# Real-time infant physiological measure monitoring system using video footage

***SCE – College of engineering.***

***Computer software engineering department.***

*Dina Aideliand 313024325*

*Yonatan Meirov 203969357*

*Dima Gurevich 312983315*

*Boris Dahtiar 307784900*

***Supervisor: Daniel Khankin***

*2015-2016*

# Abstract

Eulerian Video Magnification is a recently presented method capable of revealing temporal variations in videos that are impossible to see with the naked eye. Using this method, it is possible to visualize the flow of blood as it fills the face. From its result, a person’s heart rate is may be extracted. This work has been developed to test the feasibility of such an application.

There has been some successful effort on the assessment of vital signs, such as, heart rate, and breathing rate, in a contact-free way using a web camera and even a smartphone. However, since the Eulerian Video Magnification method was recently proposed, its implementation has not been tested on a Raspberry Pi yet. Thus, the Eulerian Video Magnification method performance for color amplification was optimized in order to execute on a Raspberry Pi device at a reasonable speed. The application implemented includes features, such as, detection of a person’s cardiac pulse, dealing with artifacts’ motion, and real-time display of the magnified movement.

Since it is a cheap method of assessing vital signs in a contact-free way, this project has potential for advancing fields, such as, telemedicine, personal health-care, and ambient assisting living.

# Table of Contents

[Real-time infant physiological measure monitoring system using video footage 1](#_Toc450064187)

[Abstract 2](#_Toc450064188)

[Table of Contents 3](#_Toc450064189)

[Motivation 5](#_Toc450064190)

[Introduction 5](#_Toc450064191)

[Chapter 1 - The Algorithm overview 6](#_Toc450064192)

[1.1 Photo-plethysmography 6](#_Toc450064193)

[1.2 Signal post-processing 6](#_Toc450064194)

[1.2.1 Independent Component Analysis 6](#_Toc450064195)

[1.2.2 Eulerian Video Magnification 7](#_Toc450064196)

[1.2.3 Detrending 9](#_Toc450064197)

[1.3 Motion Detection 10](#_Toc450064198)

[1.4 Heart Rate Estimation 10](#_Toc450064199)

[1.4.1 Power spectrum 10](#_Toc450064200)

[1.4.2 Pulse wave detection 10](#_Toc450064201)

[Chapter 2 - Implementation 11](#_Toc450064202)

[2.1 Programming Language and Platform 11](#_Toc450064203)

[2.2 Equipment 12](#_Toc450064204)

[2.2.1 RaspberryPI B 12](#_Toc450064205)

[2.2.2 Camera 13](#_Toc450064206)

[2.2.3 Smartphone 13](#_Toc450064207)

[2.3 Implementation details 14](#_Toc450064208)

[2.3.1 Overview of the Eulerian video magnification framework 14](#_Toc450064209)

[2.3.2 Face detection 15](#_Toc450064210)

[2.3.3 Signal validation 15](#_Toc450064211)

[2.3.4 Heart rate estimation 16](#_Toc450064212)

[2.3.5 Motion detection 17](#_Toc450064213)

[Algorithm Pseudo code: 18](#_Toc450064214)

[Explanation of our implementation in C++: 19](#_Toc450064215)

[References 21](#_Toc450064216)

# List of Figures

1. [Figure 1.1 - Overview of the Eulerian Video Magnification method](#_1.2.2_Eulerian_Video)
2. [Figure 1.2 - 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](#_1.2.2_Eulerian_Video)
3. [Figure 1.3 - Examples of temporal filters](#_1.2.2.2_Temporal_filtering)
4. [Figure: 1.4 - Original RR series and fitted trends (above) and detrended series without magnitude distortion (below).](#_1.2.3_Detrending)
5. [Figure 2.1 System Infrastructure Flow](#_2.2_System_Infrastructure)
6. [Figure 2.2 - RspberryPi device illustration](#_2.2.1_RaspberryPI_B)
7. [Figure 2.3 - Web camera illustration](#_2.2.2_Camera)
8. [Figure 2.4 - Smartphone illustration](#_2.2.3_Smartphone)

[TO ADD – motion detection after the fix]

# Motivation

One of the main worries of a mother after giving birth to her child is the health of the infant while asleep.

The biggest fear is of sleep apnea and/or suffocation. The cause for these is usually unknown. This situation is called “death in the cradle”. If the seize of breathing is caught in time, the infant can be saved. Currently, there are systems that implement chest movement monitoring in order to identify seize of breathing. In conducted researches [[9](#_]_Hao-Yu_Wu,)], it was found that using these systems does not improve the chance to immediately identify the issue [[13](#_RJ_Spurrier_–)] and does not provide a substantial benefit. Moreover, many of these systems’ alerts are false alarms, which lowers the reliability of the system.

Existing systems need to be in constant contact with the infant. [[14](#_Tahnk,_Jeana_Lee.)] They work using electric sensors that may endanger the infant. In addition, the systems transmit WiFi signal of their input, which may be alarming to some parents, due to the proximity of the devices to the infant.

