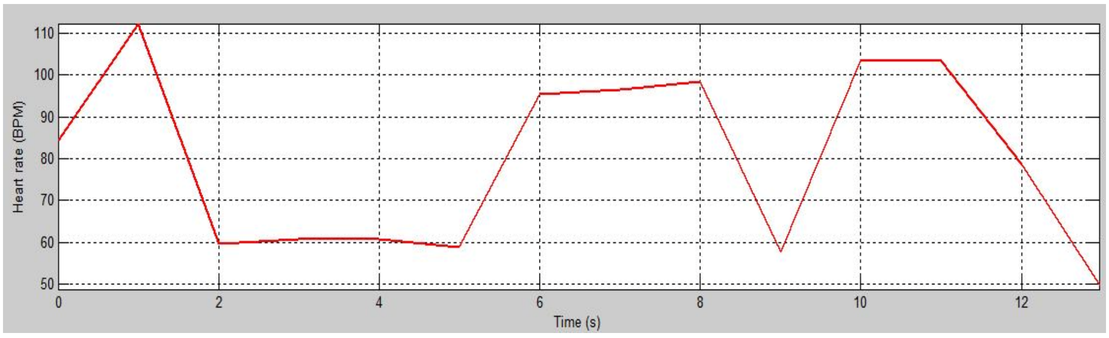


Fig. (2.8): Smoothing (final heart rate).

Chapter (3)

GUI (illustration)

3-1 Introduction

GUIs (also known as graphical user interfaces or UIs) provide point-and-click control of software applications, eliminating the need to learn a language or type commands in order to run the application.

MATLAB apps are self-contained MATLAB programs with GUI front ends that automate a task or calculation. The GUI typically contains controls such as: menus, toolbars, buttons, and sliders (as shown in **Fig. 3.1**).

Many MATLAB products, such as Curve Fitting Toolbox, Signal Processing Toolbox, and Control System Toolbox, include apps with custom user interfaces. You can also create your own custom apps, including their corresponding UIs, for others to use.

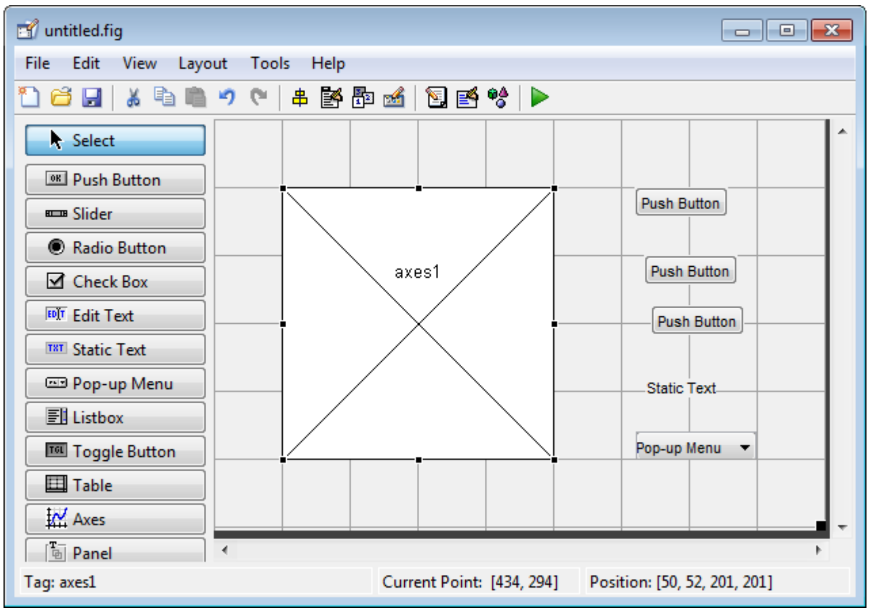


Fig.(3.1) : Components of GUI

3-2 Overview of GUI Design and the Supplied Development Tools:

- Since MATLAB is a cross-platform software package (primary platforms include Win95/NT, Unix, and Macintosh) the GUI components are derived from those in common to all the supported platforms.

- In MATLAB 4 GUI design required hand coding of GUI components; some add-on tools suchasGUIMaker1were available later.

- With MATLAB 5 a multipart GUI building tool named Guide, which stands for Graphical User Interface Development Environment, was introduced (Guide is also included in student edition.

