Universida de Vigo

Escola Técnica Superior
De Enxeñeiros De Telecomunicación

PROYECTO FIN DE CARRERA

Heart rate estimation using facial video information

AUTORA: Noa García Docampo
TUTOR: Pablo Dago Casas

Curso: 2011/12

Heart rate estimation using facial video information

Proyecto Fin de Carrera

escrito por
Noa García Docampo

bajo la tutela de
Pablo Dago Casas

y realizado durante un período de prácticas preprofesionales, reguladas por el Convenio de Cooperación Educativa, en

GRADIANT
Fundación Centro Tecnolóxico de Telecomunicacións de Galicia

Proyecto Fin de Carrera

HEART RATE ESTIMATION USING FACIAL VIDEO INFORMATION

Autora: Noa García Docampo
Tutor: Pablo Dago Casas

Composición del tribunal	
Presidente/a:	D./Dña.:
Secretario/a:	D./Dña.:
Vocal:	D./Dña.:

Nota obtenida:	

El Vocal El presidente El Secretario

En Vigo a ...... de .... de 20...

Agradecimientos

El que yo realizara este proyecto ha sido fruto de un cúmulo de pequeñas casualidades. Es por ello que quisiera dedicarle unas lineas a cada una de las personas que me ha ayudado durante este (largo) proceso.

En primer lugar gracias a papá y a mamá, por confiar en mi y por todo el apoyo que me han dado a lo largo de mi vida, en especial durante los años de carrera.

Gracias también a Berto, por aguantarme tantas horas bajo los efectos del estrés y los "nopuedo". Gracias por estar siempre ahí y ser, en cierta medida, el impulsor de que haya acabado en Vigo.

También me gustaría agradecer a todas las personas que han hecho fácil y posible mi estancia en Vigo, desde los coordinadores de los dos centros, hasta las secretarias y el personal de relaciones internacionales.

Gràcies als Batuts, perquè sense vosaltres la ETSETB, la biblioteca i els dinars no haguessin estat el mateix. Com sinó hagués aprovat Antenes i Radiocom? Gràcies a tots, i especialment a la Laia i al Alex per estar sempre al meu costat. Gràcies a Taller de So i a Telecogresca, per fer de la vida universitària quelcom més que hores d'estudi i per ensenyar-me que a la universitat no només s'hi va per a fer exàmens.

Grazas a miña compañeira de piso Lúa, que aínda que non sepa moito de telecomunicacións, ensinoume sobre moitos outros aspectos da vida e soubo aconsellarme cando máis o necesitaba. Este proxecto tamén ten un pouquiño de ela.

Gracias a Gradiant, por darme la oportunidad de hacer el proyecto con ellos, y en especial, gracias a todos los que se ofrecieron voluntarios para grabar los vídeos que usamos durante las pruebas. Gracias a Long, por ser el "probador oficial de la aplicación", por sus consejos y por su ayuda.

Por último y dejando lo mejor para el final, gracias a Pablo por todas las (muchas) horas que le ha dedicado a esto, por estar siempre dispuesto a resolver mis dudas y por su paciencia. Gracias por todo lo que he podido aprender contigo, que no ha sido poco.

HEART RATE ESTIMATION USING FACIAL VIDEO INFORMATION

Authoress: Noa García Docampo
Tutor: Pablo Dago Casas

Keywords: heart rate, estimation, independent component analysis, photoplethysmography

In recent years, new technologies used for providing clinical health care remotely have appeared and new fields like telemedicine have experienced huge advancements. New ways for monitoring patients automatically have been developed, as well as techniques for measuring physiological parameters out of the hospital.

One of these parameters is the heart rate, and it is usually used by medical professionals to assist in diagnosis. However, the information provided by the pulse is not only useful in telemedicine, but also in other fields like automatic emotion recognition, interactive videogames or sport-people monitoring.

For this reason, this project addresses the design, evaluation and implementation of a system able to estimate the heart rate of a person using only facial video information coming from a standard webcam. By the use of photoplethysmography techniques and data processing tools, the proposed methodology captures the small illumination changes produced in the user's face because of the variation in the amount of blood present in the surface of the skin. This technique allows an unobtrusive way to measure people's heart rate at any place, without any effort more than being in front of a video camera.

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

AR	Autoregressive
BSS	Blind Source Separation
CLT	Central Limit Theorem
CI	Confidence Interval
ECG	Electrocardiogram
FT	Fourier transform
HR	Heart Rate
HSV	Hue, Saturarion and Value
ICA	Independent Component Analysis
Kf	Kalman filter
MAE	Mean Absolute Error
PDF	Probability Density Function
PPG	Photoplethysmography
PSD	Power Spectral Density
PCA	Principal Component Analysis
RGB	Red, Green and Blue
ROI	Region of Interest
SE	Standrard Error

CHAPTER

1

INTRODUCTION

In animals, blood is the fluid that delivers oxygen and nutrients to the body cells in order to keep them alive. It circulates around the body through the blood vessels, because of the rhythmic contractions of the heart. Each of these contractions, also called heartbeats, produces a pulse wave in body arteries, which is the blood volume pulse or pulse. Actually, the heart rate (HR) is the number of heartbeats per unit of time and it is usually expressed in beats per minute (bpm).

The HR of a person indicates the rhythm at which the heart is working and it depends on several factors, such as age, gender, illnesses or physical activity. It can be measured at any place of the body at which an artery is close to the surface, which is where the pulse is detected.

In the next sections the motivation of this project (Section 1.1) as well as the objectives we want to achieve (Section 1.2), the work schedule (Section 1.3) and the structure of this document (Section 1.4) are detailed.

1.1 Motivation

There are several methods for measuring the HR, being the most common one the palpation method, which consists in pressuring with the index and the middle finger at a place where an artery can be compressed against a bone (such as at the wrist or the neck). Other standard techniques of measuring the HR are:

- Electrocardiogram devices (ECG), which can be seen in Figure 1.1, attach electrodes or a chest strap to the outer surface of it in order to detect the electrical changes produced on the skin because of each heartbeat.
- Pulse oximetry sensors, appearing in Figure 1.2, measure the changing light absorbency produced on the blood because of heart contractions. Theses devices consist on a pair of small diode emitting red and infrared light. The ratio of absorption of these lights is used to compute the HR through the user finger or earlobe.

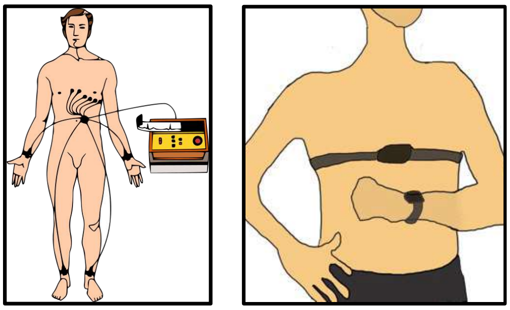

Figure 1.1: Examples of ECG devices.

In spite of being effective, standard devices can cause discomfort, irritation or pain if patients wear it for a long period of time, besides they might be complicated to be used at home without any supervision from a specialist. That is why in the last years, new solutions for non-contact measurements using photoplethysmography (PPG) have been proposed [1, 2].

PPG [3] is an optical technique of sensing blood volume changes of an organ (in this case the skin) through variations in the transmitted or reflected light, which is the same technique used in pulse oximeters. Typically, PPG worked with dedicated light sources with red or infra-red wavelengths. However, new research indicates that HR can be obtained using normal ambient light as the

Figure 1.2: Examples of pulse oximetry devices.

illumination source [2].

Recently, there have appeared some applications that can measure an user's HR in his computer [4, 5]. Although this kind of methods may not be as accurate as a ECG device, they can provide a long-term HR monitoring without being uncomfortable for patients, without requiring them to do much effort and minimizing the amount of cabling in cases where a periodical measure of HR is important. These technologies can be very helpful in increasing fields like telemedicine [6, 7], where usability is a key factor.

In particular, just in the beginning of this project, Philips released an iPad and iPhone application, which measures an user's heart rate using their cameras [8]. This application seems to use PPG techniques, but it does not allow any motion in users' face, being this one of the problems we aim to solve in this project.

1.2 Objectives

This project consists in designing, evaluating and implementing a non-contact heart rate estimation system using PPG techniques. Following the scheme proposed by [4], this work is focused in the development of a real-time application capable of detecting the HR of a person using a standard webcam. This involves capturing the small light fluctuations that are produced in the user face because of the pulse. These small light fluctuations (imperceptible to the human eye) can be sensed by a standard digital camera.

In the existent algorithms of non-contact HR measurements [8], it is essential that users remain still, breathe spontaneously and face the camera along a minute. In this context, this project presents some improvements, which include robustness to user's motion, reduction of the time exposure in front of the camera and real-time estimation process.

Specifically, this work is focused on achieving the following objectives:

- Designing a system that estimates the heart rate of a user using a standard webcam in normal ambient light conditions. The system must work in people with different kind of skins, as well as different ages and genders. Moreover, the system has to be robust to small variations in the position of the user's face.

- Analyzing the accuracy of the system, which must present similar results to the ones obtained with other conventional HR monitor. Besides, the heart rate estimator system must achieve practical results, with special interest in fields like telemedicine or automatically emotion recognition.

- Implementing a final application capable to work in real time, with a reduced exposure period in front of the camera to obtain the first HR estimation.

1.3 Work schedule

This work has been organized in several tasks in order to reach its proposals. The first task was to do a complete study of the state of the art, which includes the existing HR estimation techniques as well as the different algorithm implementations available, and collect all the necessary data for the experiments. After that, a set of essential experiments were carried out in a second task, and a collection of data results was obtained. The third task consists on performing an exhaustive study of the data collect in order to measure the accuracy of the system. At last, the final heart rate estimator was implemented using the results obtained in the previous tasks.

1.4 Document structure

This document is divided in seven chapters. The first of them is an introductory chapter where the motivation and objectives of this work are explained. In the middle chapters a description of the development of this project can be found, as well as the techniques that were used in the process. In the final chapter, conclusions and achievements, as well as the possible future lines of development, are discussed.

The content of each chapter can be summarized as:

- Chapter 1. Introduction: describes the motivation and the objectives of this project, as well as the work schedule and the structure of this document.
- Chapter 2. Face detection, tracking and skin segmentation: explains the different computer vision techniques used in this project.
- Chapter 3. Recovery of Heart Rate signal: presents the used signal processing scheme.
- Chapter 4. Bootstrap: details the statistical tool called bootstrap that is used to analyzed the experiments.
- Chapter 5. Design of the heart rate estimation system: explains the details of the experiments performed in this work.
- Chapter 6. Implementation of the heart rate estimation system: describes the final application and how it was developed.
- Chapter 7. Conclusions and Future Work: presents conclusions, achievements and possible future lines of development.

CHAPTER

2

FACE DETECTION, TRACKING AND SKIN SEGMENTATION

In this chapter, the process used to select the region of interest (ROI) from input video images is described. Section 2.1 gives a brief description of the whole process whereas technical details are explained in Sections 2.2, 2.3 and 2.4.

2.1 Introduction

As mentioned before, each heart beat produces a variation in the amount of blood present in the face skin, which causes small illumination changes in it. To detect and process these light variations, the first thing we have to do is to locate those regions in the input video image containing facial skin.

As a first approach, the region of the image where the face is located is found using a face detection algorithm. Although we use a fast and robust approach, it has two main drawbacks: it limits the user motion to avoid false negatives and includes non-skin pixels in the selected region.

The first of the drawbacks, the user motion problem, can be solved by adding a tracking system which follows the position of the user face along consecutive frames, even if it is partially occluded and cannot be detected. This allows more position and pose variations to the user, increasing robustness and usability.

The inclusion of non-skin pixels in the selected region, such as hair or background, is a strong drawback as well. By adding a skin pixel detector the system can distinguish skin from non-skin pixels. This reduces the amount of noise coming from non facial skin regions in the image and improves the precision of the system.

The final ROI selection system is composed by a face detector (Section 2.2), a tracking system (Section 2.3) and a skin segmentation algorithm (Section 2.4). A scheme of this process can be seen in Figure 2.1.

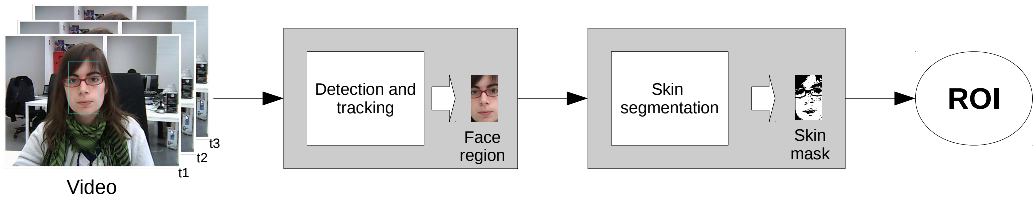

Figure 2.1: Skin pixel detection methodology.

2.2 Face detection

Face detection is the technique that determines locations and sizes of faces in input images. In literature, several algorithms have been developed to solve this problem, however, the method proposed by P. Viola and M. Jones [10] is the first algorithm to provide competitive face detection rates with real-time performance.

Viola-Jones algorithm involves obtaining a series of simple features from an input image and applying a modified version of the AdaBoost classification scheme [9] in order to detect human faces. In fact, Viola-Jones is an iterative algorithm that searches faces along the complete image using different scales. To be able to perform real-time detection, a group of weak classifiers are combined in a cascade to form a final complex classifier. Both the image descriptors used for detection and the Ada-Boost-based scheme for selecting weak classifiers are explained below, as well as the cascade construction of the final face detector.

2.2.1 Rectangular features

The Viola-Jones procedure for object detection classifies input images based on the values taken by specific simple features. These features are similar to the Haar-basis functions [11]. More specifically, the value of each feature is equal to the difference between the sum of the pixels within different rectangular regions. An example of these features is shown in Figure 2.2, where the values of the pixels in the black area are subtracted to those in the white area.

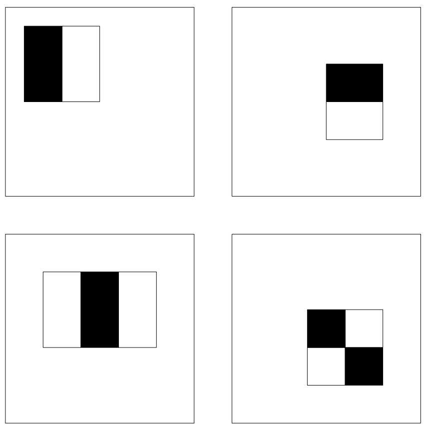

Figure 2.2: Some examples of Haar-like features used in Viola-Jones algorithm.

These features can be rapidly computed by using an intermediate representation of the image called integral image (which is very similar to the summed area table used in computer graphics for texture mapping [12]). The value of the integral image at point (x, y) is the sum of all the pixels above and to the left as it is shown in Figure 2.3. Rectangular features calculation is very efficient if integral image is done, and it becomes essential for a real time face detection.

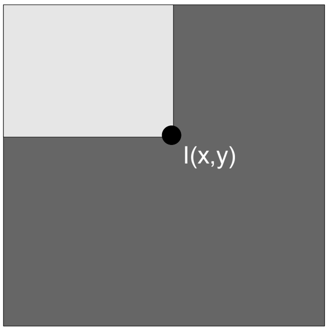

Figure 2.3: Integral image pixel.

2.2.2 Detector training

Even though each of the simple features could be computed very efficiently, computing the complete set of features is very expensive. A modified version of AdaBoost [9] is used to select these features that are more suitable for the face detection problem. In each stage, a weak classifier is designed to select the rectangular feature or features which best separate the positive and negative examples, and determines the optimal threshold to achieve low false positive rates.