The main goal is to develop a system that monitors physiological changes of a sleeping infant, using a web camera positioned outside the crib without physical contact with the infant. Given a life-threatening situation, the system will identify the issue and alert the parent. The system will be relatively cheap and available to parents from all socio-economic classes. Our system will not require unique adjustments or the purchasing of compatible equipment.

# Introduction

The method is based on the Eulerian Magnification Algorithm in real time via the web camera and a Raspberry PI device. Due to the high activity of blood on the infant forehead and the featured PPG signal in green channel, the remote camera captures a portion of the forehead and generates a sampled signal from the green channel for further processing and feature extracting.

Due to external sources at different frequencies and the noise that comes from the image sensor of digital camera, the heart beat signal has to be processed so that we could extract HRV (Heart rate variability) data.

# Chapter 1 - The Algorithm overview

This chapter focuses on the heart rate estimation on an infant’s face captured through a simple webcam.

This chapter describes, the concept that explains how the cardiac pulse is detected from an infant’s face in a remote, contact-free way.

## 1.1 Photo-plethysmography

Photo-plethysmography (PPG) is the concept of measuring volumetric changes of an organ optically. Its most established use is in pulse oximeters. PPG is based on the principle that blood absorbs more light than surrounding tissue thus variations on blood volume affect light reflectance. [[1](#_Paul_Viola_and)]

The use of dedicated light sources and infra-red wavelengths, and contact probes has been the norm. [[2](#_S.S._Ulyanov_and) [3](#_EF_Greneker._Radar) [4](#_M._Garbey,_N.)] The method used captures the pixel values (red, green, and blue channels) of the facial area of a previously recorded video. The pixel values within a region of interest (ROI) were then averaged for each frame. This spatial averaging was found to significantly increase signal-to-noise ratio. The heart rate estimation was then calculated by applying the Fast Fourier transform and the power spectrum. The green channel features a stronger cardiac signal as compared to the red and blue channels. This is a strong evidence that the signal contains variations in the blood volume, because hemoglobin absorbs green light better than red and blue light [[5](#_Wim_Verkruysse,_Lars)].

## 1.2 Signal post-processing

After obtaining the raw pixel values (red, green, and blue channels), a combination of the following methods may be used to extract and improve the reflected plethysmography signal.

### 1.2.1 Independent Component Analysis

Independent Component Analysis is a special case of blind source separation and is a technique for uncovering independent signals from a set of observations that are composed of linear mixtures of the underlying sources. [[6](#_P._Comon._Independent)]

In our project, the source of signal of interest is the cardiac pulse of the face. During the cardiac cycle, there are various changes to a blood vessels which affects the amount of reflected light and allows us to see cardiovascular events.

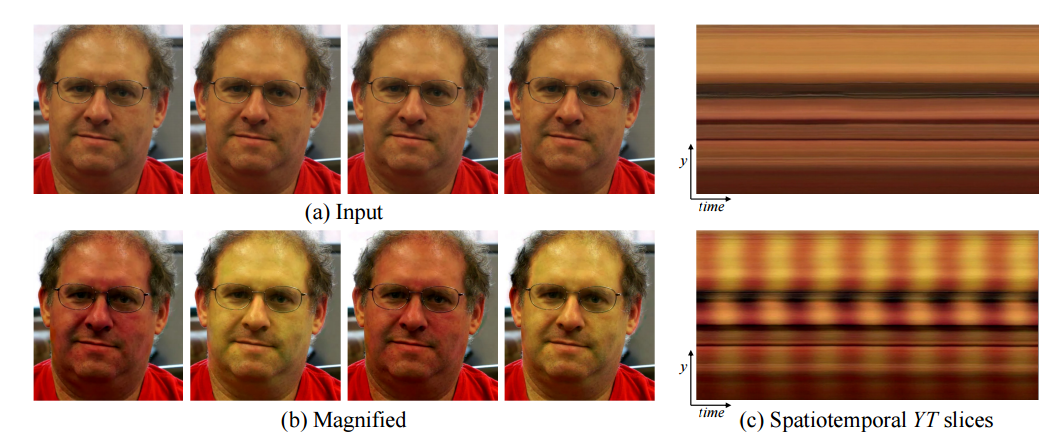
By recording a video of the ROI, the red, green, and blue (RGB) color sensors pick up a mixture of the PPG signal. Each color sensor records a mixture of the original source signals with slightly different weights. These observed signals from the red, green and blue color sensors are denoted by , and respectively, which are amplitudes of the recorded signals at time point . In conventional Independent Component Analysis model the number of recoverable sources cannot exceed the number of observations, thus three underlying source signals were assumed, represented by , and . The Independent Component Analysis model assumes that the observed signals are linear mixtures of the sources, i.e. for each . This can be represented compactly by the mixing equation

where the column vectors and the square matrix contains the mixture coefficients . The aim of Independent Component Analysis model is to find a separating or demixing matrix that is an approximation of the inverse of the original mixing matrix whose output

is an estimate of the vector containing the underlying source signals. To uncover the independent sources, must maximize the non-Gaussianity of each source. In practice, iterative methods are used to maximize or minimize a given cost function that measures nonGaussianity. [[7](#_M.Z._Poh,_D.J.) [8](#_M.Z._Poh,_D.J._1)]