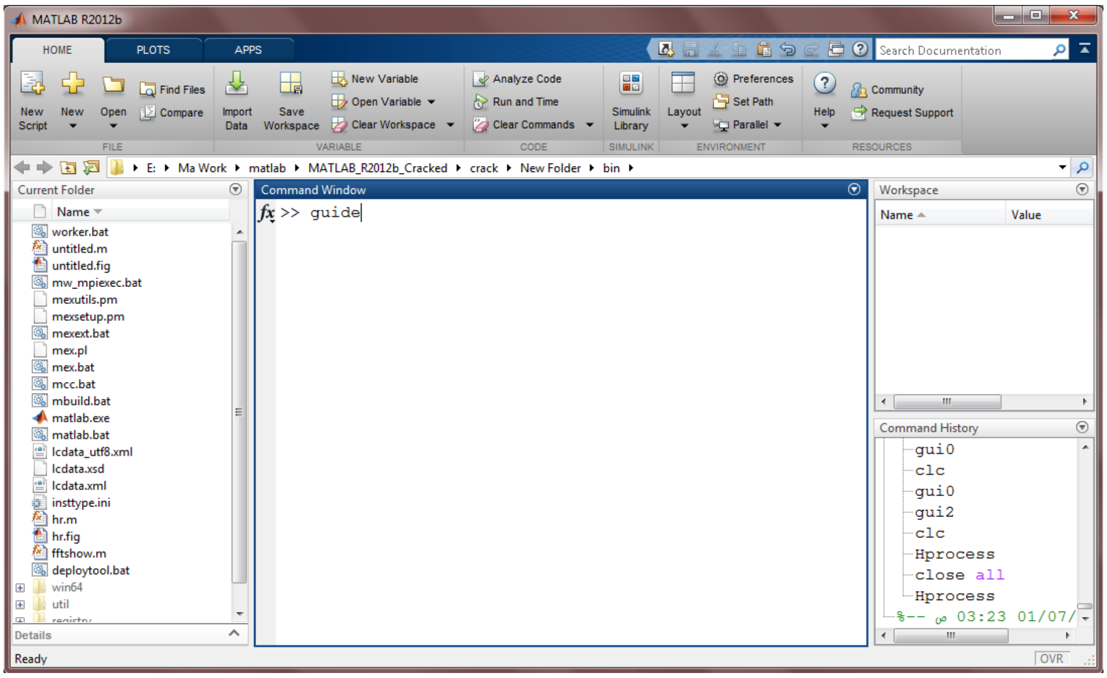
- Developing GUI’s on Matlab© 6 is a breeze and hopefully this tutorial will be sufficient . running an older version of Matlab© , this tutorial will help you get started however it will not be able to guide you all the way. I would recommend a migration to Matlab 6 as it as a more stable and more mature platform where many of the bugs, especially in Matlab's ability to handle graphical objects have been addressed.

3-3How to Build MATLAB GUI ?

Matlab GUI can be built using GUIDE (GUI Development Environment), then coding each part of the GUI from MATLAB editor. Below, we will illustrate how to build the Matlab GUI.

3-3-1 Initializing GUIDE (GUI Creator): GUIDE (graphical user interface design environment) provides tools for designing user interfaces for custom applications. Using the GUIDE Layout Editor, you can graphically design your UI. GUIDE then automatically generates the MATLAB code for constructing the UI, which you can modify to program the behavior of your application. To use the GUIDE, you should follow the following processing steps:

1. Open up MATLAB. Go to the command window and type in "guide" and click "Enter". (see **Fig. 3.2**)



```
>> guide
```

```
gui0
clc
gui0
gui2
clc
Hprocess
close all
Hprocess
%-- م 03:23 01/07/
```