In Figure 2.4 the first and the second features chosen by AdaBoost, and therefore the best features for deciding whether it is a face in the input image or not, are shown. It can be seen how these features look for detecting the region where the eyes and the nose are found. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

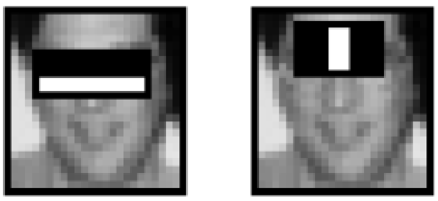

Figure 2.4: First and second features selected by AdaBoost.

2.2.3 Final detector

The cascade of classifiers used in Viola-Jones algorithm increases detection performance and reduces computation time. The main idea is that the algorithm has to start with the smallest and most efficient classifiers, which reject a large number of negative results while detecting almost all positive instances. Then, more complex classifiers can be used to classify complicated cases. Due to this, the procedure becomes faster, so most of the non-face input images will be discarded by the first classifiers. A schematic description of the final classifier is shown in Figure 2.5.

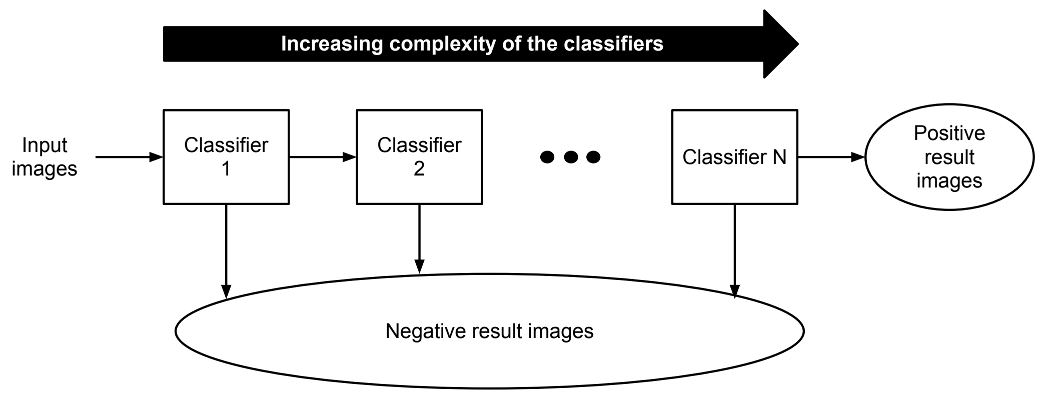

Figure 2.5: Schematic description of the Viola-Jones classifier.

2.3 Tracking

Object tracking is the technique that involves locating a moving object given consecutive video frames, and can be understood as an estimation of the object's true position based on previous measurements.

Tracking algorithms may be classified in two categories: algorithms based on filtering and data association, and algorithms relying on target representation and localization.

Filtering and data association algorithms assume that the object has an internal state which may be measured. By combining this measurement with the model of state evolution, the position of the object in the current frame is estimated. Kalman filter [13] is commonly used for this purpose in many different computer vision fields [14, 15, 16].

Target representation and localization algorithms, however, use a probabilistic model of the object appearance and try to detect this model in consecutive frames of the image sequence. Mean Shift algorithm [17] is a representative method in this category [18].

In this project a combination of Kalman filtering and Mean Shift based in the one proposed in [19] is used as the face tracking algorithm. As can be seen in Figure 2.6, Kalman filter is used for predicting the region in where the user face might be located in the current frame. Then, with the Viola-Jones algorithm, fast face detection is applied in this region. If the face detection algorithm does not match any face, a Mean Shift estimation is considered. The position given either by the face detector algorithm or the Mean Shift approach is used as the correction for the Kalman filter.

Section 2.3.1 details the Kalman filter algorithm and Section 2.3.2 describes Mean Shift algorithm. The combination method of both approaches is explained in Section 2.3.3.

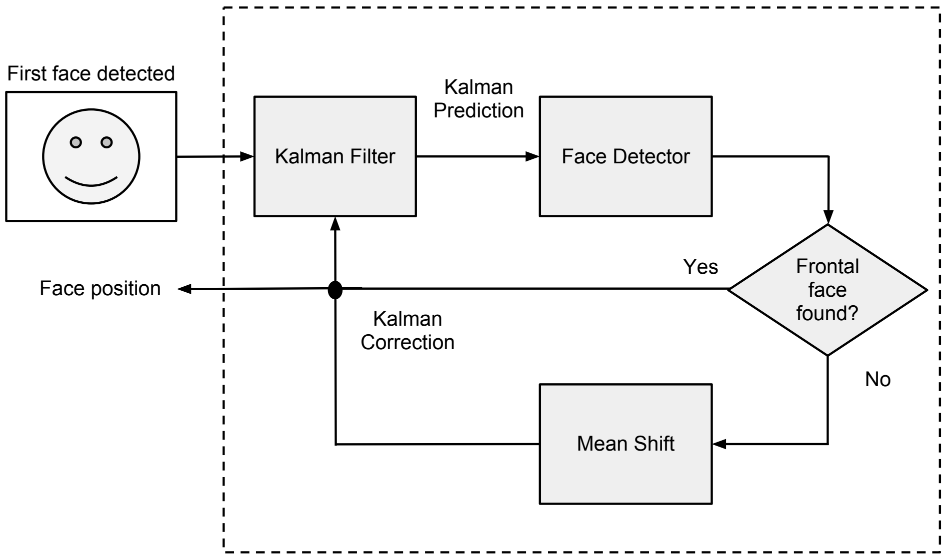

Figure 2.6: Scheme of the tracking system.

2.3.1 Kalman Filter

The Kalman filter (Kf), published by R.E. Kalman in [13], is a set of mathematical equations that provides estimation of unknown variables. It uses measurements observed over time, which usually contain noise [20], to predict a next state and make corrections on its predictions.

We assume that there is a discrete time system whose state at time n is given by vector $\mathbf{x}_n$. In the next time step, a linear operator can be applied to obtain the new state $\mathbf{x}_{n+1}$, as it is in Equation 2.1.

$$\mathbf{x}_{n+1} = \mathbf{F}_{n+1,n}\mathbf{x}_n + \mathbf{w}_{n+1} \tag{2.1}$$

where $\mathbf{F}_{n+1,n}$ is the transition matrix from state $\mathbf{x}_n$ to $\mathbf{x}_{n+1}$ and $\mathbf{w}_{n+1}$ is white Gaussian noise with zero mean and covariance matrix $\mathbf{Q}_{n+1}$.

Next, the measurement vector $\mathbf{z}_{n+1}$ is given by Equation 2.2.

$$\mathbf{z}_{n+1} = \mathbf{H}_{n+1}\mathbf{x}_{n+1} + \mathbf{v}_{n+1} \tag{2.2}$$

where $\mathbf{H}_{n+1}$ is the measurement matrix and $\mathbf{v}_{n+1}$ is white Gaussian noise with zero mean and covariance matrix $\mathbf{R}_{n+1}$.

Kf computes the minimum mean-square error estimate of the state $\mathbf{x}_k$ given the measurements $\mathbf{z}_1, ..., \mathbf{z}_k$. It is usually conceptualized as two phases as it is shown in Figure 2.7: the *Predict* phase and the *Correct* phase. In the *predict* phase, the previous state estimate is used to produce an estimate at the current time. In the *correct* phase, the current prediction and the current observation are combined to make corrections on the state estimate. Given:

- $\hat{\mathbf{x}}_n^-$, the a priori state estimate given observations up to time n.
- $\hat{\mathbf{x}}_n$, the a posteriori state estimate given measurements $\mathbf{z}_n$.
- $\mathbf{P}_n^- = E\left[(\mathbf{x}_n - \hat{\mathbf{x}}_n^-)(\mathbf{x}_n - \hat{\mathbf{x}}_n^-)^T\right]$, the a priori error covariance matrix.
- $\mathbf{P}_n = E\left[(\mathbf{x}_n - \hat{\mathbf{x}}_n)(\mathbf{x}_n - \hat{\mathbf{x}}_n)^T\right]$, the a posteriori error covariance matrix.
- $\mathbf{G}_n = \mathbf{P}_n^-\mathbf{H}_n^T\left[\mathbf{H}_n\mathbf{P}_n^-\mathbf{H}_n^T + \mathbf{R}_n\right]^{-1}$, the optimal Kalman Gain.

The procedure used in both of Kf phases is described in Algorithm 1.

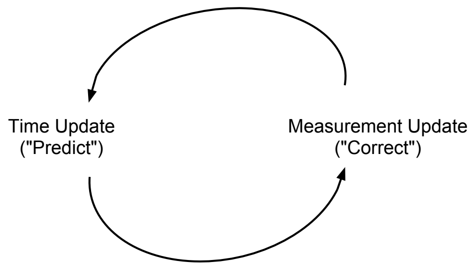

Figure 2.7: Cycles of Kalman filter.

Kalman filter applied to object tracking

Given a detected object at the discrete time t_n, we describe its spatial features as $[\mathbf{c}_n, \mathbf{v}_n, \mathbf{s}_n]$, where $\mathbf{c}_n = (c_x^n, c_y^n)^T$ is the center of the object, $\mathbf{v}_k = (v_x^n, v_y^n)^T$ its velocity and $\mathbf{s}_n = (w^n, h^n)^T$ its size.

In this case, the initial state of the Kalman filter $\hat{\mathbf{x}}_n^-$ is $\hat{\mathbf{x}}_0^- = (\mathbf{c}_0, \mathbf{v}_0, \mathbf{s}_0)^T$ and the transition matrix $\mathbf{F}$ is:

$$\mathbf{F}_{n+1,n} = \begin{pmatrix} 1 & \Delta T_{n+1} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In which the new center of the object at t_{n+1} is the same as at t_n plus $\mathbf{v}_n \cdot \Delta T_{n+1}$, where ΔT_{n+1} is the difference between t_{n+1} and t_n. However, its velocity and its size are supposed to be the same.

On the other hand, as the measurement vector is $\mathbf{z}_{n+1} = (\mathbf{c}'_{n+1}, \mathbf{s}'_{n+1})^T$, where $\mathbf{c}'_{n+1}$ and $\mathbf{s}'_{n+1}$ are the center and the size measurements of the object, respectively. Therefore, the measurement matrix $\mathbf{H}$ is:

$$\mathbf{H}_{n+1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

2.3.2 Mean Shift

Mean-Shift is an iterative procedure that tries to locate an object by finding the maximum of a density function given discrete data sampled from that

Algorithm 1 Kalman filter

1. Initialization:

$$\hat{\mathbf{x}}_0 = E\left[\mathbf{x}_0\right],$$
$$\mathbf{P}_0 = E\left[\left(\mathbf{x}_0 - E\left[\mathbf{x}_0\right]\right)\left(\mathbf{x}_0 - E\left[\mathbf{x}_0\right]\right)^T\right].$$

2. Prediction:

$$\hat{\mathbf{x}}_n^- = \mathbf{F}_{n,n-1}\hat{\mathbf{x}}_{n-1},$$
$$\mathbf{P}_n^- = \mathbf{F}_{n,n-1}\mathbf{P}_{n-1}\mathbf{F}_{n,n-1}^T + \mathbf{Q}_n,$$
$$\mathbf{G}_n = \mathbf{P}_n^-\mathbf{H}_n^T\left[\mathbf{H}_n\mathbf{P}_n^-\mathbf{H}_n^T + \mathbf{R}_n\right]^{-1}$$

3. Estimation:

$$\hat{\mathbf{x}}_n = \hat{\mathbf{x}}_n^- + \mathbf{G}_n\left(\mathbf{z}_n - \mathbf{H}_n\hat{\mathbf{x}}_n^-\right),$$
$$\mathbf{P}_n = \left(\mathbf{I} - \mathbf{G}_n\mathbf{H}_n\right)\mathbf{P}_n^-.$$

Goto the Prediction step for the next prediction.

function, as it is described in [17]. The basic aim of this algorithm is to characterize the target object by its appearance model and find an image region in which the matching of this model becomes maximum.

The probability density function (pdf) of the **target object** is approximated by a histogram of m bins, $\hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1,\dots,m}$. To form the histogram, only the pixels inside an ellipse surrounding the object are taken into account. The u-th histogram bin is given by:

$$\hat{q}_u = C_O \sum_{i=1}^{n} k\left(\|\mathbf{x}_i^*\|^2\right)\delta\left[b\left(\mathbf{x}_i^*\right) - u\right]$$

where:

- n is the number of pixels contained into the ellipse.
- $\mathbf{x}_i^*, i = 1, \dots, n$ is the normalized spatial location of the i-th pixel. The pixel locations are normalized by dividing the pixel's coordinates by the ellipse's semi-axes dimensions h_x and h_y. The center of the ellipse is assumed to be at the origin of the axes.

- $b : \Re^2 \rightarrow \{1, ..., m\}$ is a quantizing function which associates each pixel color in x_i with each bin u.
- C_O is a normalization factor such as $\sum_{u=1}^{m} \hat{q}_u = 1$.
- δ is the Kronecker delta function.
- $k(x), k : [0, \infty) \rightarrow \Re$ is the profile of the kernel applied to every pixel to increase the weight of the pixels closer to the center of the ellipse.

In the next frame the pdfs of the candidate objects at each candidate location $\mathbf{y}$ are also approximated by m-bin histograms. The u-th histogram bin at location $\mathbf{y}$ is given by:

$$\hat{p}_u(\mathbf{y}) = C_C^Y \sum_{i=1}^{n} k\left(\|\mathbf{y} - \mathbf{x'}_i\|^2\right) \delta\left[b\left(\mathbf{x'}_i\right) - u\right]$$

where $\mathbf{x'}_i, i = 1, ..., n$ is the normalized pixel's coordinates in the candidate ellipse, $\mathbf{x'}_i, i = 1, ..., n$ is the normalized pixel coordinates inside the target candidate ellipse and C_C^Y is a normalization factor such as $\sum_{u=1}^{m} \hat{p}_u(y) = 1$.

In order to compare the similarity between the target histogram and the candidate histogram at each location $\mathbf{y}$, the distance between $\hat{\mathbf{q}}$ and $\hat{\mathbf{p}}(\mathbf{y})$ is defined as:

$$d(y) = \sqrt{1 - \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right]}$$

where:

$$\rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathbf{y})\, \hat{q}_u} \tag{2.3}$$

is, in fact, the similarity function between $\hat{\mathbf{q}}$ and $\hat{\mathbf{p}}(\mathbf{y})$ and it is called Bhattacharyya coefficient [21].

To locate the object in the image, the distance between $\hat{\mathbf{q}}$ and $\hat{\mathbf{p}}(\mathbf{y})$ must be minimized, which is equivalent to maximize the Bhattacharyya coefficient. The ellipse center is initialized at a location $\hat{\mathbf{y}}_0$ which is the ellipse center in the previous image frame. Expression 2.3 can be approximated using linear Taylor approximation, resulting in:

$$\rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_\mathbf{0})\, \hat{q}_u} + \frac{C_C}{2} \sum_{u=1}^{n} w_i k\left(\|\mathbf{y} - \mathbf{x_i}\|^2\right),$$

where

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_\mathbf{0})}}\delta\left[b(\mathbf{x_i}) - u\right].$$

As the first term is independent from $\mathbf{y}$, only the second term must be maximized.

2.3.3 The tracking method

In this project we used a tracking system based in the one proposed in [19] as a combination of Kalman filter and Mean-Shift algorithm. The main idea is to find the position of the object (measurement in the previous frames) and forward it to Kf to estimate the current position (estimation).

Once a face is located, a prediction about the region of the image in where the face might be located in the next frame is done using the previous measurements informations and Kf. Then, the algorithm looks for detecting a face in the predicted region given by Kf either with Viola Jones algorithm or with Mean-Shift technique. When the face is obtained, measurements of both its size and center are used to correct the Kf prediction. The result is a iterative process as it is shown in Figure 2.6.