### 1.2.2 Eulerian Video Magnification

The basic approach of the Eulerian Algorithm is to consider the time series of color values at any spatial location (pixel) and amplify variation in a given temporal frequency band of interest. For example, in the figure below we automatically select, and then amplify, a band of temporal frequencies that includes plausible human heart rates. The amplification reveals the variation of redness as blood flows through the face. For this application, temporal filtering (see more extensive explanation below) needs to be applied to lower spatial frequencies (spatial pooling) to allow such a subtle input signal to rise above the camera sensor and quantization noise. [[9](#_]_Hao-Yu_Wu,)]



**Figure 1.2** 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. **Source:** [[9](#_]_Hao-Yu_Wu,)]

The temporal filtering approach not only amplifies color variation, but can also reveal low-amplitude motion. For example, in one of the videos, we show that we can enhance the subtle motions around the chest of a breathing baby.

#### 1.2.2.2 Spatial filtering

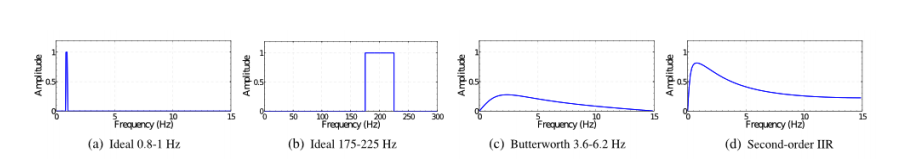
Spatial filtering is used to significantly increase signal-to-noise ratio [[10](#_Wim_Verkruysse,_Lars_1)]. Subtle signals, such as, an infant’s heart rate from a video of its’ face, may be enhanced this way. For this purpose a layer of the Gaussian pyramid is computed, which may be obtained by successively scaling down the image by calculating the Gaussian average for each pixel.

However, for the signal of interest to be revealed, the spatial filter applied must be large enough. The following is an equation to estimate the size for a spatial filter needed to reveal a signal at a certain noise power level:

where represents the signal over spatial frequencies, and since the wavelength, , cutoff of a spatial filter is proportional to its radius, , the signal may be represented as . The noise power, , can be estimated using to the technique mentioned in “Noise estimation from a single image” [[11](#_C._Liu,_W.T.)]. Finally, because the filtered noise power level, , is inversely proportional to it is possible to solve the equation for , where is a constant that depends on the shape of the low pass filter.

#### 1.2.2.2 Temporal filtering

Temporal filtering is used to extract the motions or signals to be amplified. Thus, the filter choice is application dependent. For motion magnification, a broad bandpass filter, such as, the butterworth filter, is preferred. A narrow bandpass filter produces a more noise-free result for color amplification of blood flow. For a real-time implementation low-order IIR filters can be useful for both: color amplification and motion magnification. These filters are illustrated below.



**Figure 1.3** Examples of temporal filters. **Source:** [[9](#_Hao-Yu_Wu,_Michael)]

#### 1.2.2.2 Emphasize color variations for human pulse

Using the right configuration can help extract the desired signal. There are four steps to take when processing a video using the Eulerian Video Magnification method:

1. select a temporal bandpass filter;

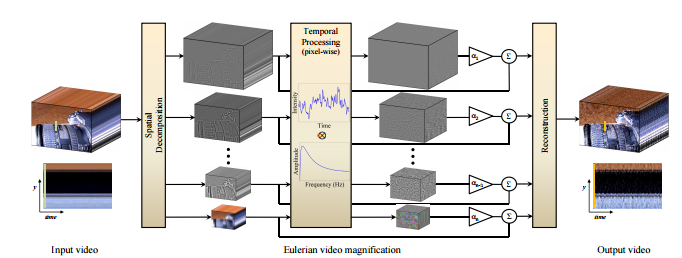
2. select an amplification factor,;

3. select a spatial frequency cutoff (specified by spatial wavelength, ) beyond which an attenuated version of α is used;

4. select the form of the attenuation for —- either force to zero for all , or linearly scale down to zero.

For human pulse color variation, two temporal filters may be used, first selecting frequencies within 0.4-4Hz, corresponding to 24-240 beats per minute (bpm), then a narrow band of 0.83- 1Hz (50-60 bpm) may be used, if the extraction of the pulse rate was successful.