Fig.(3.2): Guide in command window

2. Choose the first option Blank GUI "Default" . (as shown in **Fig. 3.3**)

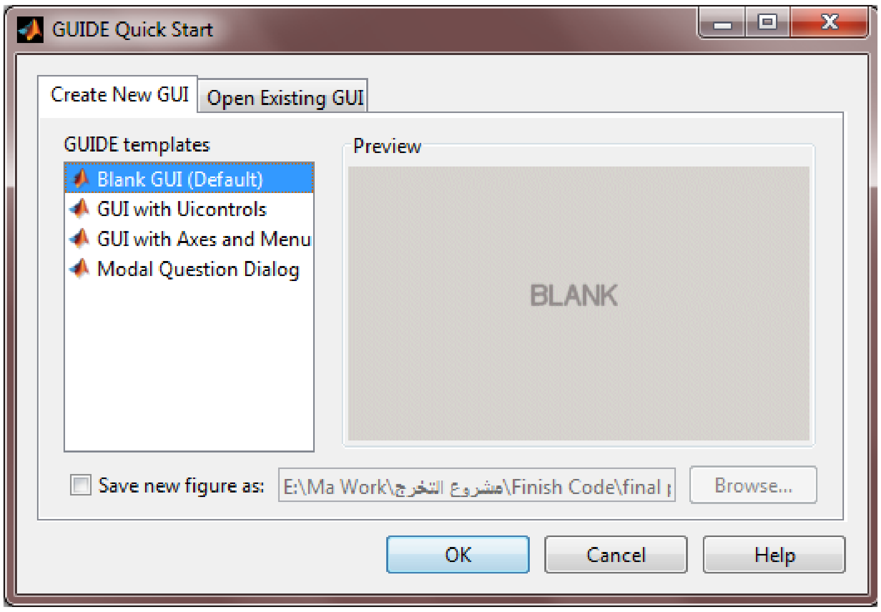


Fig.(3.3): Blank GUI default

3-Click on the "ok" button in the left hand side of the window. This will allow you to drag and drop any button. (see **Fig. 3.4**)

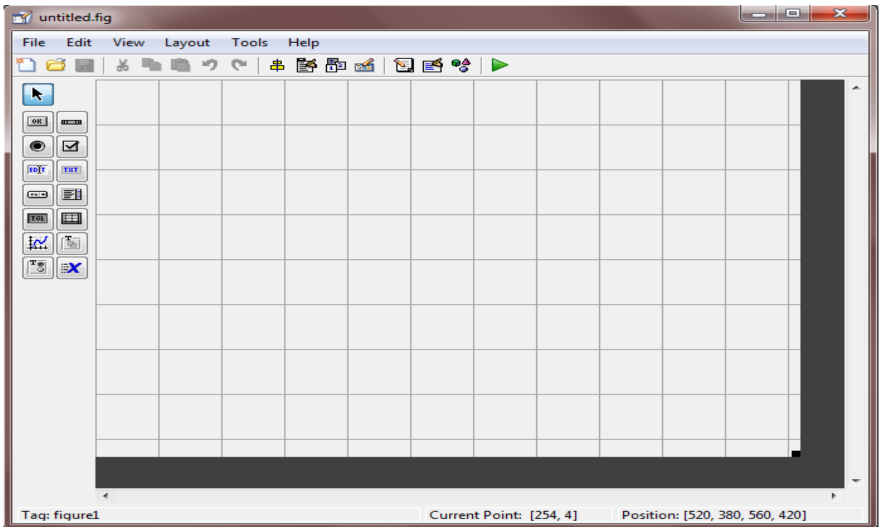

untitled.fig
File Edit View Layout Tools Help
Tag: figure1
Current Point: [254, 4]
Position: [520, 380, 560, 420]

fig.(3.4): Window of untitled1

4-Simply drag and drop items from the left 2 columns into the grid area to create your GUI . The different types of components are as in **Fig.(3.1)**.

5-Click once and hold down the button and drag your mouse until the square that this form is of the size you'd like.

6-Release the mouse button and you will see your push button appear as shown in (**Fig. 3.5**).