2.4 Skin pixel detection

As it is explained in the previous sections, the amount of blood present in the face skin produces variations in the reflected light due to the blood absorbency of the light. In order to minimize the inclusion of noise in the estimation coming from facial regions of non-skin, we implemented a skin pixel detection algorithm.

Skin detection can be defined as the process of selecting which pixels of an input image correspond to human skin. Generally, it is a pixel-level process involving a pre-processing of color space transformation. Skin pixel classification should provide coverage of all different skin types and cater for as many different lighting conditions as possible. In literature, several algorithms have been proposed for skin color pixel classification. They include piecewise linear classifier [22], [23], the Bayesian classifier with histogram technique [24], [25], Gaussian classifiers [26], [27] and the multilayer perceptron [28]. Also, different color spaces have been used in skin segmentation such as RGB, HSV or

YCbCr. The different techniques are studied and compared in [29].

As skin-pixel detection is not the main focus of this project, a simple and real time algorithm was implemented. For each frame, the selected ROI around the face of the user is converted into HSV (Hue, Saturation, Value) color space. Acording to [30, 31, 32] some robustness may be achieved via the use of luminance invariant color spaces as HSV. Based on this and our own empirical studies, we decided to use this color space.

In the skin pixel detection method that we used, firstly a color-histogram is calculated from the HSV transformed face image using hue and saturation channels. As most of the pixels in the detected face region correspond to skin pixels, by calculating its histogram we obtain an estimation of the skin color probability function. Using this estimation, the probability of each pixel of containing skin is obtained. Finally, a threshold on these probabilities is applied to create a skin mask. The threshold function used is:

$$I'(x,y) = \begin{cases} 1 & \text{if} \quad I(x,y) \geq th \\ 0 & \text{Otherwise} \end{cases}$$

where $I(x,y)$ is the value at the pixel location (x,y), $I'(x,y)$ is the new value at that pixel location and the threshold value th is chosen experimentally.

Some results obtained with this approach are shown in Figure 2.8. As it can be observed, areas which are too illuminated or too dark, like the ones around the nose, are eliminated by the skin segmentation algorithm. Although they are actually skin pixels, they do not contain real lightness information.

Figure 2.8: Examples of the skin detection algorithm.

CHAPTER

3

RECOVERY OF HEART RATE SIGNAL

Once the ROI of the video sequence is defined, color features from the video frames have to be extracted and processed in order to obtain a heart rate estimation. The procedure to recover the heart rate signal is explained in this chapter, as well as the techniques used in the process.

3.1 Introduction

As it is described in Chapter 2, for each frame of the video sequence we obtain a ROI of facial skin pixels. These selected pixels contain information of light fluctuations produced in the skin due to the variation in the amount of blood present in the human face. To extract this information, the ROI is separated into the three RGB (red, green, blue) channels and is spatially averaged over all skin pixels. Three color signals over time are obtained, as it is shown in Figure 3.1.

These RGB observed traces, also known as raw traces, are supposed to be a linear combination of some underlying source signals, one of which may be the heart rate signal. Independent Component Analysis (ICA) technique is

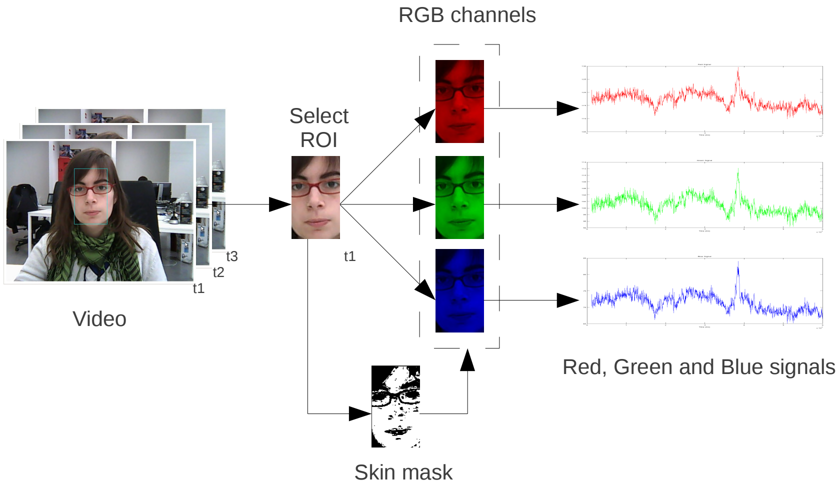

Figure 3.1: RGB traces recovery.

then used to recover these underlying source signals and estimate the heart rate signal.

In the recovery procedure, the raw traces are processed with the methodology shown in Figure 3.2. As it can be observed, first the raw RGB signals are *pre processed* and treated to become ready for their analysis. After that, they are decomposed into three independent source signals using the ICA approach. Source signals are then *post processed* in order to select the heart rate signal from them and obtain the heart rate estimation. Procedures used in the *pre processing*, ICA and *post processing* block are detailed in Section 3.2, Section 3.3 and Section 3.4, respectively.

3.2 Pre processing

The aim of the *pre processing* block is to transform the three raw traces obtained from the webcam recordings into three normalized traces that can be used in subsequent approaches of the process. At this point, two procedures are used: resampling and filtering.

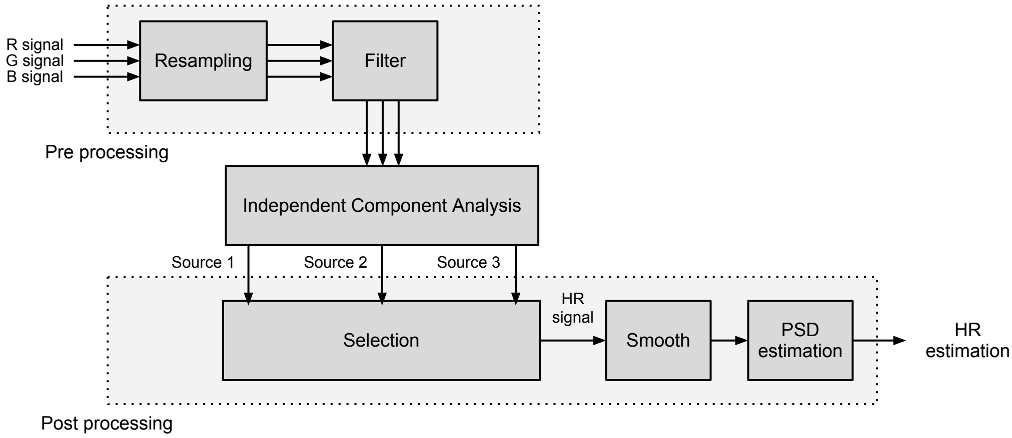

Figure 3.2: Heart Rate estimation recovery methodology.

3.2.1 Resampling

Each frame of the webcam recording is obtained at intrinsically irregular intervals, which means that the raw traces are unevenly sampled. However, some techniques used in the recovery methodology (including Independent Component Analysis explained in Section 3.3 and Power Spectral Density estimation described in Section 3.4.2) operate on time series with uniform intervals between samples. To apply these techniques, therefore, requires the raw traces to be *resampled* at uniform intervals. For this purpose, we use a linear interpolation approach to obtain an evenly sampled signal.

Having an unevenly sampled signal $y(t)$ and given a sampling frequency f_s, the aim is to find a time vector with uniform intervals $T_s = \frac{1}{f_s}$ between samples. Then, the k-th sample of our resampled $y'(t)$ signal is calculated at time $t_k = kT_s$ using linear interpolation approximation as:

$$y'(t_k) = y(t_n) + \frac{y(t_{n+1}) - y(t_n)}{t_{n+1} - t_n}\,(t_k - t_n)$$

where $y(t_n)$ and $y(t_{n+1})$ are the previous and the following data points of $y(t_k)$ in the original signal, respectively.

3.2.2 Filtering

In order to focus our RGB signals in the fast variations produced in it and not in the slow variations coming from noisy low frequency sources, the observed signals are detrended and normalized as follows:

$$y'_i(t) = \frac{y_i(t) - \mu_i(t)}{\sigma_i}$$

where $\mu_i(t)$ and σ_i are the local mean at t and the standard deviation of $y_i(t)$, respectively. Figure 3.3 shows an example of three RGB signals before and after being detrended.

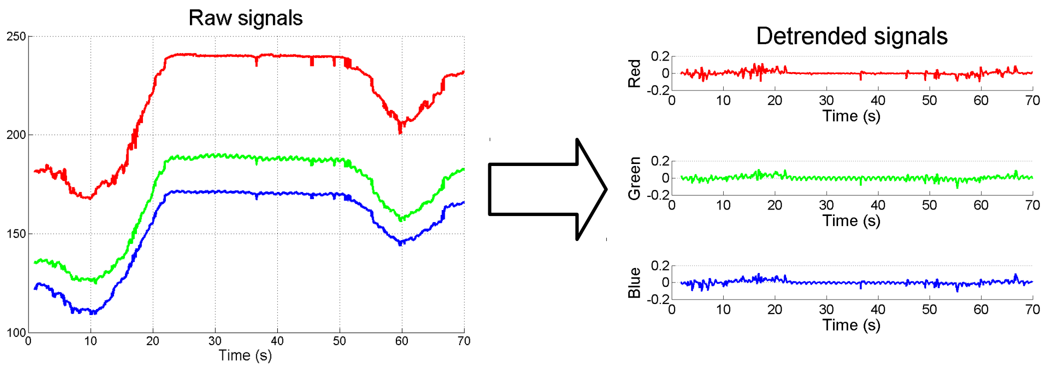

Figure 3.3: Example of three RGB signals before and after being detrended.

The normalized traces are then bandpass-filtered with a FIR filter with 128 coefficients with 0.33 Hz and 4 Hz as the lower and the upper cutoff frequencies, respectively. It corresponds with the range of 20-240 bpms, which is the frequency range of interest in the heart rate measurements. The frequency response of the filter, in bpm, can be seen in Figure 3.4.

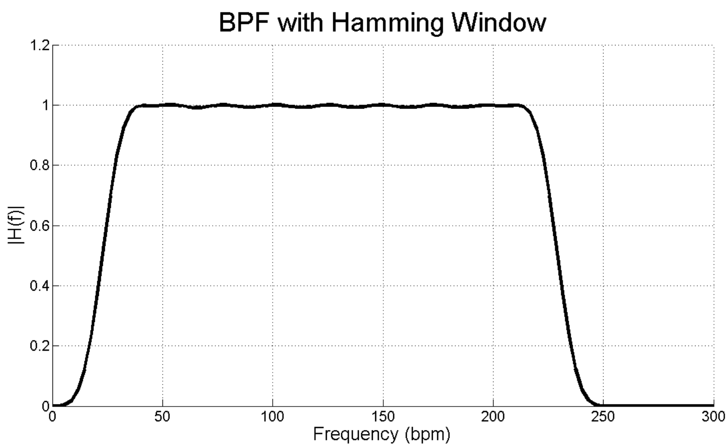

Figure 3.4: Frequency response of the bandpass filter.

3.3 Independent Component Analysis

Independent Component Analysis (ICA) [33] is a method for Blind Source Separation (BSS) [34]. BSS is a procedure involving the separation of a set of independent signals from a set of mixed signals, where the mixing process is unknown.

To understand in a better way the problem of BSS, the *cocktail party* classical example is related in 3.3.1 before presenting the formal definition of ICA in 3.3.2.

3.3.1 Cocktail Party Problem

The *cocktail party problem* is a classical example of BSS [35]. Imagine you are in a room with a large number of people talking simultaneously, like in a cocktail party. Despite there are a large number of acoustic signals in that room, in general people can follow properly one of the discussions. This is because the human brain can handle the auditory source separation problem, but in fact, it is a very tricky problem in digital signal processing.

Several approaches have been proposed for the solution of blind source separation, some of the most successful are principal components analysis (PCA) [36] and ICA [33]. The key for solving the problem is to have several independent mixture signals, like in the case of the human ear, in which the mixture of signals obtained from the right ear is different from the mixture of signals obtained from the left ear.

3.3.2 Formal definition

If we have $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_m)^T$ zero-mean m-dimensional random variables that are, in fact, a linear mixture from $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_n)^T$ n-dimensional independent sources, the linear transformation of the source variables can be expressed as:

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{3.1}$$

where $\mathbf{A}$ is the $m \times n$ mixture matrix that contains the mixture coefficients. It is usually assumed that the number of sources is equal to the number of mixture signals, so $m = n$. In this context, both $\mathbf{A}$ and $\mathbf{s}$ are unknown. The aim of ICA is to obtain a constant matrix $\mathbf{W}$, which is the inverse of the estimate of the matrix $\mathbf{A}$, so that the linear transformation of the observed variables become:

$$\hat{\mathbf{s}} = \mathbf{W}\mathbf{x} \tag{3.2}$$

being $\hat{\mathbf{s}}$ an estimate of the vector $\mathbf{s}$ containing the source signals.

ICA finds the independent components by maximizing the statistical independence of the source components. However, there exist many ways to express independence. Motivated by the central limit theorem (CLT), one way of measuring independence is the quantification of non-Gaussianity. The CLT [37] states that the sum of many independent random variables, each with finite mean and variance, will tend to be distributed according to the normal distribution. That means that the observed signals x_i are *more gaussian* than sources signals s_i. Therefore, separating the mixed signals is understood as moving away from the normal distribution. Because of that, one of the requirements of ICA algorithms is that sources signals cannot be gaussian sources.

Mutual information is a mathematical tool used for measuring the non-gaussianity of n random variables $y_i, i = 1, ..., n$, which measures, in fact, the independence of that variables. The matrix $\mathbf{W}$ in Equation 3.2 is then determined so that the mutual information of the transformed components s_i is minimized.

Though ICA is a robust mathematical tool, there are three ambiguities we should consider. The first one is that the variances of the independent components cannot be determined, so the amplitude of the heart rate signal recovered might not be the real signal amplitude. That is the reason why some times the signal waveform is not recovered properly. Another ambiguity is that the order in which ICA implies the signals is random. That involves that we must include an algorithm for selecting the signal of interest among the three possible signals. Also, the sign of the independent components is not specified, which is not a big drawback for our purposes.

3.4 Post processing

In the *post processing* stage the three separated signals obtained from the ICA are processed in order to get the estimated heart rate.

3.4.1 Component selection algorithm

After applying ICA technique, three independent components are obtained. However, as it was mentioned before, the order in which ICA returns the independent component is random and cannot be predicted. Because of that, a component selection algorithm to determine which is the heart rate signal must be implemented.

As one of the three components is expected to be the heart rate signal, which has a strong periodicity, the method for choosing the signal of interest consists on selecting the component whose normalized power spectrum contains the highest peak. An example of this procedure using real samples is shown in Figure 3.5. As it can be seen, the first and the second sources of that example do not present an outstanding frequency peak as the third one, which is supposed to be the heart rate signal, does.