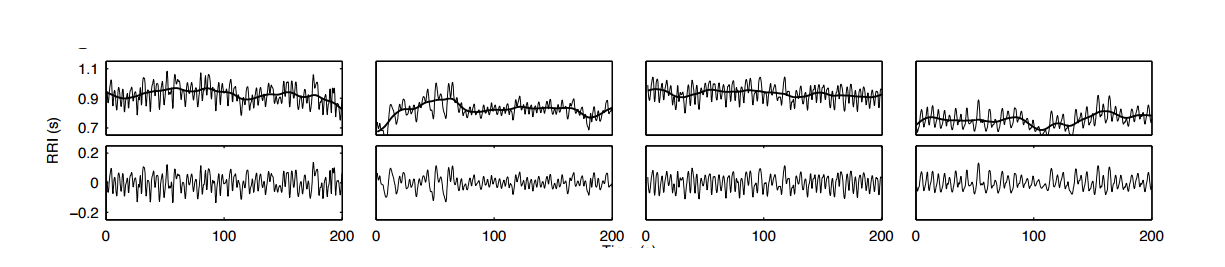
To emphasize the color change as much as possible, a large amplification factor, , and spatial frequency cutoff, , is applied. With an attenuation of α to zero for spatial wavelengths below. The resulting output can be seen in figure 1.3 (to check)



**Figure 1.1** Overview of the Eulerian Video Magnification method **Source:** [[9](#_]_Hao-Yu_Wu,)]

### 1.2.3 Detrending

Detrending is a method of removing very large ultralow-frequency trends an input signal without any magnitude distortion, acting as a high-pass filter.



**Figure: 1.4** Original RR series and fitted trends (above) and detrended series without magnitude distortion (below). **Source:** [[12](#_Mika_P_Tarvainen,)]

The method consists of separating the input signal, , into two components, as , where is the nearly stationary component, and is the low frequency aperiodic trend component.

An estimation of the nearly stationary component, , can be obtained using the equation below [[12](#_Mika_P_Tarvainen,)].

where is the identity matrix, is the discrete approximation of the second order, and is the regularization parameter.

This formula presents an example of what this method is able to achieve. The example uses real RR series and the effect of the method on time and frequency domain analysis of heart rate variability is demonstrated not to lose any useful information. [[9](#_]_Hao-Yu_Wu,)]

## 1.3 Heart Rate Estimation – Power Spectrum

In order to convert the extracted PPG signal into a number of beats per minute (bpm), further processing must be done. The method which achieves this goal is Power Spectrum.

The method uses Fourier transform, which is a mathematical transform capable of converting a function of time, , into a new function representing the frequency domain of the original function. To calculate the power spectrum, the resulting function from the Fourier transform is then multiplied by itself. Since the values are captured from a video, sequence of frames, the function of time is actually discrete, with a frequency rate equal to the video frame rate, . The index, , corresponding to the maximum of the power spectrum can then be converted into a frequency value, , using the equation:

Where is the size of the signal extracted. can then be multiplied by 60 to convert it to beats per minute, and have an estimation of the heart rate from the extracted signal.

## 1.4 Motion Detection

Motion detection method is an algorithm for detecting moving objects from a static background scene based on frame difference.

A sequence of frames is captured through the camera, and for each 2 frames, we calculate the absolute difference between the consecutive frames, and store them in a Difference image in the system.

The Difference image is then converted into gray image and translated into a Binary image (Threshold). A Morphological filter is then applied to the Binary image, in order to remove noise.

There are 3 methods which are commonly used in Motion detection:

* Consecutive frame subtraction
* Background subtraction
* Optical flow

We are using the **Consecutive frame subtraction** method.

Detection of a moving object from a sequence of frames is performed by calculating frame differences. The goal of this approach is to detect the moving object from the differences between the current frame, and the reference frame, by subtracting them from one another. This method is commonly used for Motion detection.

Any motion detection needs to handle a number of critical situations such as:

* Image noise, due to a poor quality image source
* Small movements of non-static objects
* Shadow regions are projected by foreground objects and are detected as moving objects.

To prevent such false positives we use Gaussian Blur, and Morphological filtering to remove Image noise.

### 1.4.1 Difference between Two Consecutive Frames

I (k) is supposed to be the value of the k frame in image sequences. I (k+1) is the value of the (k+1) frame in image sequences. The absolute differential image (D1) is defined as follows:

I (d (k, k+1)) = |I (k+1) – I (k)| = D1

### 1.4.2 Transformation of absolute differential image to Gray Image

There are holes in moving object area, and a contour of a moving object is not closed entirely. The absolute differential image is transformed into a gray image to facilitate further operations.

RGB to Gray: Gray= 0.299\*R + 0.587\*G + 0.114\*B

### 1.4.3 Filtering and Binarizing a Transformed Gray Image

A Gaussian blurring algorithm is applied to smooth the images. Due to tiny variations in the digital camera sensors, no two frames will be 100% the same — some pixels will mostcertainly have different intensity values. Gaussian smoothing algorithm is applied to average pixel intensities across an *11 x 11* region. This helps smooth out high frequency noise that could throw the motion detection algorithm off.

In order to remove the holes, the differential image (D1) is passed through the Gauss low pass filter, and binarized using Binary threshold to receive the binary image (D2).