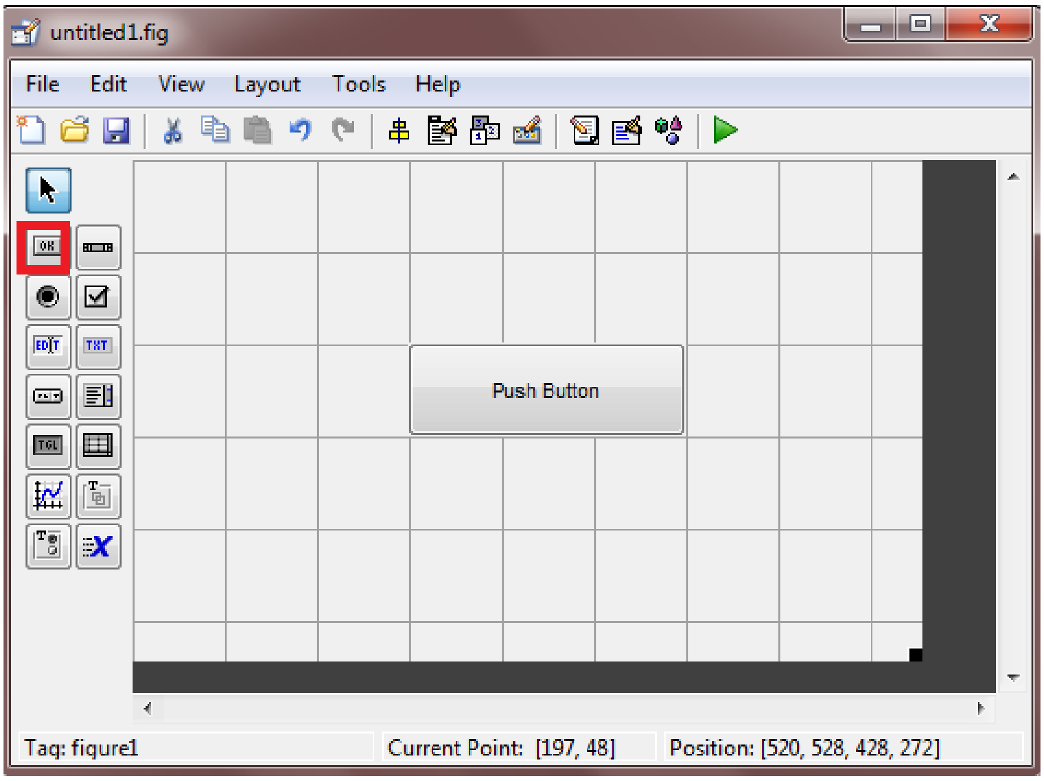
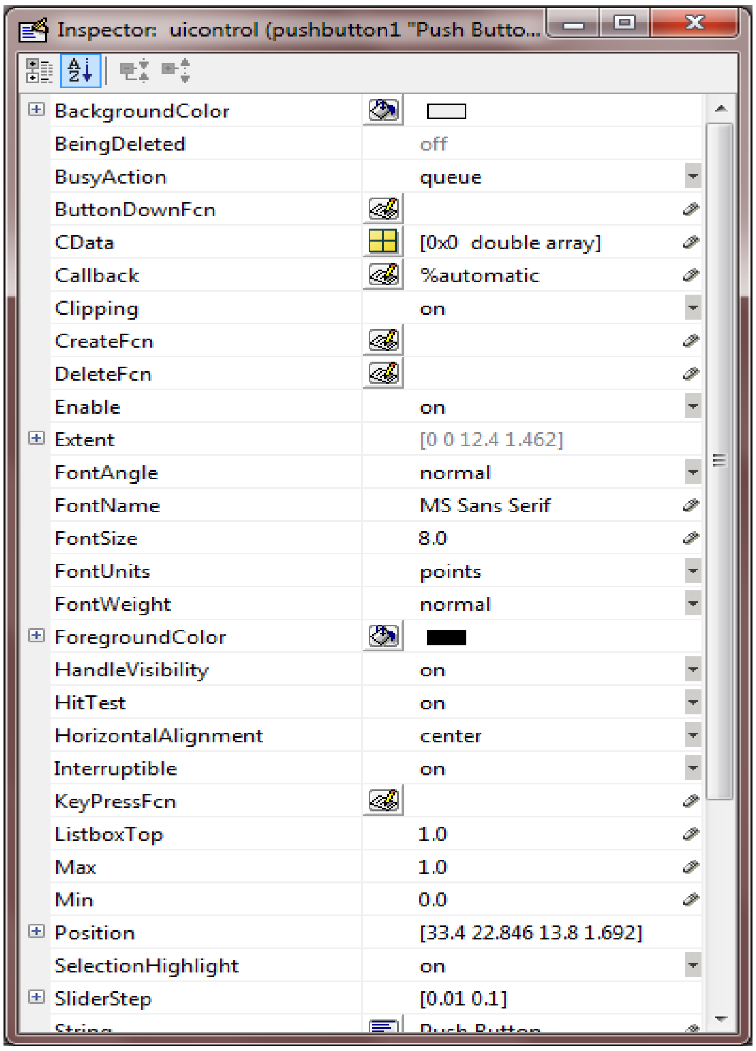

untitled1.fig
File Edit View Layout Tools Help
Push Button
Tag: figure1
Current Point: [197, 48]
Position: [520, 528, 428, 272]

Fig.(3.5): Example to show push button in the area

7-Double click on the pushbutton you just created. A property manager will pop up. (see **Fig. 3.6**)



Inspector: uicontrol (pushbutton1 "Push Butto...