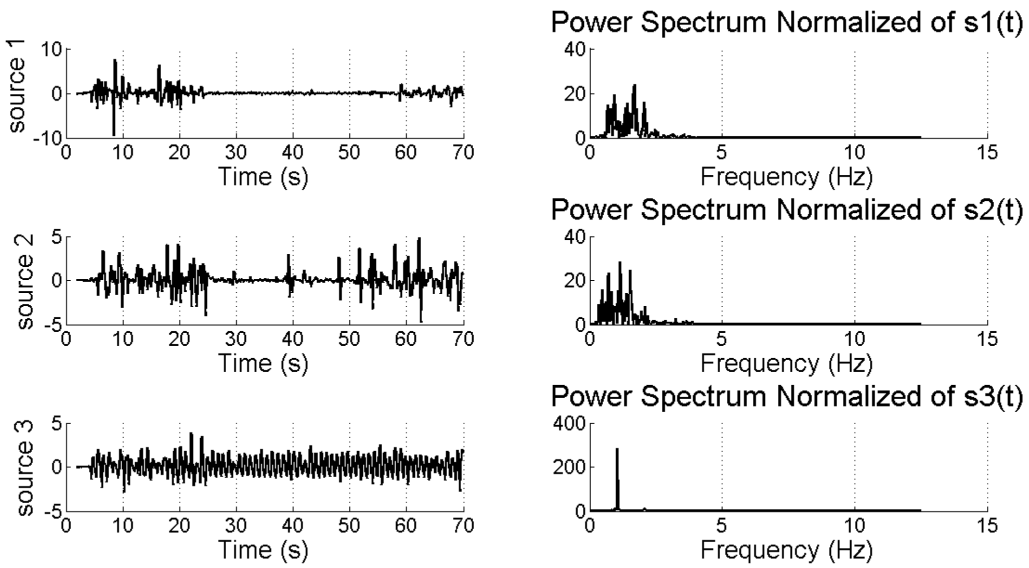

Figure 3.5: Real sample example of the component selection method. In this case, the selected source is the third one.

In order to leave out noise, the separated source signal is then smoothed using a five-point moving average filter.

3.4.2 Spectral density estimation

Power Spectral Density (PSD) is used to obtain the power spectrum of the heart signal, which characterizes the frequency content of that signal. PSD of a $x(n)$ signal is defined as:

$$P_x\left(e^{jw}\right) = \sum_{k=-\infty}^{\infty} r_x\left(k\right) e^{jkw}$$

with

$$r_x\left(k\right) = \lim_{N\to\infty}\left\{\frac{1}{2N+1}\sum_{n=-N}^{N} x\left(n+k\right) x^*\left(n\right)\right\}$$

As our measured signal $x(n)$ has a finite number of samples, we have to obtain an estimation of the PSD. Techniques for spectrum estimation can be divided into two categories: parametric methods and non-parametric methods. Parametric approaches assume that the underlying stationary stochastic process has a certain structure which can be described using a small number of parameters while non-parametric approaches explicitly estimate the covariance or the spectrum of the process without assuming that the process has any particular structure. Periodogram and Welch's methods [38] are both examples of non-parametric approaches. By contrast, Autoregressive models [39] can be used as a parametric spectrum estimation method.

Periodogram

The periodogram is one of the most commonly used non-parametric spectrum estimation techniques. A windowed segment of samples from the process $x(n)$ is taken as $x_M(n) = x(n)w(n)$, where the window function w, classically the rectangular window, contains M nonzero samples. Its autocorrelation estimate is:

$$\hat{r}_x\left(k\right) = \frac{1}{M}\sum_{n=-\infty}^{\infty} x_M\left(n+k\right) x_M^*\left(n\right) = \frac{1}{N} x_M\left(k\right) * x_M^*\left(-k\right)$$

The periodogram is defined as the Fourier transform of this autocorrelation estimate, so:

$$\hat{P}_{periodogram}\left(e^{jw}\right) = \frac{1}{M} X_M\left(e^{jw}\right) X_M^*\left(e^{jw}\right) = \frac{1}{M}\left\|X_M\left(e^{jw}\right)\right\|^2$$

where $X_M(e^{jw}) = \sum_{n=0}^{M-1} x_M(n)\, e^{-jnw}$ is the Fourier transform of the windowed signal.

The main advantage of the periodogram is that it is a fast spectrum estimator. However, using the whole signals for the estimation might results in a noisy measure.

Welch's method

Welch's method [38], named after P.D. Welch, is based on the concept of using an average of modified periodograms. In this case, the data $x(n)$ of length N is partitioned in segments of length L, possibly overlapping and with successive segments starting at D samples of difference. Then, the i-th segment is $x_i(n) = x(n + iD)$ with $i = 0, ..., L - 1$.

Suppose we have K segments and that $N = L + (K - 1)D$ like in Figure 3.6. For each segment a modified periodogram, which is a periodogram with a non rectangular window, is calculated. The spectral estimate is then the average of these periodograms:

$$\hat{P}_{\text{Welch}}\left(e^{jw}\right) = \frac{1}{KLU} \sum_{i=1}^{K} \left| \sum_{n=0}^{L-1} w(n) x(n + iD) e^{-jnw} \right|^2$$

with

$$U = \frac{1}{L} \sum_{n=0}^{L-1} |w(n)|^2$$

Welch's method reduces noise in the estimated power spectra compared to the periodogram technique despite having less frequency resolution due to the use of less samples for each window.

Autoregressive model

Autoregressive (AR) model is a type of random process which attempt to predict an output of a system, X_t, based only on the previous outputs [39]. The autoregressive model of order p, AR(p), is defined as:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t$$

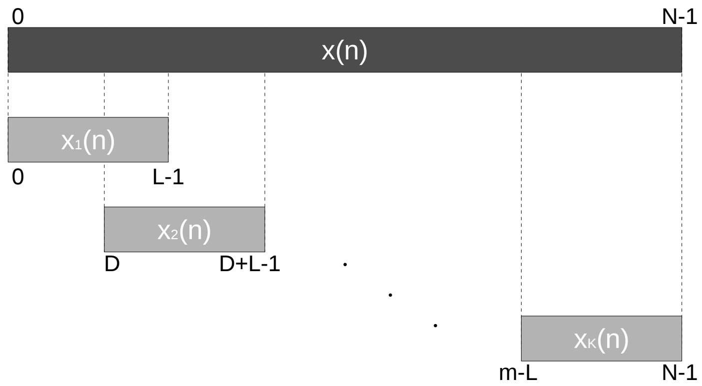

Figure 3.6: Data segmentation in the Welch's method.

where $\varphi_1, ...\varphi_p$ are the parameters of the model, c is a constant often omitted and ε_t is white noise. The parameters of the model can be determined from the autocorrelation function of the process using Yule-Walker equations, which are:

$$\gamma_m = \sum_{k=1}^{p} \varphi_k \gamma_{m-k} + \sigma_\epsilon^2 \delta_{m,0} \qquad m = 0, ..., p$$

where γ_m is the autocorrelation function of X, σ_ϵ is the standard deviation of the input noise process and $\delta_{m,0}$ is the Kronecker delta function. As the last part of the equation is non-zero only if $m = 0$, the equation is usually solved by representing it as a matrix for $m > 0$:

$$\begin{bmatrix} \gamma_0 & \gamma_{-1} & \gamma_{-2} & \cdots \\ \gamma_1 & \gamma_0 & \gamma_{-1} & \cdots \\ \gamma_2 & \gamma_1 & \gamma_0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \end{bmatrix}$$

For $m = 0$,

$$\gamma_0 = \sum_{k=1}^{p} \varphi_k \gamma_{-k} + \sigma_\epsilon^2$$

which allows to solve σ_ϵ^2.

The power spectrum of an AR(p) process with noise variance σ_Z^2 is [39]:

$$\hat{P}_{AR}\left(e^{jw}\right) = \frac{\sigma_Z^2}{|1 - \sum_{k=1}^{p} \varphi_k \gamma_k e^{-jkw}|^2}$$

AR models are based in pole models and that is why they can estimate very well the spectrum peaks of our signal. However, the order of the filter, p, has to be chosen properly to have a correct estimation. If the order is smaller than the optimum order, spectral peaks will become wider. In the other way round, if the order of the filter is higher, spurious spectral peaks would appear in the estimation.

CHAPTER

4

BOOTSTRAP

In order to analyze the behavior of the heart rate estimator system, several experiments were performed. The results of that experiments were then thoroughly analyzed and compared with statistical tools. In this chapter, Bootstrap techniques for analyzing a particular estimator are presented.

4.1 Introduction

The bootstrap [40] is a general methodology used to calculate different estimators from data, especially when the theoretical distribution is unknown or when the sample size available is insufficient. This technique can provide accurate answers where classical approaches cannot be used and also presents similar results when other techniques may be applied [41].

Generally, when we have a estimator $\hat{\theta}$ of an unknown parameter θ, we would like to obtain some measures for its accuracy, such as mean, standard error, bias or confidence intervals. Some of these parameters can be calculated analytically, however, when we want to calculate some complicated statistic or we cannot make assumptions of the probability distribution of the samples, the bootstrap method can be used.

The bootstrap method consists on estimating the sampling distribution F on the basis of the observed data instead of assuming a mathematical approach like, for example, the normal curve. It is done by creating new data sets, called bootstrap samples, from the original data set by resampling it with replacement. Each bootstrap sample has the same size of the original data. By performing a sufficient number of times this procedure a good estimate of the probability distribution of the original data, $\hat{F}$, is obtained, and therefore the statistics of interest can be calculated in the usual way.

For example, suppose we have 41 heart rate measurements coming from both a physiological monitor device and our webcam estimator system. We call $\mathbf{h_R} = (h_{R_1}, h_{R_2}, ..., h_{R_{41}})$ the vector that contains the 41 monitor heart rate measurements, and $\mathbf{h_O} = (h_{O_1}, h_{O_2}, ..., h_{O_{41}})$ the vector that contains the 41 webcam heart rate estimations, being h_{R_j} and h_{O_j} data coming from the same measurement, as it is summarized in Table 4.1.

MeasureID	HR monitor	HR webcam
1	73.8281	63.2813
2	46.5820	46.4063
3	67.6758	63.9844
4	63.2813	63.2813
5	43.9453	44.2969
⋮	⋮	⋮
40	66.7969	66.7969
41	61.5234	63.9844

Table 4.1: Original data consisting in 41 measurements.

Let's imagine that we want to obtain some statistics of the absolute error of this data, $|\mathbf{e}| = |\mathbf{h_R} - \mathbf{h_O}|$, and we decide to use the bootstrap technique. Figure 4.1 displays the distribution of the data's absolute error as well as each measurement's absolute error.

In order to apply the bootstrap technique, first of all we have to create B new bootstrap samples by resampling the original data set with replacement, as it is shown in Table 4.2, where $B = 1000$.

The bootstrap technique exists in a nonparametric and a parametric version. In the nonparametric version, the underlying probability distribution is estimated from the empirical distribution of the data. In the parametric boot-

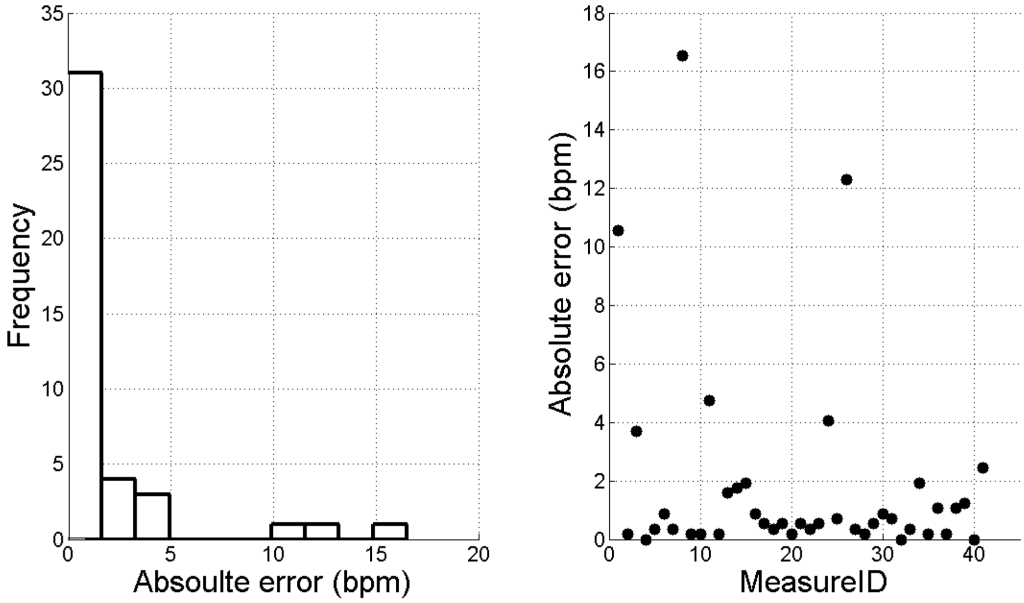

Figure 4.1: Frequency of the absolute error (left) and absolute error of each measurement (right).

strap, a parametric model is used for estimating the sampling distribution. As the amount of data samples available in this project is not very large, a nonparametric version of the bootstrap method is used in order to analyze the statistics of the data obtained in our experiments.

In the next sections, procedures for estimating the mean (Section 4.2), the standard error (Section 4.3) and the confidence interval (Section 4.4) are detailed.

Bootstrap sample	Measurements
1	$\binom{h_{R_{21}}}{h_{O_{21}}}, \binom{h_{R_{23}}}{h_{O_{23}}}, \binom{h_{R_4}}{h_{O_4}}, ..., \binom{h_{R_{37}}}{h_{O_{37}}}$
2	$\binom{h_{R_{39}}}{h_{O_{39}}}, \binom{h_{R_1}}{h_{O_1}}, \binom{h_{R_{28}}}{h_{O_{28}}}, ..., \binom{h_{R_1}}{h_{O_1}}$
$\vdots$	$\vdots$
1000	$\binom{h_{R_9}}{h_{O_9}}, \binom{h_{R_{16}}}{h_{O_{16}}}, \binom{h_{R_{40}}}{h_{O_{40}}}, ..., \binom{h_{R_{37}}}{h_{O_{37}}}$

Table 4.2: Example of the creation of 1000 bootstrap samples by sampling from the original data with replacement. Each bootstrap sample contains the same data length as the original data.

4.2 Bootstrap estimates of the mean

The mean of a random variable is the average of all its possible values, weighted by its probabilities. It is usually very easy to compute.

Having the observed data $\mathbf{x} = (x_1, x_2, ..., x_n)$ consisting of independent observations $X_1, X_2, ..., X_n \sim_{ind} F$, where F represents an unknown probability distribution of the samples, the bootstrap estimation of the mean involves obtaining a large number (B) of bootstrap samples, each one denoted as $\mathbf{x}_b^* = (x_1^*, x_2^*, ..., x_n^*)\,, b = 1, 2, ..., B$. For each $\mathbf{x}_b^*$, the mean of the data is evaluated. The final bootstrap estimated mean is the average of all the $\bar{m}_b, b = 1, 2, ..., B$.

Following the example described in Section 4.1, we now compute the mean absolute error (MAE) of our data using the bootstrap technique. Table 4.3 summarizes the bootstrap estimates of the mean using $B = 1000$ bootstrap samples. It can be observed that for each bootstrap sample a mean absolute error, $\bar{m}_b$, is obtained. The average of all these MAEs is the bootstrap estimated MAE, which results very similar to the standard estimated MAE. Figure 4.2 shows the bootstrap distribution of the mean, as well as the bootstrap estimated MAE and the standard estimated MAE.

	Original data	Bootstrap samples					
	$\lvert\mathbf{e}\rvert$	$\lvert\mathbf{e}\rvert_1^*$	$\lvert\mathbf{e}\rvert_2^*$	$\lvert\mathbf{e}\rvert_3^*$	$\lvert\mathbf{e}\rvert_4^*$	...	$\lvert\mathbf{e}\rvert_{1000}^*$
$\bar{m}_b$		1.2862	1.9250	1.1876	1.7450	...	2.0665
MAE	**1.8307**	**1.8150**					

Table 4.3: Example of the bootstrap estimates of the mean. We can observe that the MAE obtained with the bootstrap technique is very similar to the MAE obtained with the standard procedure.

4.3 Bootstrap estimates of standard error

Standard error (SE) is the standard deviation of the sampling distribution of a statistic and it is an important measure of the statistic accuracy because it reflects how much fluctuation it may show.