(D2) (x, y) = >Where (x, y) is a pixel coordinates in image.

### 1.4.4 Morphological filtering

Morphological image processing is a collection of operations related to the shape of features in an image, such as boundaries, skeletons, etc. In any given technique, we probe an image with a small shape or template called a structuring element, which defines the region of interest or neighborhood around a pixel. [[15](#_Nishu_Singla._Motion)]

Morphology functions like: Binary Erosion, Dilation - only works on gray-scale or binary images.

In our motion detection Algorithm we use the Binary Erosion function.

The basic idea in Binary Morphology is to probe an image with a simple, pre-defined shape, drawing conclusions on how this shape fits or misses the shapes in the image. This simple "probe" is called a Structuring Element, and is itself a Binary image.

# Chapter 2 - Implementation

## 2.1 Programming Language and Platform

The programming language that we use is C++. In C++, we can develop a prototype quickly and easily. The main usage of the mathematical libraries in C++ is relatively easy, and can be used to convert the algorithm to code. C++ is known for its graphic rendering prowess, and this seems like the logical choice of language.

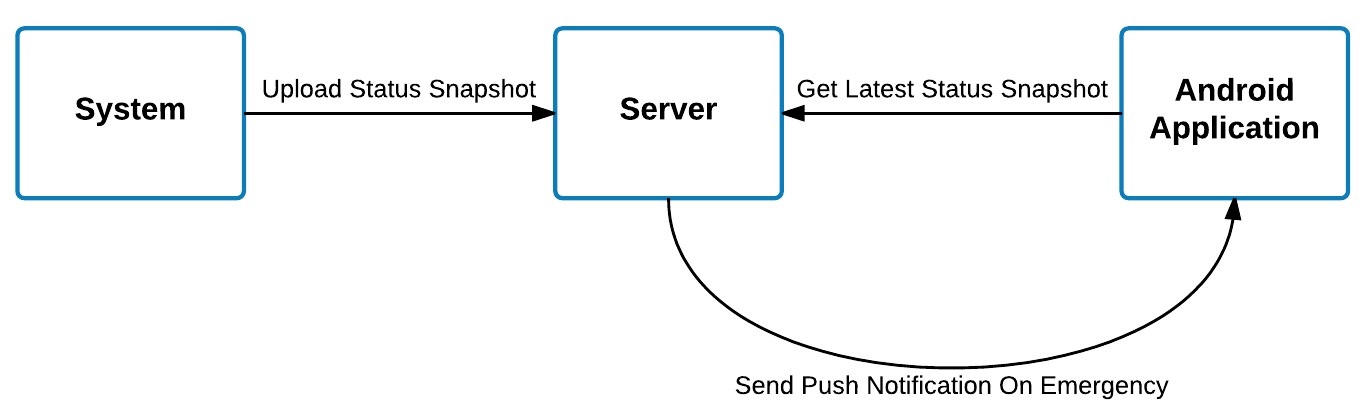
In order to access the camera functionality and process the video we will use OpenCV (Computer Vision Library) library. OpenCV is a library of programming functions mainly aimed at real-time image processing. To support these, it also includes a statistical machine learning library. Moreover, it is a cross-platform and open source library that is free to use and modify under the BSD license. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Our main development environment will be Visual Studio IDE.

## 2.2 System Infrastructure

The system needs to provide reliable access via the mobile application. In addition, a persistent data storage is needed in order to avoid data loss on system failure, which can be caused by outside influences (e.g. power break).

For this purpose, we have built a basic API server that will allow communication with the system, via the android application, and store the latest status snapshot of the infant.



**Figure 2.1** System Infrastructure Flow

To provide easier communication between the system and the android application, the server uses a RESTful API.

A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data. This provide us remote control on the data storage of the server, and a reliable storage space, that is accessible from anywhere. It also provide us a needed modularity, and flexibility, which is needed in larger scale project that includes multiple components and systems.

The server itself is built using Node.js + Express, and hosted on Heroku. Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine, which is used for developing server-side Web applications. Express is a minimal and flexible Node.js web application framework that provides a set of features for web and mobile applications. Heroku is a cloud Platform-as-a-Service (PaaS) supporting several programming languages, which provides cloud hosting for web apps.

The android application that communicates with the server, as seen in figure 2.1, provides a secure remote login into the system, and the ability to see live snapshot updates every 5 secs. The connection from the application to the server, is done over HTTPS, and provides a secure channel to the system.

The Android application itself, is built using Ionic Framework. Ionic is an open-source SDK for hybrid mobile app development, built on top of AngularJS and Apache Cordova. It provides tools for developing hybrid mobile apps using Web technologies like HTML, CSS, and JavaScript.

## 2.3 Hardware

### 2.3.1 RaspberryPI B



**Figure 2.2** RspberryPi device illustration

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device, which enables people of all ages to explore Computing, and learn how to program in languages like Python and C++.