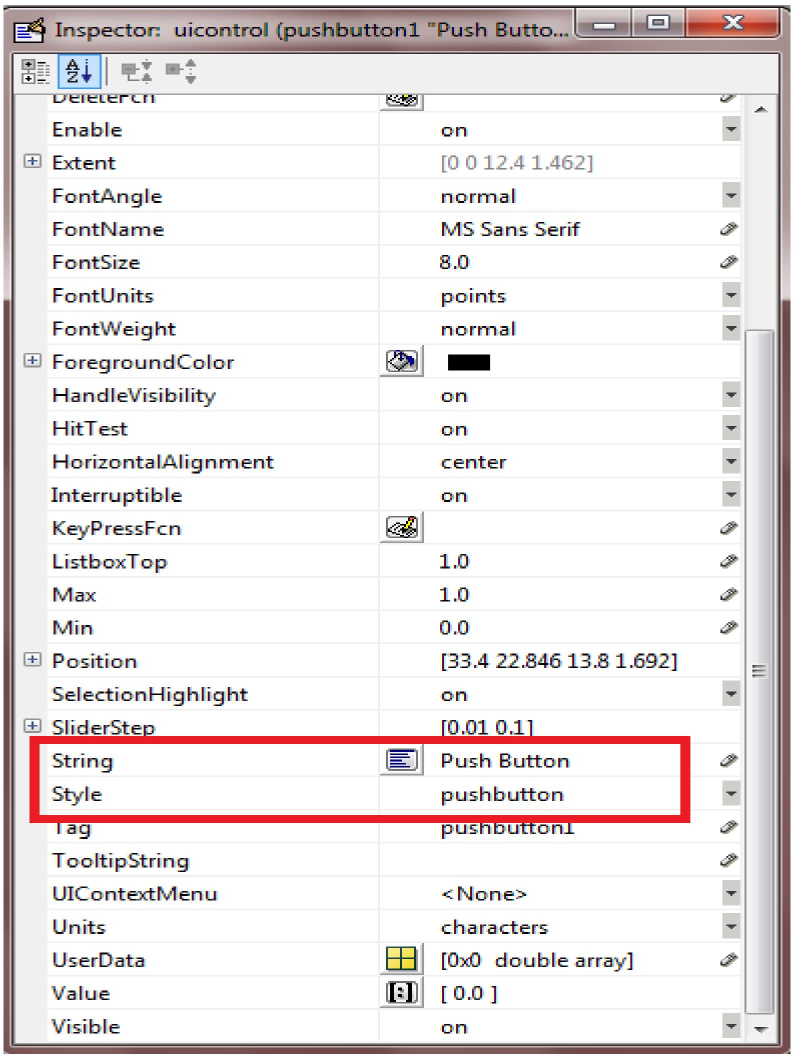
BackgroundColor	⬜
BeingDeleted	off
BusyAction	queue
ButtonDownFcn	
CData	[0x0 double array]
Callback	%automatic
Clipping	on
CreateFcn	
DeleteFcn	
Enable	on
Extent	[0 0 12.4 1.462]
FontAngle	normal
FontName	MS Sans Serif
FontSize	8.0
FontUnits	points
FontWeight	normal
ForegroundColor	⬛
HandleVisibility	on
HitTest	on
HorizontalAlignment	center
Interruptible	on
KeyPressFcn	
ListboxTop	1.0
Max	1.0
Min	0.0
Position	[33.4 22.846 13.8 1.692]
SelectionHighlight	on
SliderStep	[0.01 0.1]
String	Push Button

Fig.(3.6): Properties of push button

3-3-2 Coding from MATLAB editor

The GUI is not yet saved, so at the top of the window it is shown untitled.fig. Once the work is saved the title of the GUI will be reflected here. *.fig is the extension of GUI figure files. Generally, a GUI requires two files the figure (*.fig) files where various components are aligned and the code (*.m) files where the coding is done.