Suppose that we have a statistic of interest, $\theta(\mathbf{x})$, and we want to estimate its standard error with the bootstrap method. The standard error $(\sigma(F))$ of

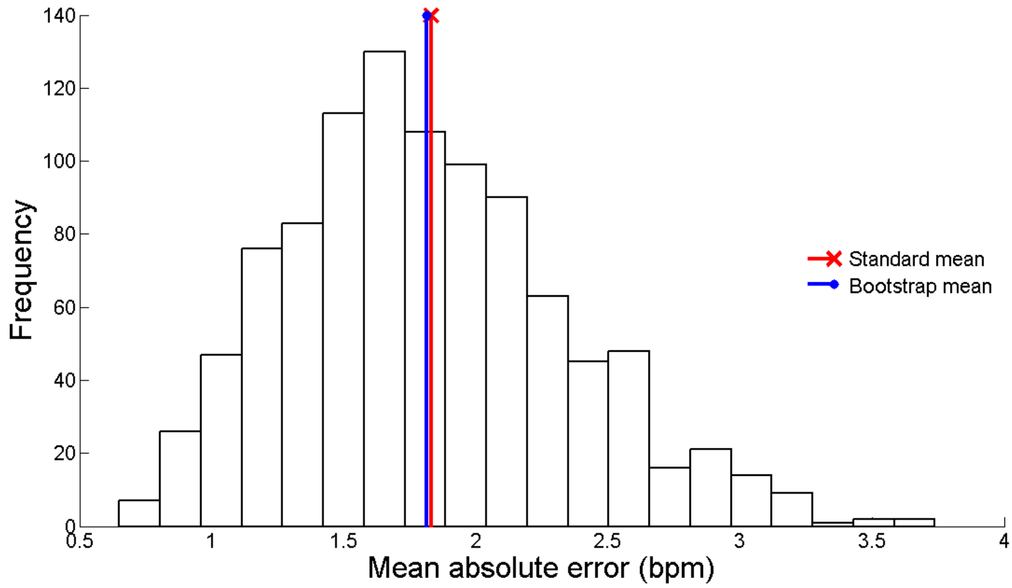

Figure 4.2: Distribution of 1000 MAE from bootstrap samples.

the estimate statistic $\hat{\theta}$ is a function of the unknown sampling distribution F. Using the bootstrap estimation of the sampling distribution $\hat{F}$, we can estimate the standard error as $\hat{\sigma} = \sigma(\hat{F})$. The algorithm for estimating the standard error is detailed in Algorithm 2.

Algorithm 2 Bootstrap estimates of the standard error.

1. Obtain a large number of bootstrap samples $\mathbf{x}_1^*, \mathbf{x}_2^*, ..., \mathbf{x}_B^*$.
2. Evaluate the statistic of interest $\hat{\theta}^*(b) = \hat{\theta}(\mathbf{x}_b^*), b = 1, 2, ..., B$ for each bootstrap sample $\mathbf{x}_b^*$.
3. Calculate the sample standard deviation of the $\hat{\theta}^*(b)$ values.

To compute the standard error of the MAE obtained in Section 4.2 using bootstrap, we have to calculate the standard deviation of the B bootstrap MAE, $\bar{m}_b$. A comparison between the results obtained using the standard procedure and the bootstrap technique is shown in Table 4.4.

The bootstrap method of estimating the standard error of a statistic becomes very valuable in those cases where there is not a theoretical formula for

	Original data	Bootstrap samples					
	$\lvert\mathbf{e}\rvert$	$\lvert\mathbf{e}\rvert_1^*$	$\lvert\mathbf{e}\rvert_2^*$	$\lvert\mathbf{e}\rvert_3^*$	$\lvert\mathbf{e}\rvert_4^*$	...	$\lvert\mathbf{e}\rvert_{1000}^*$
$\bar{m}_b$		1.2862	1.9250	1.1876	1.7450	...	2.0665
MAE	1.8307	1.8150					
SE	**0.5406**	**0.5179**					

Table 4.4: Example of the bootstrap estimates of the standard error.

computing the standard error.

4.4 Bootstrap confidence intervals

A confidence interval (CI) for a statistic of interest, θ, consists in a pair of numbers between which is supposed to be the true value of θ with a certain probability. Confidence intervals are usually more informative about a statistic than a single point estimate.

A confidence interval for θ is usually obtained from the estimator $\hat{\theta}$ by considering the probability distribution of $\hat{\theta} - \theta$. Denoting the α-percentile of the distribution of $\hat{\theta} - \theta$ as s_α, the confidence interval for θ is:

$$\hat{\theta} - s_{1-\alpha/2} \leq \theta \leq \hat{\theta} - s_{\alpha/2}$$

and the probability of containing the true unknown value of θ is equal to $1 - \alpha$.

Bootstrap methods for estimating confidence intervals use percentiles of the bootstrap samples distribution. The sample α-percentile of the bootstrap distribution is given by the $(B)\alpha$-th ordered estimator value (being the ordination as $\hat{\theta}^*(1) \leq \hat{\theta}^*(2) \leq ... \leq \hat{\theta}^*(B)$). For example, the $\alpha = 0.025$ and $\alpha = 0.975$ percentiles of a sample of size $B = 2000$ are the 50th and 1950th elements of the sorted bootstrap samples, respectively.

Several bootstrap methods have been proposed for constructing confidence intervals and many different techniques are available for this purpose. The differences between the bootstrap methods arise from the choice of the statistic, or the way to convert percentile to confidence intervals.

Basic bootstrap confidence intervals: basic bootstrap confidence intervals are based on that the distribution of $\hat{\theta} - \theta$ is approximated by that of

$\hat{\theta}^* - \hat{\theta}$. Thus, the percentiles of s_α of the distribution of $\hat{\theta} - \theta$ are replaced by the quantiles of the bootstrap approximation. Then, the basic confidence interval is:

$$\hat{\theta} - s^*_{B(1-\alpha/2)} \leq \theta \leq \hat{\theta} - s^*_{B(\alpha/2)}$$

Studentised bootstrap (bootstrap-t) confidence intervals: studentised bootstrap confidence intervals are constructed from the studentised statistic $\frac{\hat{\theta}-\theta}{\hat{\sigma}}$ by approximating $\hat{\theta}$ and $\hat{\sigma}$ with the bootstrap distribution. For each bootstrap sample, it is calculated:

$$t^*_b = \frac{\hat{\theta}^*_b - \hat{\theta}}{\hat{\sigma}^*_b} \tag{4.1}$$

where $\hat{\theta}^*_b$ is the estimate from the b-th bootstrap sample and $\hat{\sigma}^*_b$ an estimate for the standard error from the same sample.

The percentiles $t_{\alpha/2}$ and $t_{1-\alpha/2}$ of the distribution $\frac{\hat{\theta}-\theta}{\hat{\sigma}}$ are approximated by the $t^*_{B(\alpha/2)}$ and $t^*_{B(1-\alpha/2)}$, respectively. Finally, the studentised bootstrap confidence interval is:

$$\hat{\theta} - t^*_{B(1-\alpha/2)}\hat{\sigma} \leq \theta \leq \hat{\theta} - t^*_{B(\alpha/2)}\hat{\sigma}$$

In classical approaches, either the distribution of $\hat{\theta} - \theta$ or $\frac{\hat{\theta}-\theta}{\hat{\sigma}}$ are usually approximated by, for instance, a normal distribution. The bootstrap approximation correctly captures the skewness of $\hat{\sigma}$ and uses it to obtain confidence intervals for θ, which are not necessarily symmetric around the point estimate $\hat{\theta}$.

Following the example that we have been developing along this chapter, we are now computing the confidence interval of the MAE obtained in Section 4.2 with $\alpha = 0.05$. For the basic confidence interval, we have to sort the bootstrap sample means $\bar{m}_b$ and choose the ones corresponding to $B\alpha/2$-th and to $B(1-\alpha/2)$-th of the sorted bootstrap samples.

On the other hand, if we want to calculate the studentised confidence interval, firstly we have to calculate $t^*_b, b = 1, 2, ..., B$ as in 4.1. Then, the B t^*_b values are sorted in order to calculated the minimum and the maximum point of the CI as:

$$CI_{min} = MAE - \mathbf{t^*_{sorted}}(B(1-\alpha/2))\hat{\sigma}$$

$$CI_{max} = MAE - \mathbf{t}^*_{\mathbf{sorted}}(B(\alpha/2))\hat{\sigma}$$

Table 4.5 shows a brief summary of the parameters that are involved in the process of the bootstrap estimates of the confidence intervals, whereas the studentised confidence interval of the data bootstrap distribution can be seen in Figure 4.3, where it can be seen that the bootstrap method correctly captures the skewness of the empirical distribution and the confidence interval is not symmetric around the estimated MAE.

	Bootstrap samples					
	$\lvert\mathbf{e}\rvert^*_1$	$\lvert\mathbf{e}\rvert^*_2$	$\lvert\mathbf{e}\rvert^*_3$	$\lvert\mathbf{e}\rvert^*_4$	...	$\lvert\mathbf{e}\rvert^*_{1000}$
$\bar{m}_b$	1.2862	1.9250	1.1876	1.7450	...	2.0665
$\hat{\sigma}^*_b$	1.1876	1.7450	1.5263	2.0665	...	2.0451
MAE	1.8150					
$\hat{\sigma}$	0.5179					
t^*_b	-0.2015	-0.0214	-0.1187	0.0638	...	0.1646
$\bar{m}_b$ sorted	0.5959	0.6002	0.6302	0.6474	...	3.6228
t^*_b sorted	-1.4595	-1.2578	-1.1387	-1.0957	...	0.3399
Basic CI	$CI_{min} = 0.9132$			$CI_{max} = 2.8982$		
Studentised CI	$CI_{min} = 1.6852$			$CI_{max} = 2.1874$		

Table 4.5: Example of the bootstrap estimates of confidence intervals.

The different methods for performing confidence intervals are described and compared in [42]. Although the studentised interval estimates approach has a higher computational cost, it is considered to be the most reliable method, and for this reason it was used for the analysis of the data obtained in this project.

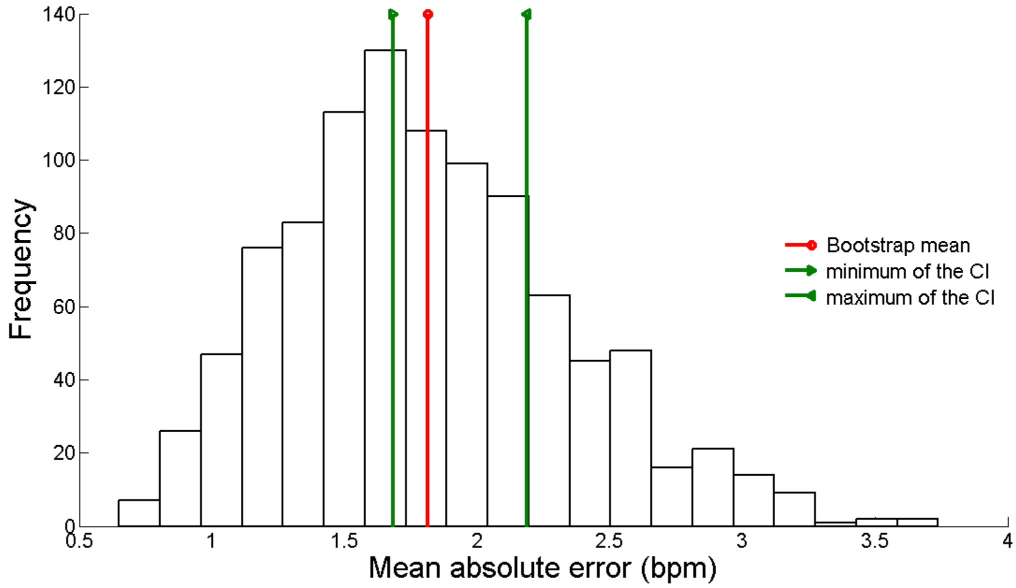

Figure 4.3: Example of the studentised bootstrap confidence interval. It can be seen that the bootstrap method correctly captures the skewness of the empirical distribution, therefore the bootstrap CI is not symmetric around the estimated MAE.

CHAPTER

5

DESIGN OF THE HEART RATE ESTIMATION SYSTEM

As it is explained in the first chapter, the aim of this project is to design and implement a real-time heart rate estimator using facial information from a standard webcam. To select the final configuration of the application among the parameters involved in the process described in Chapter 3, several tests had to be performed. In this chapter the results obtained in these tests are explained.

5.1 Introduction

In literature, there exist several methods for measuring the heart rate of a person using non-contact approaches. Typically, these techniques used dedicated infra-red light sources, although recent work has shown that the heart rate can be obtained using only ambient light [1, 2].

Following the methodology described in [4] and [5], we designed a heart rate estimator system in two separated phases. The first one was the image processing phase, which includes the procedure described in Chapter 2. In this phase, the face of the user is detected along the image region and those pixels that come from a facial skin region are selected. The second phase was the data processing module, which processes the data coming from the video image using the scheme presented in Chapter 3, until a heart rate estimation is obtained.

The design of the heart rate estimator system involves varying the second of the two phases, in order to select the parameters that offer the most accurate results. To achieve that, a set of tests in which these parameters were modified, was performed in MATLAB. The results of the test were then statistically analyzed with bootstrap techniques (which are explained in Chapter 4).

This chapter is divided in the following sections: both the perfomance set-up of the experiments and the parameters we took under test are explained in Section 5.2, whereas the results of the tests are presented in Section 5.3. Finally, some examples of how the system works are shown in Section 5.4.

5.2 Design of the system

As it is mentioned before, several tests were implemented in MATLAB in order to select the configuration which gives a better estimation in the heart rate recovery procedure. To perform the tests, however, firstly we had to collect enough data information, so a set of 41 videos were recorded to 30 volunteers.

5.2.1 Experiments set-up

All the videos were recorded in a office room using a standard video camera that captures frames at 20 frames per second (fps), approximately, in color (24-bit RGB with 3 channels $\times$ 8 bits/channel) and with pixel resolution of 640×480. The videos were recorded at 0.5 meters of distance from the subjects, approximately, and they were saved in an uncompressed AVI format. Volunteers were asked to remain still and wear a finger pulse sensor in their index finger while the video was being recorded. The physiological device used (HRS-06UF USB Finger-clip Pulse Wave Sensor [43]) works at the sampling frequency of 300 hertz. It was connected to a laptop through an USB port and its measurements, used as a heart rate signal reference, were stored. Both video recordings and finger pulse data traces were 70 seconds long traces. For each one of the videos, the three RGB color traces were obtained with the procedure described in Section 3.1.

5.2.2 Parameters under test

The posterior processing procedure of the three RGB traces (explained in detail in Chapter 2) involves three main modules, as it is shown in Figure 5.1:

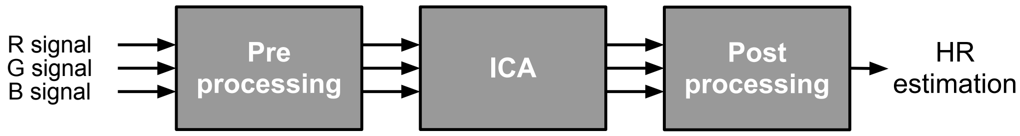

Figure 5.1: Main modules involved in the recovery of the heart rate signal.

1. Pre-processing.
2. Independent Component Analysis.
3. Post-processing.

It was only in the first and the last modules where some of the parameters were changed during tests performance. The details of which parameters were taken under consideration are described in the next paragraphs.

Pre-processing: In this module the color signals are prepared for the subsequent process, in which two parameters are tested.

1. *Video length.* Videos were cut after 10, 15, 25, 45 and 60 seconds in order to test how the length of the sample affects the quality of the estimation.
2. *Sampling frequency (f_s).* We tested sampling frequencies of 8, 16, 24 and 32 hertz.