It is capable of doing everything you would expect a desktop computer to do. The Raspberry Pi has the ability to interact with the outside world, and has been used in a wide array of digital maker projects, from music machines and parent detectors.

In addition, the Raspberry Pi has an USB port, Ethernet, and we have the ability to connect a web camera.

### 2.3.2 Camera

**Figure 2.3** web camera illustration

The number of frames per second that the camera is capable of delivering is very important, because each frame will be converted to a signal sample. The number of signal samples per second is the number of frames per second. The more FPS the camera provides, the more detail you can build in the PPG signal. Heart rate measurements require a minimum sampling frequencies of 30 Hz. So we need camera with ability to provide at least 30 FPS.

At capture time, it is recommended to:

* Minimize extraneous motion. Put the camera on a tripod. If appropriate, provide support for your subject (e.g. hand on a table, stable chair).
* Minimize image noise. Use a camera with a good sensor. Make sure there is enough light.
* Record in the highest spatial resolution possible and have the subject occupy most of the frame. The more pixels covering the object of interest - the better the signal you would be able to extract.

### 2.3.3 Smartphone



**Figure 2.4** Smartphone illustration

The system can be used with any standard Android Smartphone, and does not require any special configurations, other than installing the app.

The system will communicate with the app on the smartphone, and alert the parents when there is a life-threatening situation. You will be able to view snapshots of the infant situation (e.g. BPM, image of the infant) via the app, in a 5 sec interval.

## 2.4 Implementation details

The overall algorithm was divided into several steps, illustrated in figure 2.4. The language used to implement the desktop application and library was C++. In addition, for the image processing operations, the computer vision library, OpenCV, was used.

### 2.4.1 Algorithm Pseudo code

Main Loop:

define VideoCapture and extract width, height and fps

define minFaceSize as 40% of the frame

create FrameHandler instance

load LBP Cascade XML for face detection

loop while ESC key has not been pressed and frame is not empty

read frame from camera

send snapshot to server every 60 seconds

detectMultiScale - locate all faces and insert to discoveredFaces list

if discoveredFaces.size >= knownFaces.size //same amount of faces or new faces

for each knownFace

locate the closest face in discoveredFaces

process (see Face Processing below)

for each discoveredFace in discoveredFaces that is not in knownFaces

create new face instance and add to knownFaces

process (see Face Processing below)

if discoveredFaces.size < knownFaces.size //lesser amount of faces

set all knownFaces as unmatched

for each discoveredFace in discoveredFaces

match to closest unmatched knownFace

set knownFace as matched

process (see Face Processing below)

remove all unmatched faces from knownFaces

if there is no face with a valid heart rate

send alert to server

run motion detection (see Motion Detection below)

Face Processing:

update face size and position and re-adjust ROI position

magnify ROI using Eulerian Video Magnification

if signal is stable and not too noisy

detrend and normalize the pulse signal

apply mean filter to the pulse signal

if pulse signal is valid

estimate heart rate

else reset all components and estimate Motion Detection

Motion detection:

capture sequence of frames at regular itrervals

for each 2 frame:

calculate difference between the frames and output them to a difference image

convert the difference image into gray scale and translated into binary image (threshold)

apply morphological filter to the binary image, to remove noise

### 2.4.2 Algorithm Explanation

#### 2.4.2.1 Face detection

The faces are detected using detectMultiScale given a minimum face size of 40% of the frame (in order to improve performance). The detected faces are added to discoveredFaces list.

The FrameHandler class holds a list of knownFaces that had been previously detected.

The function detectFacesAndCalcHeartRate handles 2 cases:

1. The number of knownFaces <= detectedFaces (a new face was detected or the faces remain as previously):

Should there be any knownFaces, the algorithm first matches the discoveredFaces to the knownFaces. It does so by iterating over the knownFaces and locating which face from the disocveredFaces is closest to the current knownFace.

Once the closest face is discovered, the face size and position are updated accordingly.

If there are no new faces, or there are discovered faces that hadn’t been matched yet, loop over remaining discoveredFaces and create a new Face instance for them. Store the newly created instance in the knownFaces list.

1. The number of the knownFaces > discoveredFaces (One or more faces are missing from the current frame):

Should a face or faces disappear from the frame, the algorithm has to match the currently known faces to the discovered faces and determine which are no longer relevant.

It does so by iterating over all knownFaces and marking them as unmatched (boolean value). Afterwards, it will iterate over the discoveredFace and locate the closest knownFace (as opposed to case 1).

Once a closest face had been located, the knownFace is marked as matched and the face size and position are updated accordingly. At this point, every unmatched face is removed from the knownFaces list.

#### 2.4.2.2 Face processing

After the faces were recognized and matched, they are sent to a function named updateSizeAndPosition in the Face class. The function calculates the face rectangle ratio and executes openCV’s interpolate function in order to update new face size and position. Afterwards, the forehead (ROI) rectangle size and position are calculated relatively to the new face rectangle.