1-Change the name of properties (style &string)with the name you want as shown in (**Fig. 3.7**).


Inspector: uicontrol (pushbutton1 "Push Butto...

DeleteFcn	
Enable	on
Extent	[0 0 12.4 1.462]
FontAngle	normal
FontName	MS Sans Serif
FontSize	8.0
FontUnits	points
FontWeight	normal
ForegroundColor	■
HandleVisibility	on
HitTest	on
HorizontalAlignment	center
Interruptible	on
KeyPressFcn	
ListboxTop	1.0
Max	1.0
Min	0.0
Position	[33.4 22.846 13.8 1.692]
SelectionHighlight	on
SliderStep	[0.01 0.1]
String	Push Button
Style	pushbutton
Tag	pushbutton1
TooltipString	
UIContextMenu	<None>
Units	characters
UserData	[0x0 double array]
Value	[0.0]
Visible	on

Fig.(3.7): Change the name of string and style.

2-Now click on file then save to save your work. This will also pop up the code for your program.

3-Locate the line of code in the code editor that says function varargout= pushbutton1_Callback(hobject , evendata , handles, varargin). This is the callback function.
Any code below this will be executed whenever the user pushes the button. Here we will make this change the text in the text box when the user clicks on the button.

3-4 Our Complete GUI Design:

Our GUI has the following features :

- ❑ Fully automated
- ❑ Start by Capturing the video
- ❑ End with the heart rate measurement

The complete GUI consists of three slide pages:

1. Start/master Page

Contains our Logo, and start button that manage user to transfer to the next page.

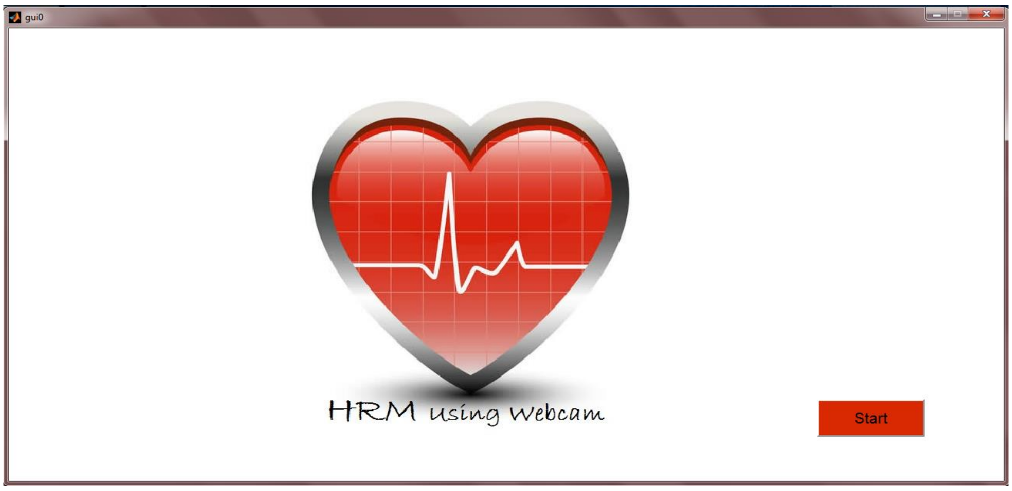


Fig.(3.8): start page handle using GUI

2. First page (capturing the video):

As soon as pressing the "REC" push button ,the video begins recorded ,and to change number of frames use the Edit text "FPS of recording" and enter the desired number of frams; also can change name of video through the "file mane" Edit text.Then the video will be saved in "Heart Rate Measurment Using Webcam" folder that is created automatically while run the code. The "Next" and "Previous" Push buttons are using to move to start/master and second GUI pages respectively. (see **Fig.3.9**).