Post-processing: In this block, the final techniques of the recovery procedure are done. In our tests we included the parameters involved with:

1. *Spectrum estimation technique.* Three different methods were tested: the Periodogram, Welch's Method and Autoregressive Models. Moreover, in each one of these methods, some of its parameters were also tested:
   - Periodogram: 512, 1024 and 2048 samples are taken in the Fourier transform (FT).

- Welch's method: windows of 5, 10, 15 and 20 seconds with none, 15, 30, 60 and 75 % overlap are taken from the whole trace. Also 512, 1024 and 2048 samples are taken in the FT.
- AR models: different orders of the AR models such as 2, 4, 8, 16, 32 and 64 were tested.

A summary of the parameters used in our tests is shown in Table 5.1, which relates each parameter with its possible values.

Parameter	Values	Module
Sampling frequency	[8, 16, 24, 32] Hz	Pre processing
Video length	[10, 15, 25, 45, 60] s	
Spectral estimation method		Post processing
Periodogram		
Samples Fourier transform	[512, 1024, 2048]	
Welch's Method		
Samples Fourier transform	[512, 1024, 2048]	
Window length	[5, 10, 15, 20] s	
Overlap	[0, 15, 30, 60, 75] %	
AR models		
Order	[2, 4, 8, 16, 32, 64]	

Table 5.1: Parameters taken under test and its possible values.

In the MATLAB tests, each group of RGB traces from each video was processed with all the possible configurations, so 41 heart rate estimations were obtained for each set of parameters. These heart rate estimations were compared to the measurements obtained with the physiological sensor device, which were called *real measurements*, in order to measure the error produced in each estimation.

As the number of samples was only 41, in order to analyze the accuracy of each configuration the bootstrap method was used. 1000 new bootstrap samples were created from the original data by sampling with replacement, and then the mean of the absolute error and its confidence intervals (with a confidence level of 95%) were calculated for each configuration.

5.3 Results

In this section, a brief summary of the results obtained with the performance of the MATLAB tests is presented.

5.3.1 Sampling frequency

As it is explained in Section 5.2.2, a parameter that we took under test was the pre-processing sampling frequency.

The webcam we used in our experiments recorded frames at an approximately rate of 20 fps, although regular sampling cannot be assumed. The sampling frequency of the pre-processing module (f_s) is the frequency at which the raw RGB signals are linear interpolated to obtain an evenly sampled signal.

Some examples of the MAE obtained with the different sampling frequencies tested are shown in Figure 5.2. It can be observed that with the lowest f_s too much error is obtained, while it decreases for higher f_s. As it was expected, the minor error is obtained when the sampling frequency is similar to the webcam recording speed, which implies that the system is using as much information as it is available without adding much interpolation noise.

5.3.2 Spectrum estimation

In this section, the influence of the spectrum estimation method in our system is analyzed in detail.

The three PSD estimation methods used in our tests are properly explained in Section 3.4.2, where it can be seen that the quality of the estimation produced by the periodogram and the Welch's method mostly depends on the number of samples of the signal we have, whereas the quality of the estimation produced by the AR models depends on the correct selection of the order of the filter.

As it is shown in Figure 5.3, the periodogram estimator presents its best results when the length of the video is in the range of 25 to 45 seconds. On the other hand, the MAE obtained with the Welch's method presents an inverse relation with the duration of the videos, so the longer the video is, the lower error it has, as it can be seen in Figure 5.4. However, Welch's method seems

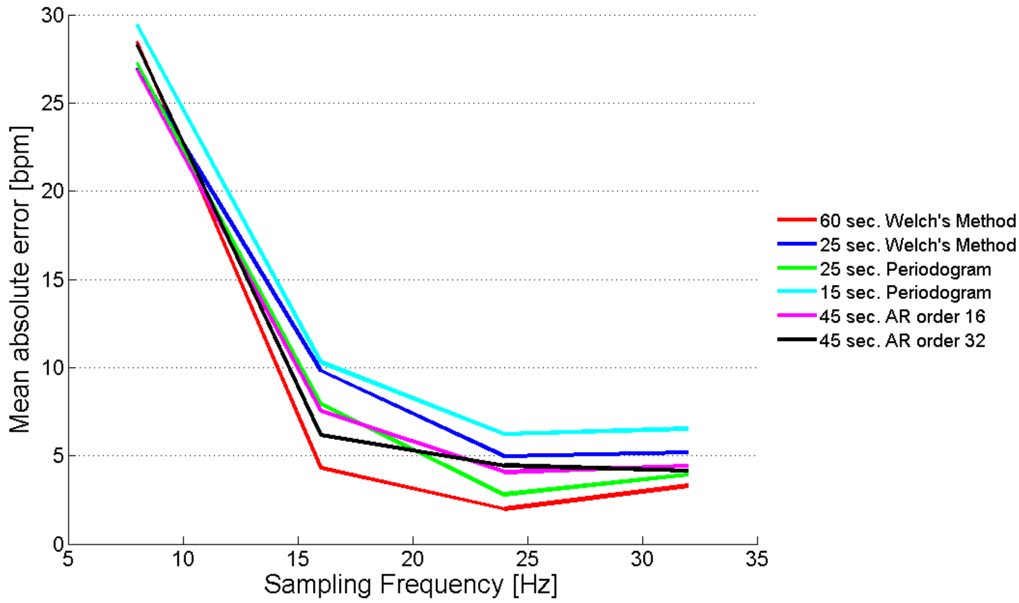

Figure 5.2: Examples of the mean absolute error obtained with different sampling frequencies.

to have acceptable results with 45 seconds of video length.

AR models is a quite robust method in front of the duration of the videos, as it can be observed in Figure 5.5. For its properties, AR models were able to produce good estimations with only few seconds of the video length, as it can be seen in the comparative Figure 5.6, where the three PSD estimation methods are presented. Nevertheless, increasing the length of the video sequence does not improve its performance significantly.

However, the main drawback of the use of AR models involves the estimation of the order of the filter since a wrong order would cause a wrong estimation. Figure 5.7 shows how this parameter affects the AR estimation.

5.4 Practical examples

In this section, some practical examples of the system performance and which are the configurations that show better results are presented.

As it can be seen in Table 5.4, the first five configurations that present lower

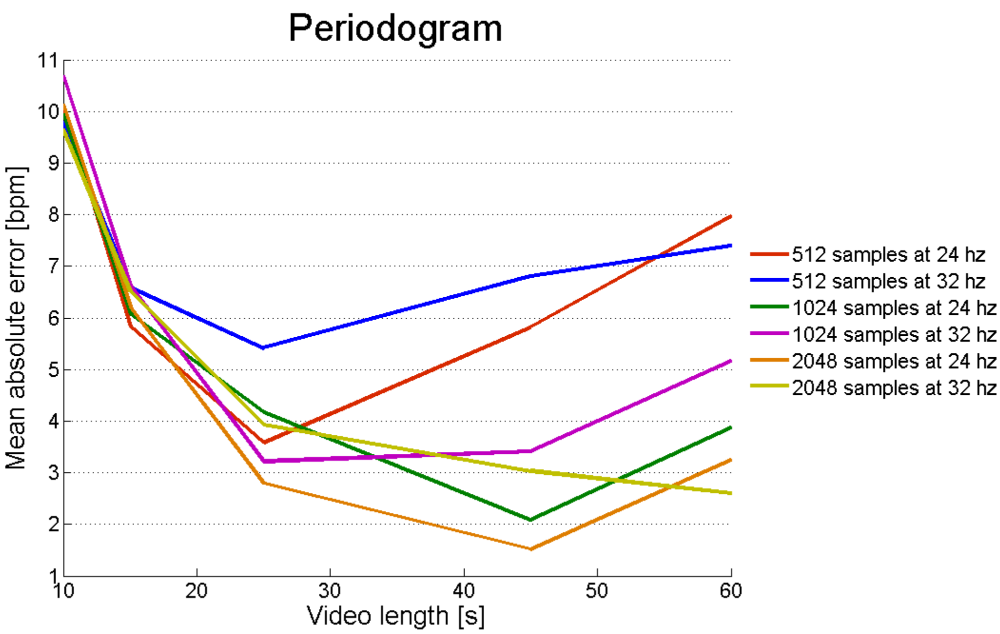

Figure 5.3: Examples of the mean absolute error using the periodogram.

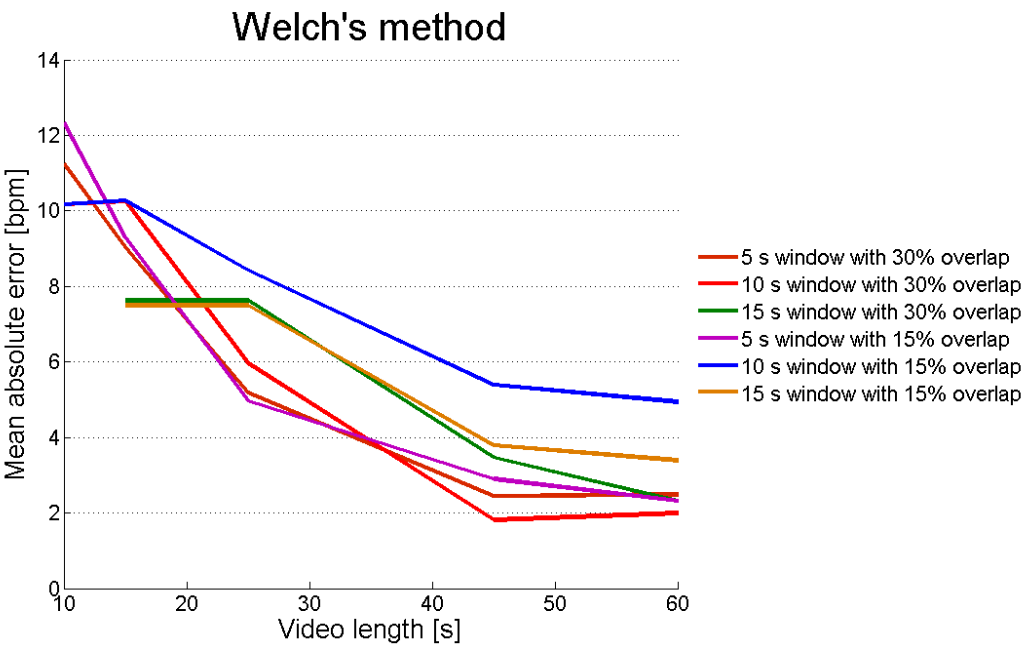

Figure 5.4: Examples of the mean absolute error using Welch's method.

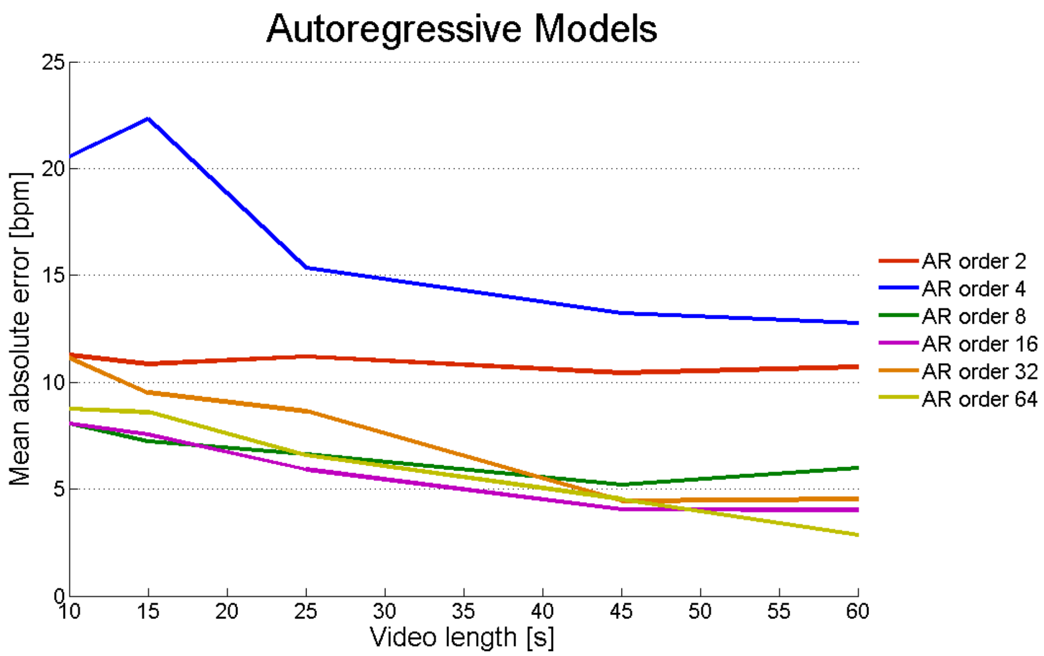

Figure 5.5: Examples of the mean absolute error using AR models.

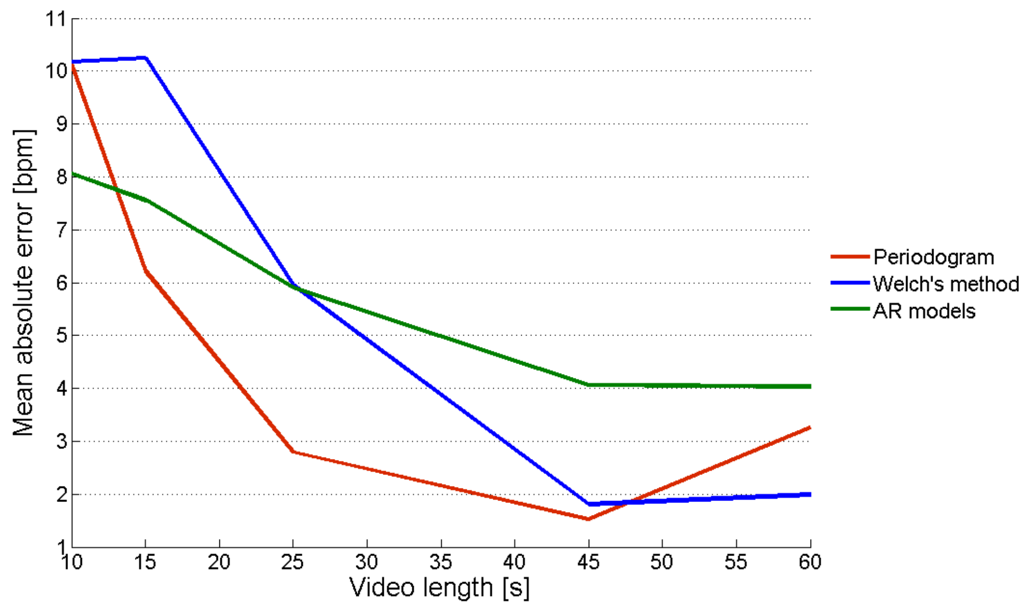

Figure 5.6: Comparison of the mean absolute error that the three PSD estimators present.