The Eulerian Video Magnification algorithm is applied to the ROI which produces an amplified frame signal. From this amplified signal, the green channel is extracted and an average pixel value is calculated and inserted to an array of raw signal. The signal is checked for stability (if it’s not too noisy), if the raw signal’s standard deviation is smaller or equal to half of amplification factor. Should the signal be stable the following operations are applied:

1. Detrending, as explain in section […….]
2. Normalization, the normalized signal is obtained from the detrended signal by subtracting the mean of the signal and dividing it by the detrended signal standard deviation.
3. Mean filter, the normalized signal is blurred 3 times using the default kernel value:

Which smoothing the normalized signal and reducing noise. Smoothing is done by sliding a window (kernel or filter) across the whole image and calculating each pixel a value based on the value of the kernel and the value of overlapping pixels of original image. This process is mathematically called as convolving an image with some kernel.

#### 2.4.2.3 Heart Rate Estimation

After the Face Processing stage, a band-limit filter removes frequencies outside the interest band. This reduces the noise in later processing steps and make the resulting heart rate signal smoother. Our band of interest is: 40-240 bpm. The following calculation applies:

Total Rows \* (Min Frequency OR Max Frequency) / 60 / (FPS + 1)

Then, an estimation of the BPM is made using the Power Spectrum method. The specifics of the method is mentioned in [section 1.3](#_1.3_Heart_Rate).

#### 2.4.2.4 Overview of the Eulerian video magnification framework

As mentioned in the previous chapters, the Eulerian Video Magnification method is capable of magnifying small motions and amplifying color variations which may be invisible to the naked eye.

First, we identify the faces in the camera frame. Once a face is detected and locked on, the magnification process will begin on the forehead.

The system first decomposes the input video sequence into different spatial frequency bands, and applies the same temporal filter to all bands. The filtered spatial bands are then amplified by a given factor α. Then, they are added back to the original signal, and collapsed to generate the output video. This process can be seen in figure 1.1.

In our case, we use the system to reveal unseen motions captured by a Digital web camera, and amplify the green channel, in order to extract the BPM (heartbeats per minutes).

algorithm-flow.pdf

**Figure 2.4**: Overview of the implemented algorithm to obtain the heart rate of an infant from a webcam or video using the Eulerian Video Magnification method.

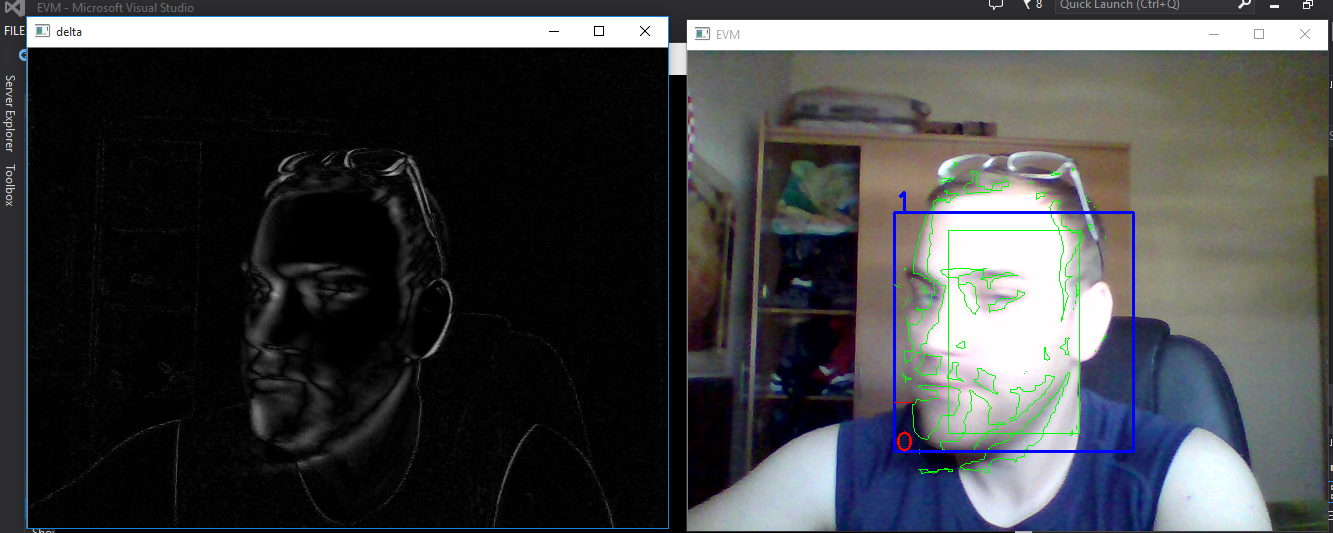
### 

### Motion detection Implementation

The motion detection algorithm is executed in case a valid heart rate can’t be estimated from the captured frames. The algorithm detects motion in order to identify infant’s movement and breathing. The moving infant is detected from the differences between 2 images (frames). The difference image shown as a white contour on a black background.