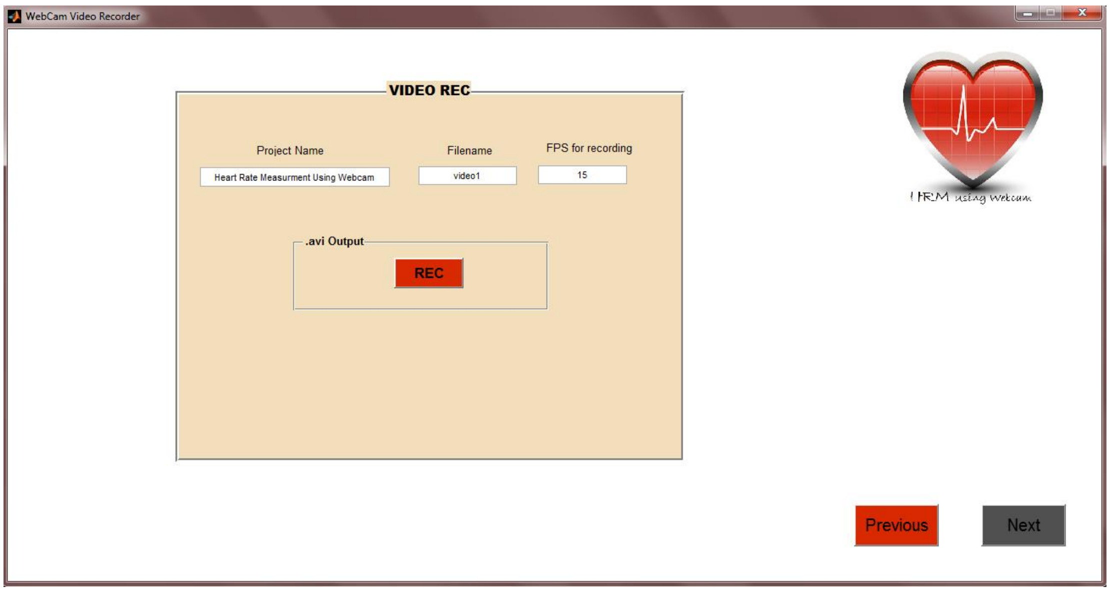

Project Name: Heart Rate Measurment Using Webcam
Filename: video1
FPS for recording: 15

Fig.(3.9): first page handle using GUI.

3. Second page (the heart rate measurement) :

It is the GUI handle for the heart rate measurement code . the "RUN" push button -multifunction button- is used to run the code, plot the human's heart rate, and also display the human's heart beat per minute

Also "Next" and "Previous" push buttons are using to move to the start/master and first slide pages.

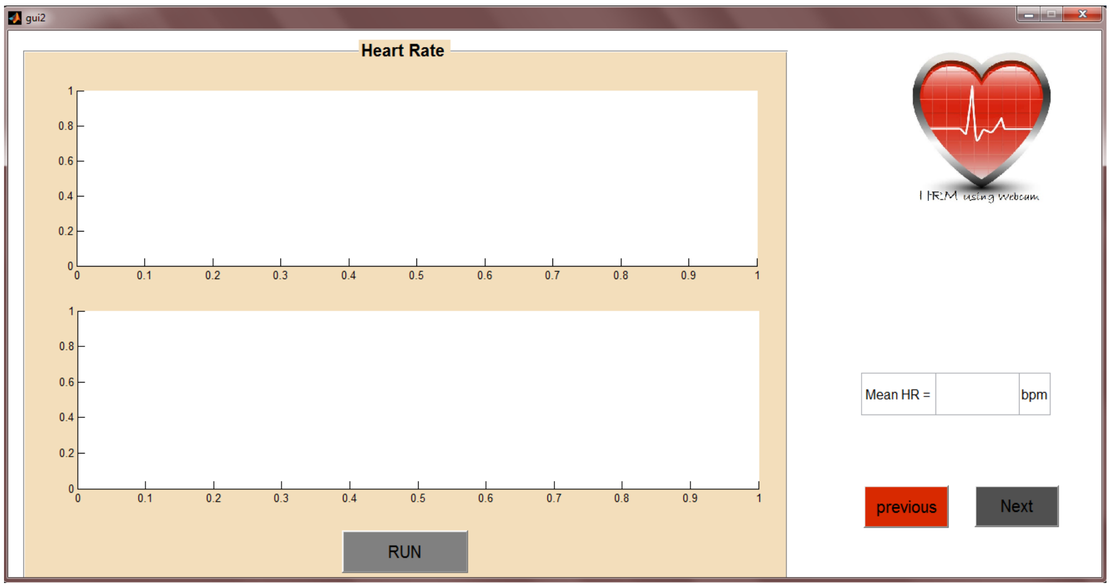


Fig.(3.10): second page handle using GUI.