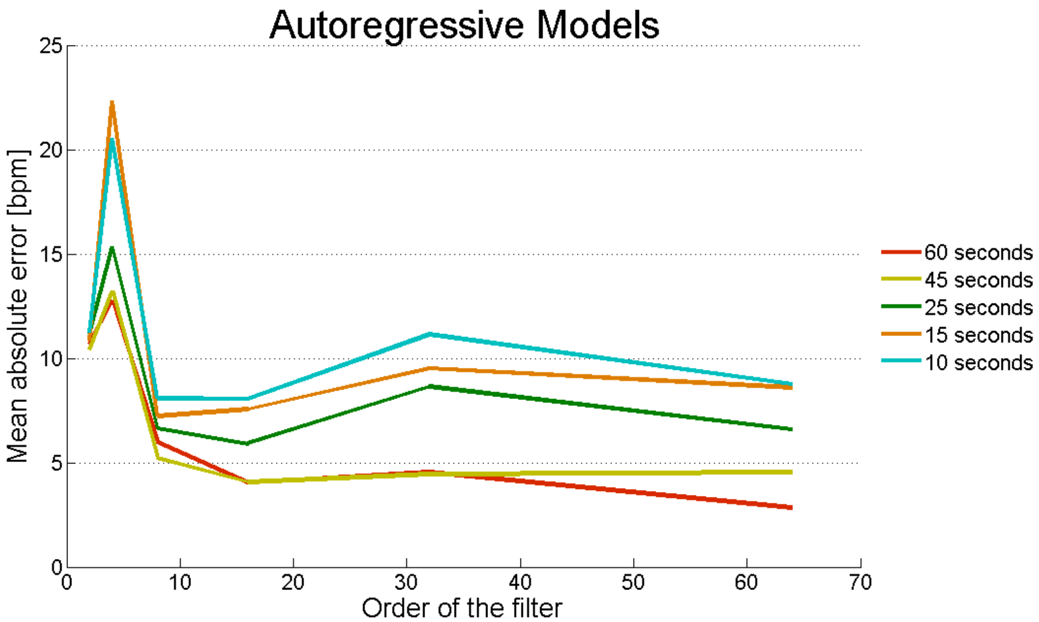

Figure 5.7: Examples of the influence of the order of the filter in AR models.

MAE include some common parameters, like the sampling frequency. Despite the fact that the lowest error is obtained using the periodogram estimator, Welch's method seems to be the most stable method to be used. Because of that, we took the second of the top configurations instead of the first one for the performance of the following examples.

Figures 5.8 and 5.9 show two real examples of how the system works. In the first one, an acceptable recovery of both the signal waveform and the heart rate estimation is done. In the second example, the system cannot properly recover the waveform of the signal, although it is able to find its periodicity and finally does a good heart rate estimation.

	Configurations				
	1st	2nd	3rd	4th	5th
fs [Hz]	24	24	24	24	24
Spectrum estimator	Periodogram	Welch	Welch	Welch	Welch
Length [s]	45	45	45	60	60
Window [s]	-	10	10	10	10
Overlap [%]	-	30	30	75	60
Samples FT	2048	2048	1024	2048	2048
Mean abs. error [bpm]	1,523	1,8094	1,856	1,9345	1,9641
IC min [bpm]	1,4161	1,6833	1,7354	1,7753	1,8015
IC max [bpm]	1,8656	2,2716	2,2683	3,3113	3,2766

Table 5.2: Configurations that present lower error and their statistics.

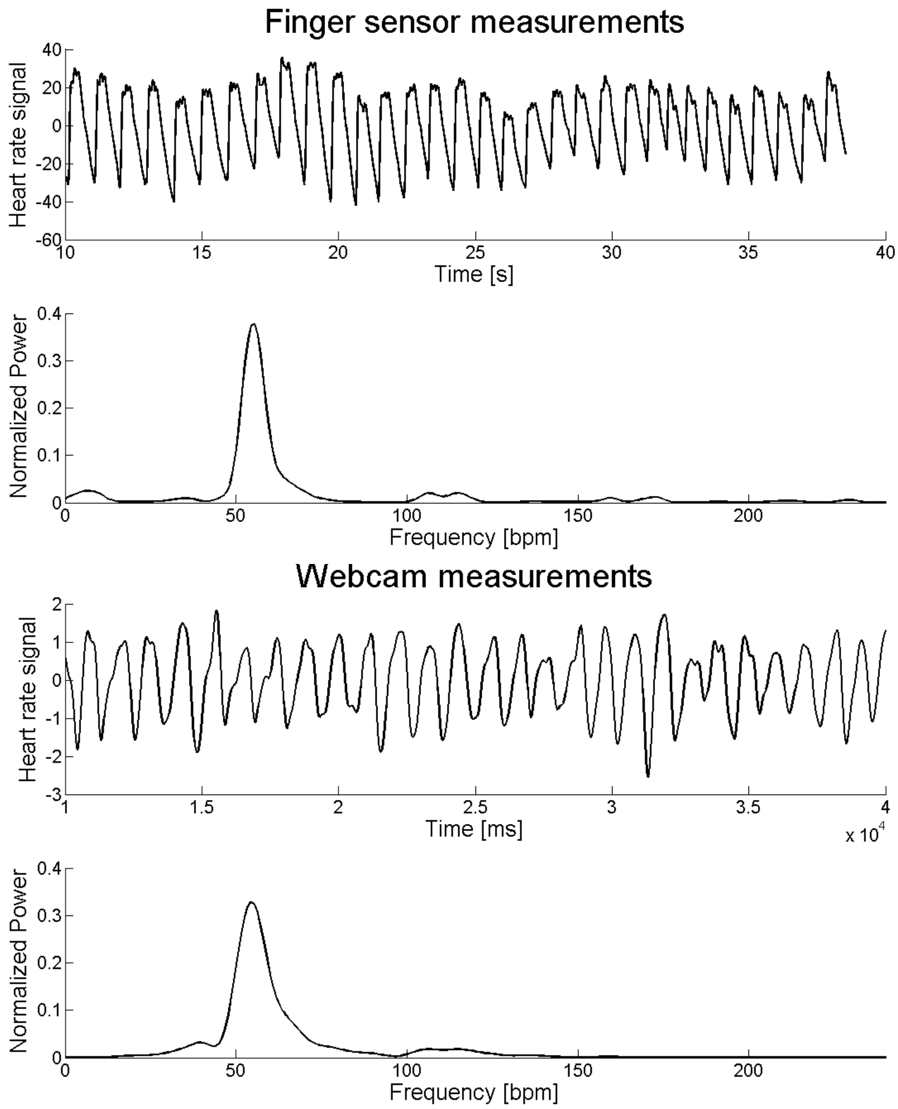

Figure 5.8: Comparison between the signal obtained with the finger sensor device (up) and the signal obtained with the webcam procedure (down).

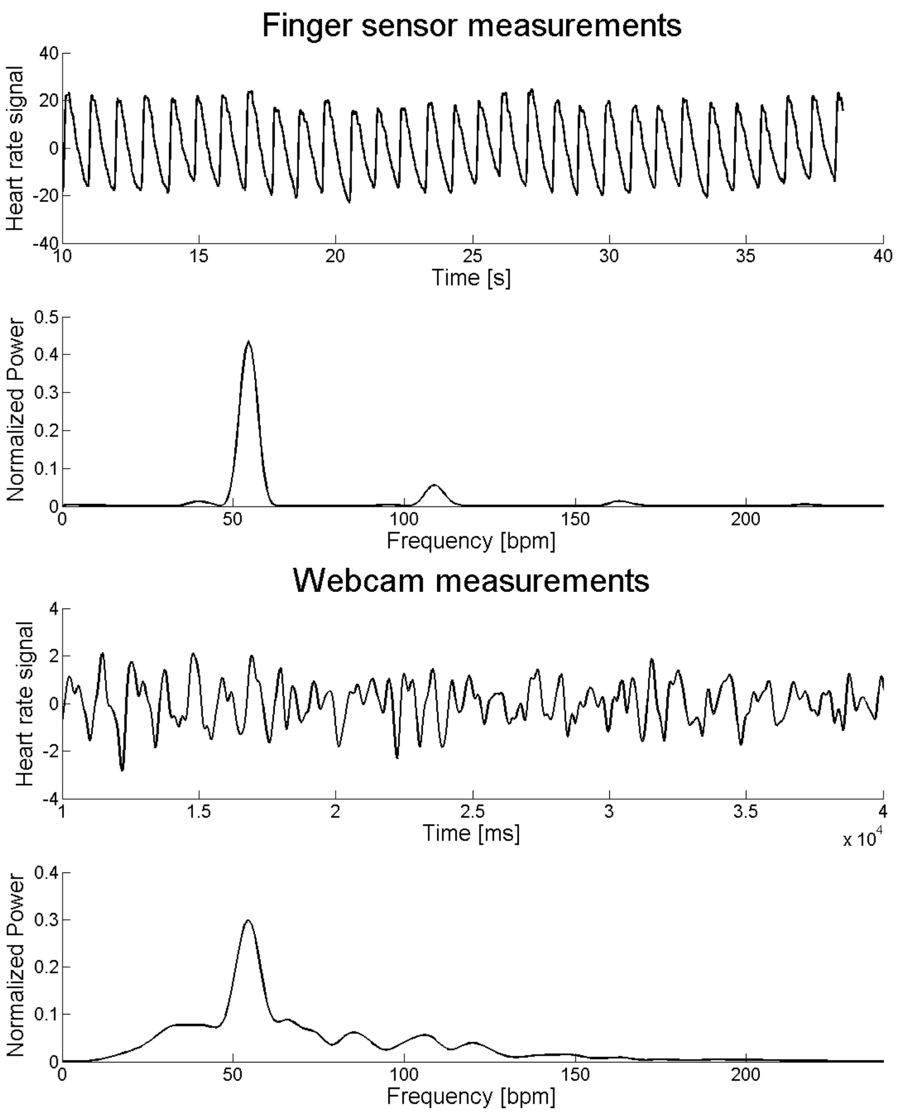

Figure 5.9: Another example of the signal obtained with the webcam procedure. In this case, despite not recovering the signal waveform properly, the system finds its strong periodicity and obtains a good heart rate estimation.

CHAPTER

6

IMPLEMENTATION OF THE HEART RATE ESTIMATION SYSTEM

After seeing the performance of the tests explained in Chapter 5 and the posterior analysis of their results, the final implementation of the real-time heart rate estimator system is detailed in this chapter.

6.1 Introduction

The aim of the implementation of the heart rate estimator system is to develop a final application that can estimate an user's heart rate using video facial information coming from a standard webcam. The final implementation must accomplish two main objectives:

- It must work in real time, therefore the estimation of the heart rate has to be done while the video is being captured.
- The estimation of the heart rate has to be accurate to the real user's heart rate. To do this, results of the tests performed in Chapter 5 may

be used, as well as the image processing techniques described in Chapter 2.

The implementation of the system, which was developed in C++, follows the scheme presented in Figure 6.1, in which three main modules can be observed: the ROI selection system, the obtainment of each channel information and the data processing module.

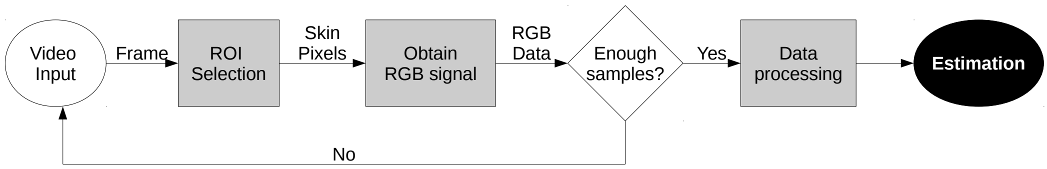

Figure 6.1: Scheme of the real time heart rate estimator system.

The first and the second modules are done frame by frame as they are captured through the camera. However, the data processing module needs to have a minimum number of samples in order to calculate the final estimation, therefore the estimation can only be done if enough samples has already been collected. A proper description of each module can be consulted in the following sections.

6.2 ROI selection

First of all, for each frame of the video input, which is recorded at the average speed of 20 fps, a ROI is selected with the process described in Chapter 2. For this purpose, the Viola-Jones algorithm (which is explained in Section 2.2) included in the 2.4 version of the OpenCV library [44] is used, as well as other computer vision algorithms also included in this library. The tracking and skin pixel selection methods have also been implanted using this library.

6.3 Data extraction

Once the ROI of each frame is selected, the red, the green and the blue channels are spatially averaged and three data measurements, besides the time instant at which the frame was recorded, are obtained. Because of the fact that the data processing phase needs to have a minimum number of samples, the four signals (red, green, blue and time) obtained in this process have to be stored until they can be processed.

Supposing that L samples are needed in order to do the next data processing, and that we want to have a heart rate estimation each D samples (with $D \leq L$), the scheme in which samples are obtained and saved is presented in Figure 6.2. As it can be seen, a window of size L (in the figure in color gray) waits for the incoming samples. The first L samples are stored (in the figure in color black) until the window is full, and then, they are processed. After that, the window is moved D samples away in order to store D new samples. Each time the window is full, a data processing is done.

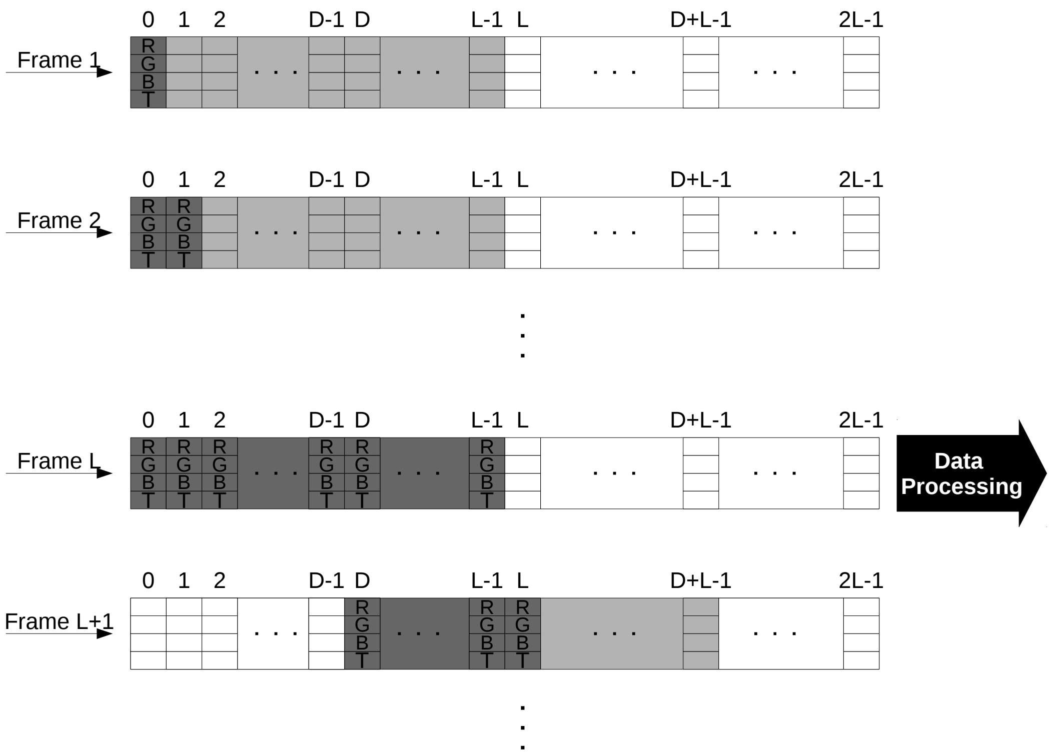

Figure 6.2: Scheme of how samples are stored in order to be used in the data processing module.

6.4 Data processing

Each block of L data samples is processed following the scheme presented in Chapter 3 and applying the results presented in Section 5.3. Despite the fact that the interface of the application allows the user to change the values of some of the parameters tested in Section 5.2.2, the best configuration is the one described in Section 5.4 and it is the one that is used by default, which has the following parameters:

- Sampling frequency: 24 Hz.
- Spectrum estimator: Welch's method with 10 s of window length, 30% overlap and 2048 samples in FT.
- Video length for estimations: 45 s.
- Time between estimations: 5 s.

Some classes and functions of the IT++ library [45] are used in this module in order to implement the signal processing and the mathematical techniques needed, such as fastICA (an algorithm for computing ICA, Section 3.3, very fast) or power spectrum calculation using the Welch's method, which is detailed in Section 3.4.2.

An example of the application interface can be seen in Figure 6.3, where one of the volunteers is actually testing the system by monitoring his heart rate with a chest strap and a watch. In the figure can be observed how the region of the image where the face is located is selected (white rectangle) and it also can be seen the power spectrum and the amplitude of the latest estimated signal, as well as the latest heart rate estimation.