Computing the difference between two frames is a simple subtraction, it takes the absolute value of their corresponding pixel intensity differences of 2 consecutive frames:

An example of a difference image can be seen below:

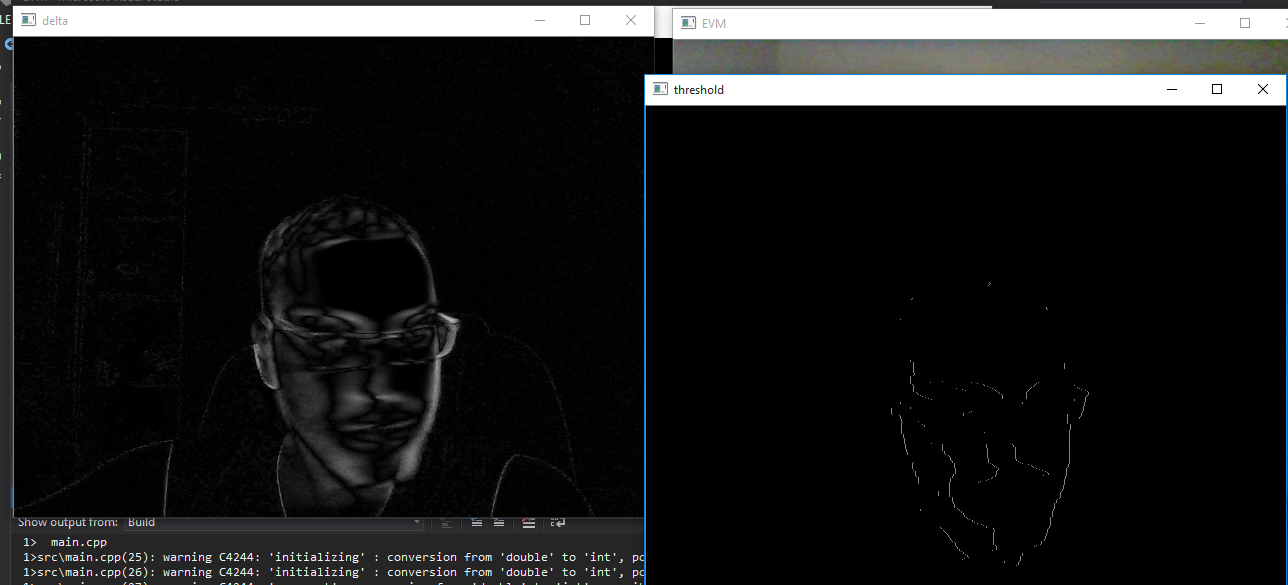


**Figure 3.2:** An example of the frame delta

The background of the image is clearly black. However, regions that contain motion are much lighter. This implies that larger frame differences indicate that motion is taking place in the image.

Then, a threshold on the difference image is applied to reveal regions of the image that have significant changes in pixel intensity values. If the delta is less than 2, we discard the pixel and set it to black. If the delta is greater than 2, we will set it to white.

An example of our threshold binary image can be seen below:



**Figure 3.3:** Thresholding the difference image to segment to the right.

Given this threshold image, it’s simple to apply contour detection to find the outlines of these white regions. It draws contours surrounding the motion region and updates the status string to indicate that the infant is moving.

# 

# References

1. Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In computer Vision and Pattern Recognition, 2001.
2. S.S. Ulyanov and V.V. Tuchin. Pulse-wave monitoring by means of focused laser beams scattered by skin surface and membranes.
3. EF Greneker. Radar sensing of heartbeat and respiration at a distance with applications of the technology, 1997.
4. M. Garbey, N. Sun, A. Merla, and I. Pavlidis. Contact-free measurement of cardiac pulse based on the analysis of thermal imagery, 2007.
5. Wim Verkruysse, Lars O Svaasand, and J Stuart Nelson. Remote plethysmographic imaging using ambient light, 2008.
6. P. Comon. Independent component analysis, a new concept?, 1994.
7. M.Z. Poh, D.J. McDuff, and R.W. Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation, 2010.
8. M.Z. Poh, D.J. McDuff, and R.W. Picard. Advancements in noncontact, multiparameter physiological measurements using a webcam, 2011.
9. Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William T. Freeman. Eulerian video magnification for revealing subtle changes in the world, 2012.
10. Wim Verkruysse, Lars O Svaasand, and J Stuart Nelson. Remote plethysmographic imaging using ambient light, 2008.
11. C. Liu, W.T. Freeman, R. Szeliski, and S.B. Kang. Noise estimation from a single image, 2006.
12. Mika P Tarvainen, Perttu O Ranta-aho, and Pasi A Karjalainen. An advanced detrending method with application to hrv analysis, 2002.
13. RJ Spurrier – BabyGearLab. How to Choose the Best Video Monitor for You and Your Baby, 2015.
14. Tahnk, Jeana Lee. 6 Gadgets to Keep Your Newborn Safe, 2013.
15. Nishu Singla. Motion Detection Based on Frame Difference Method, 2014.