Chapter (4)

Results and Discussions

4.1Results

We have developed a computer program to measure the heart rate. The program can be setup on a personal computer or a laptop. The program is programmed to capture a video of the person face, then it processes it to output the heart rate and indicates whether it is normal or abnormal. The program has been tested on a dataset of fifty (50) persons on a laptop with Intel(R) core™ i5-5200U CPU @ 2.20 GHz processor and 4 GB RAM. The program has achieved a promising results with a percentage error of 6.12 % and speed of one minute. The program has achieved an accuracy of diagnosis (normal or abnormal) of 93.88 %.

4.2 Data base

We validate the program on a larger database to be sure that the program is reliable. This is a sample of data base:

Person	Manual	Measurements Using webcam	Error	Accuracy
a1	80	79.9	0.00125	0.99875
a2	80	72.09	0.0989	0.9011
a3	71	70.25	0.0128	0.9872
a4	80	78.05	0.0244	0.9756
a5	74	73.4	0.0081	0.9919
a6	70	66.6	0.0485	0.9515
a7	65	64.1488	0.01309	0.98691
a8	66	65.3474	0.009	0.991
a9	62	60.314	0.027	0.973
a10	75	67.1926	0.1041	0.8959
a11	61	57.259	0.0613	0.9387
a12	83	80.264	0.033	0.967

a13	76	71.3	0.062	.938
a14	88	84.8	0.036	0.964
a15	74	61.23	0.173	0.827
a16	80	79.49	0.0064	0.9936
a17	80	61.55	0.23	0.77
a18	89	59.81	0.328	0.672
a19	66	62	0.06	0.94
a20	71	72.09	0.15	0.85
a21	85	87.5	0.029	0.971
a22	71	72.9	.01	0.99
a23	71	70.25	0.02	0.98
a24	60	63.56	0.059	0.941
a25	77	92.5663	0.2	0.8
a26	75	82.4	0.09	0.91
a27	66	62	0.06	0.94
a28	80	79.8	0.0025	0.9975
a29	84	90	0.07	0.93
a30	67	70	0.045	0.955
a31	68	66	0.025	0.975
a32	98	97	0.01	0.99
a33	88	88	0	1
a34	76	77	.013	0.987
a35	80	80	0	1

Chapter (5)

Conclusion and Future work

5.1Conclusion

The current measurement of the heart rate requires either an electrocardiogram (about two pairs of electrodes, electrocardiogram machine, and analyzer) or a dedicated device in the hospital to measure the heart rate. Our methodology offers faster and simpler measurement and reduced resources by using only a simple webcam and a data analyzer (a simple smart phone is quite enough. In addition, this will avoid the need to go to the hospital).

In addition, it can serve physicians by making it possible to follow up the patient anywhere, at any time without having to go to the hospital.

5.2 Future work

- We plan to present our program as a commercial product that can be run on a personal computer with a camera or laptop (an executable file that can be run on any workspace or any platform).
- In addition, we plan to extend this program to be run as a mobile application.
- Once we validate the program on a larger database and be sure that the program is reliable and robust on different patients, we will start to commercialize the product to the market.
- Our initial results indicate that our tool will represent an efficient alternative to the current techniques of heart rate measurements, especially in terms or portability and ability to be used at any place and at any time.

References and links

1- Heart disease and stroke statistics—2016 update: a report from the American Heart Association [published online ahead of print December 16, [2015] Circulation. doi: 10.1161/CIR.0000000000000350.

2- Hao-Yu Wu et al "Eulerian Video Magnification for Revealing Subtle changes in the world" (ACM Transactions on Graphics (TOG) SIGGRAPH 2012 Conference Proceedings,Vol 31 Issue 4,Article No 65,July 2012)

3- Measuring heart rate with a smartphone camera http://www.ignaciomellado.es/blog/Measuring-heart-rate-with-a-smartphone-camera.

4- Heart Rate Measuring Techniques—Taking Your Pulse Manually *www.glencoe.com/sites/common.../heart_rate...heart/the_heart_activity_4.pdf*

5- The Best Heart Rate Monitor Apps http://www.livescience.com/49653-best-heart-rate-monitor-apps.html