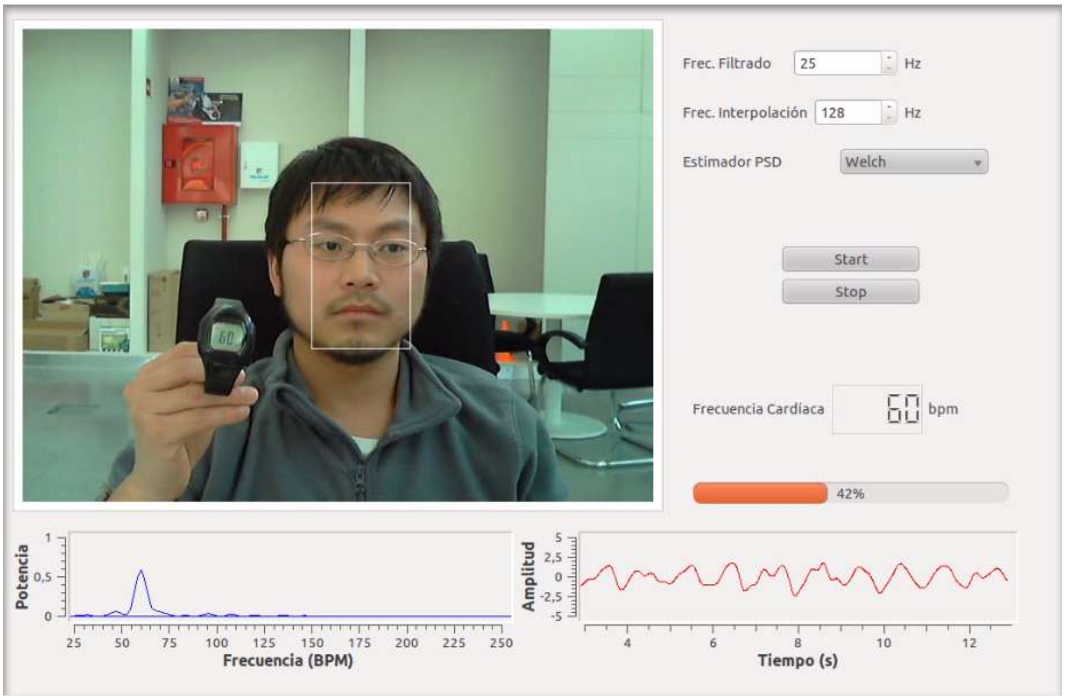

Figure 6.3: Interface of the final application being used by one of the volunteers.

CHAPTER

7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

This section summarizes the conclusions that can be extracted from the development of this project. Achievements in both the image processing and the data processing phase are described, as well as the general conclusions about the whole project.

Improvements in the motion robustness of the system have been developed respect to the existent methodologies proposed in [4, 5] by implementing the tracking and the skin segmentation system. Moreover, because of the exhaustive tests performed in this project, some improvements in the data processing phase have been also acquired. It has been proved that the system works not only with natural ambient light but also with fluorescent light from inside an office.

A complete study of the most reliable parameters for the data processing phase has been developed. This study has revealed important results about the implementation of the system, such as the best sampling frequency to be used or the most robust PSD estimator. It has been proved that the best f_s is that one that is closer to the recording speed of the webcam, in our case 24 Hz, whereas the most robust PSD estimator that was taken under test is the Welch's method estimator, which presents a mean absolute error between 1,7 bpms and 2,3 bpms.

In the other hand, it has been seen that AR methods can estimate the heart rate from a short video sequence, although this kind of PSD estimator is very affected by the correct selection of the order of the filter. Periodogram, in spite of being the simplest PSD estimation method, has shown very irregular results. It turned out to be very affected by the number of samples available, resulting in bad estimations both in cases where the number of samples available were too low or too high.

A final and functional application in which an user can estimate its heart rate in an unobtrusive manner by using its own computer has been implemented. This project has proved that a real - time estimation of the heart rate can be done using only facial video information, and the study of the reliability of the system showed that the mean absolute error obtained is between **1,7 bpms** and **2,3 bpms**.

During the realization of this project, an exhaustive study of the involved algorithms and techniques has been carried out. In particular, independent component analysis, which is the base of the heart rate recovery procedure, as well as power spectral density estimation and computer vision techniques. A deep background in computer vision algorithms (as face detection, tracking systems and skin segmentation) as well as knowledge in image processing tools (OpenCV) have been acquired.

Finally, with the development of this project, skills in project management have been acquired as well as experience in working in a real environment.

7.2 Future work

The main lines of work derived from this project are oriented to improve the performance of the heart rate estimator system, in order to achieve better results and more accurate estimations.

A first approximation to improve the system proposed in this project would be to implement a better approach to solve the motion robustness problem.

The scheme proposed in this project, despite being more robust in front of motion than the existent techniques, cannot properly recover the heart rate if the user does not remain relatively still. With motion, the extracted RGB traces are affected by a greater number of signals than the available mixture signals, and ICA is unable to separate the underlying independent components. This could be solved by improving the ROI selection system or by using some non-Blind Source Separation techniques (such as the Bayesion approach [46]) that take into account some a-priori information about the heart signal.

Another interesting improvement that can help to achieve better results in the heart rate estimation system would be to implement a more complex and efficient skin segmentation algorithm, as the ones compared in [29]. It would help to select more efficiently those pixels that comes from a skin face region.

Providing information about the lightness variation of the video background would help to detect those light fluctuations that have nothing to do with the heart signal. By using this technique, some noise coming from other illumination sources can be reduced. Moreover, different color space for extracting the light information can be used. A first approximation in this direction is done in [47].

As the heart rate varies with the age and the gender of the user [48], an interesting future work to measure the effectiveness of the system would be to carry on an exhaustive study of the working of the system along different age stages.

As the proposed system can be used for monitoring the heart rate of a patient from far away from a hospital, it may be interesting to add a facial recognition module, which would help to keep a database in which each user information would be automatically saved.

Another interesting future line of investigation would be to find methodologies to measure more complex physiological parameters such as heart rate variability, blood oxygen level or blood pressure.

Information proceeding from the heart signal is usually very useful for knowing some aspects of a person, like feelings or physical condition. Another application in which this project can be very useful is in emotions recognition, as physiological parameters are often used in that kind of applications [49]. These applications can be used in fields like the automotive industry, in order to measure the stress level of the driver [50].

Moreover, this technology can be adapted to be used in heart rate detection applications, where knowing if a user has or not pulse is more important than the estimate HR. Although some modifications of the methodology proposed in this project would be necessary, this kind of application can be used in fields like liveness detection in facial recognition systems, where it can be used to detect if an impostor is trying to access the system.

Finally, improvements in the recovery of the heart signal must be done. As it was seen, although good heart rate estimation can be reached in most cases, the heart signal sometimes cannot be properly recovered. If this issue could be solved, the methodology proposed in this project could be used in more complex fields like data encryption, following the developments proposed in [51].

BIBLIOGRAPHY

[1] C. Takano and Y. Ohta. Heart rate measurement based on a time-lapse image. Medical Engineering and Physics, vol. 29, pp. 853-857, 2007.

[2] W. Vekruysse, L.O. Svaas and J.S. Nelson. Remote plethysmographic imaging using ambient light. Optics Express, 2008.

[3] J. Allen. Photoplethysmography and its application in clinical physiological measurement. Physiological Measurement, vol. 28, pp. R1-R39, 2007.

[4] M.Z. Poh, D.J. McDuff and R. Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. Optics Express, vol. 18, pp. 10762-10774, 2010.

[5] M.Z. Poh, D.J. McDuff and R. Picard. Advancements in noncontact, multiparameter physiological measurements using a webcam. IEEE Transactions on Biomedical Engineering, vol. 58, no. 1, pp. 7-11, 2011.

[6] EFE. La telemedicina ayudará a ahorrar costes sanitarios en la UE (online). [Consulted: June 2012] Available: <http://www.lavozdegalicia.es/sociedad/2011/05/11/0003130512661416676968 8.htm>.

[7] R. Lizcano. La telemedicina se consolida en Galicia para ganar agilidad (online). 2011. [Consulted: June 2012] Available: <http://www.elcorreogallego.es/galicia/ecg/telemedicina-consolida-galicia-ganar-agilidad/idEdicion-2011-03-07/idNoticia-646659/>.

[8] Philips. Philips Vital Signs Camera (online). 2011. [Consulted: November 2011] Available: <http://www.vitalsignscamera.com/>.

[9] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Proceedings of the Second European Conference on Computational Learning Theory, pp. 23-37, 1995.

[10] P. Viola and M. Jones. Robust Real Time Detection, Second International Workshop on Statistical and Computational Theories of Vision-Modeling. Learning, Computing, and Sampling, 2001.

[11] C. Papageorgiou, M. Oren and T. Poggio. A general framework for object detection. International Conference on Computer Vision, 1998.

[12] F. Crow. Summed-area tables for texture mapping. In Proceedings of SIGGRAPH, vol. 18, pp 207-212, 1984.

[13] R.E. Kalman. A New Approach to Linear Filtering and Prediction Problems. Transaction of the ASME-Journal of Basic Engineering, pp. 35-45, 1960.

[14] X. Xie, R. Sudhakar, and H. Zhuang. Real-time eye feature tracking from a video image sequence using kalman filter. IEEE Transaction on Systems, Man and Cybernetics, vol. 25, no. 12, pp. 1568-1577, 1995.

[15] F. Xu, X. Liu and K. Fujimura. Pedestrian detection and tracking with night vision. IEEE Transactions on Intelligent Transportation Systems, vol. 6, no. 1, pp. 63-71, 2005.

[16] E. Cuevas, D. Zaldivar and R. Rojas. Kalman filter for vision tracking. Technical Report B 05-12, Freier Universitat Berlin, Institut fur Informatik, 2005.

[17] Y. Cheng. Mean Shift, Mode Seeking, and Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, pp. 790-799, 1995.

[18] D. Comaniciu, R. Ramesh and P. Meer. Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, 564-577, 2003.

[19] V. Karavasilis, C. Nikou, and A. Likas. Visual Tracking by Adaptive Kalman Filtering and Mean Shift. Artificial Intelligence: Theories, Models and Applications, pp. 153-162, 2010.

[20] G. Welch and G. Bishop. An introduction to the Kalman Filter. University of North Carolina, Department of Computer Science, TR 95-041, 1995.

[21] T. Kailath. The Divergence and Bhattacharyya Distance Measures in Signal Selection. IEEE Trans. Commun. Technol., vol. COM-15, no. 1, pp. 52-60, 1967.

[22] D. Chai and K.N. Ngan. Face Segmentation Using Skin Color Map in Videophone Applications. IEEE Trans. Circuits and Systems for Video Technology, vol. 9, no. 4, pp. 551-564, 1999.

[23] C. Garcia and G. Tziritas. Face Detection Using Quantized Skin Color Regions Merging and Wavelet Packet Analysis. IEEE Trans. Multimedia, vol. 1, no. 3, pp. 264-277, 1999.

[24] M.J. Jones and J.M. Rehg. Statistical Color Models with Application to Skin Detection. Int'l J. Computer Vision, vol. 46, no. 1, pp. 81-96, 2002.

[25] H. Wang and S.F. Chang. A Highly Efficient System for Automatic Face Detection in Mpeg Video. IEE Trans. Circuits and Systems for Video Technology, vol. 7, no. 4, pp. 615-628, 1997.

[26] J. Yang and A. Waibel. A Real-Time Face Tracker. Proc. IEEE Workshop Applications of Computer Vision, pp. 142-147, 1996.

[27] H. Greenspan, J. Goldberger and I. Eshet. Mixture Model for Face Color Modeling and Segmentation. Pattern Recognition Letters, vol. 22, p. 1525-1536, 2001.

[28] S.L. Phung, D. Chai and A. Bouzerdoum. A Universal and Robust Human Skin Color Model Using Neural Networs. Proc. INNS-IEEE Int'l Joint Conf. Neural Networks, vol. 4, pp. 2844-2849, 2001.

[29] S.L. Phung, A. Bouzerdoum and D. Chai. Skin segmentation using color pixel classification: analysis and comparison. IEEE Transactions on pattern analysis and machine intelligence, vol. 27, pp. 148-154, 2005.

[30] J. Yang, L. Weier, and A. Waibel. Skin-color modeling and adaptation. Proc. Asian Conf. on Computer Vision, pp. 687-694, 1998.

[31] J.C. Terrillon and S. Akamatsu. Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images. Proc. Vision Interface, pp. 180-187, 1999.

[32] L. Sigal, S. Sclaroff and V. Athitsos. Skin color-based video segmentation under time-varying illumination. IEEE Trans. Pattern Anal. Mach. Intell, vol. 26, pp. 863-877, 2004.

[33] P. Comon. Independent Component Analysis - A new concept?. Signal Processing, vol. 36, pp. 287-314, 1994.

[34] J.F. Cardoso. Blind signal separation: statistical principles. Proceedings of the IEEE, vol. 86, num. 10, pp. 2009-2025, 1998.

[35] B. Sierra. Aprendizaje Automático: conceptos básicos y avanzados. Prentice Hall, 2006.

[36] I.T. Jolliffe. Principal Component Analysis. Springer, 2002.

[37] J. Jacod. A.N. Shiryaev. Limit Theorems for Stochastic Processes. Springer, 1987.

[38] P.D. Welch. The use of fast Fourier transform for the estimation of power spectra: A method based on time-averaging over short, modified periodograms. IEEE Trans. Audio Electroacoustics, vol. 15, pp. 70-73, 1967.

[39] S.L. Marple. Digital Spectral Analysis with Applications. Prentice-Hall, Englewood Cliffs, 1987.

[40] B. Efron and R. Tibshirani. Bootstrap Methods For Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy. Statistical Science, vol. 1, pp. 54-77, 1986.

[41] R.W. Johnson. An Introduction to the Bootstrap. Teaching Statistics, vol. 23, pp. 49-54, 2001.

[42] R. Wehrens, H. Putter and L.M.C. Buydens. The bootstrap: a tutorial. Chemometrics and intelligent laboratory systems, num. 54, pp. 35-52, 2000.

[43] Digital Pulse Wave Finger-clip Sensor (online). 2010. [Consuted: November 2011]. Available: <http://www.heartsensor.info/uf06/uf06.html>.

[44] OpenCV (online). 2012. [Consulted: June 2012]. Available <http://opencv.org/>.

[45] IT++ Documentation (online). 2010. [Consulted: March 2012]. Available <http://itpp.sourceforge.net/current/>.

[46] K.H Knuth. A Bayesian approach to source separation. Proceedings in the ICA'99, Aussios (France), pp. 283-288, 1999.

[47] P. Sahindrakar. Improving motion robustness of contact-less monitoring of heart rate using video analysis. Technische Universiteit Eindhoven, Department of Mathematics and Computer Science, 2011.

[48] K. Umetani, D. H. Singer, R. McCraty and M. Atkinson. Twenty-four hour time domain heart rate variability and heart rate: relations to age and gender over nine decades. Journal of the American College of Cardiology, vol. 31, pp. 593-601, 1998.

[49] K.H. Kim, S.W. Bang and S.R. Kim. Emotion recognition system using short-term monitoring of physiological signals. Medical and Biological Engineering Computing, vol. 42, pp. 419-427, 2004.

[50] L. Yu, X. Sun and K. Zhang. Driving Distraction Analysis by ECG Signals: An Entropy Analysis. Internationalization, design and global development, vol. 6775, pp. 258-264, 2011.

[51] C.K. Chen, C.L. Lin, C.T. Chiang and S.L. Lin. Personalized information encryption using ECG signals with chaotic functions. Information Sciences, vol. 193, pp. 125-140, 2012.