Analysis and Visualization of the Two-Dimensional Blood Flow Velocity Field from Videos

by

Jun Yang

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

We estimate the velocity field of the blood flow in a human face from videos. Our approach first performs spatial preprocessing to improve the signal-to-noise ratio (SNR) and the computational efficiency. The discrete Fourier transform (DFT) and a temporal band-pass filter are then applied to extract the frequency corresponding to the subjects heart rate. We propose multiple kernel based k-NN classification for removing the noise positions from the resulting phase and amplitude maps. The 2D blood flow field is then estimated from the relative phase shift between the pixels. We evaluate our approach about segmentation as well as velocity field on real and synthetic face videos. Our method produces the recall and precision as well as a velocity field with an angular error and magnitude error on the average.

Index Terms: Blood Flow, Velocity Field, PCA, DFT, Feature Extraction, KNN, Multiple Kernel Learning Classification, Denoising, Synthetic Video

Acknowledgements

I would like to express my sincerest appreciation and thanks to my supervisor, Professor Abdulmotaleb El Saddik, whose guidance, support and patience to my graduate experience. I would like to thank you for encouraging my research, giving me useful remarks and offering me help in the research of this master thesis.

Sincere thanks go to Dr. Benjamin Guthier for the invaluable assistance and guidance provided throughout my whole research. You taught me a lot about my research. At the same time, you are also a very kind, easygoing person and good friend. It is a memorable experience to work with you.

Also, I would like to thank all the colleges in Multimedia Computing Research Laboratories. Special thanks to Dr. Shiai Zhu and Dr. Hussein Al Osman. You provided me advice and assistance during my studies and research in University of Ottawa.

I would like to thank my family, who have supported me throughout the entire process to strive towards my goal in Canada. This work is dedicated to them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Cardiovascular disease, also called heart disease, is the leading cause of deaths worldwide. This class of diseases involves the heart and blood vessels, also called as heart disease. The statistics from the World Health Organization (WHO) suggest that in 2012, around 17.5 million people died from cardiovascular disease, accounting for 31% of all global deaths. Of these deaths, an estimated 7.4 million were due to coronary heart disease, and 6.7 million were due to stroke. Cardiovascular disease affects the heart and blood vessels and causes serious circulatory system pathologies, such as heart attack and stroke[20]. Although lifestyle change, social change, medicine, and surgery can be used to treat the disease, survivors will still suffer from it.

Blood flow velocity is a valuable piece of information in the diagnosis and treatment of cardiovascular disease[28]. Figure 1.1 illustrates the formed elements (blood cells) in the blood vessels. As one of the most important elements in the cardiovascular system, blood flow velocity indicates the level of brain activity. For example, stroke is a result of sudden interruption of the blood flow to the brain, which occurs when part of the brain is deprived of the oxygen and blood it needs, such as when a piece of blood clot breaks off and then gets stuck in the artery, which leads to the death of brain tissues. The more blood will flow through the specific region in such a period implies that more blood supply was needed

Figure 1.1: Blood flow information used for cardiovascular disease diagnosis.[4]

by the brain (i.e., more oxygen needed), thus indicating a higher level of brain activity. Conversely, when a stroke occurs, the patient's blood vessels will be blocked, which causes abnormal blood flow velocities.

The current predominant techniques for monitoring blood flow still require patients to wear inconvenient equipment that causes discomfort and pain[28]. However, there is growing interest in non-contact blood flow measurement, particularly for those whose skin is fragile and allergic to equipment. Currently, estimating blood flow by carotid ultrasound is the dominant method, which is also painless and harmless. Several studies have proposed the detection of blood flow using a Doppler velocimeter and ultrasound[36][17][33][56].

Kasai et al.[36] developed an algorithm to estimate the mean velocity in a large spatial domain by exploiting an autocorrelation technique. Another method, proposed by Jensen et al.[33], is able to visualize the distribution of blood flow velocity in the region of interest. An algorithm to estimate rates of blood flow in microvascular networks has been introduced in[21].

1.2 Problem statement

In our work, we exploit the subtle color variation of the human skin due to blood circulation to estimate the velocity field of the blood flow. Our work is inspired by the Eulerian video magnification for revealing subtle changes, including subtle color changes on the skin and imperceptible motions in the world[55]. The idea of Eulerian video magnification is that for the input video, we consider each pixel as the time series and amplify the variation in a given temporal frequency band of interest. X He et al.[27] measured the propagation time delay between neck and wrist using this technique. The most important difference in our technique is that instead of exploiting this method to amplify and make subtle color variations visible, we want to extract quantitative information about heartbeats.

Our method is also an extension of previous works that measure the heart rate from a video[43][53]. These methods utilize pixel values in the facial region and temporally band-pass filter the signals with a proper band. Verkruysse et al.[53] measured the cardiovascular pulse wave signal on the human face from video and extracted the heart rate from variations in the green channel. Poh et al.[43] performed Independent Component Analysis (ICA) to extract a single pulse wave from an RGB signal, which is spatially averaged over the face region, and analyzed features in the frequency domain to detect the heart rate. We go one step further and aim to extract not only the heart rate, but also the velocity of the blood flow at any point on the face.

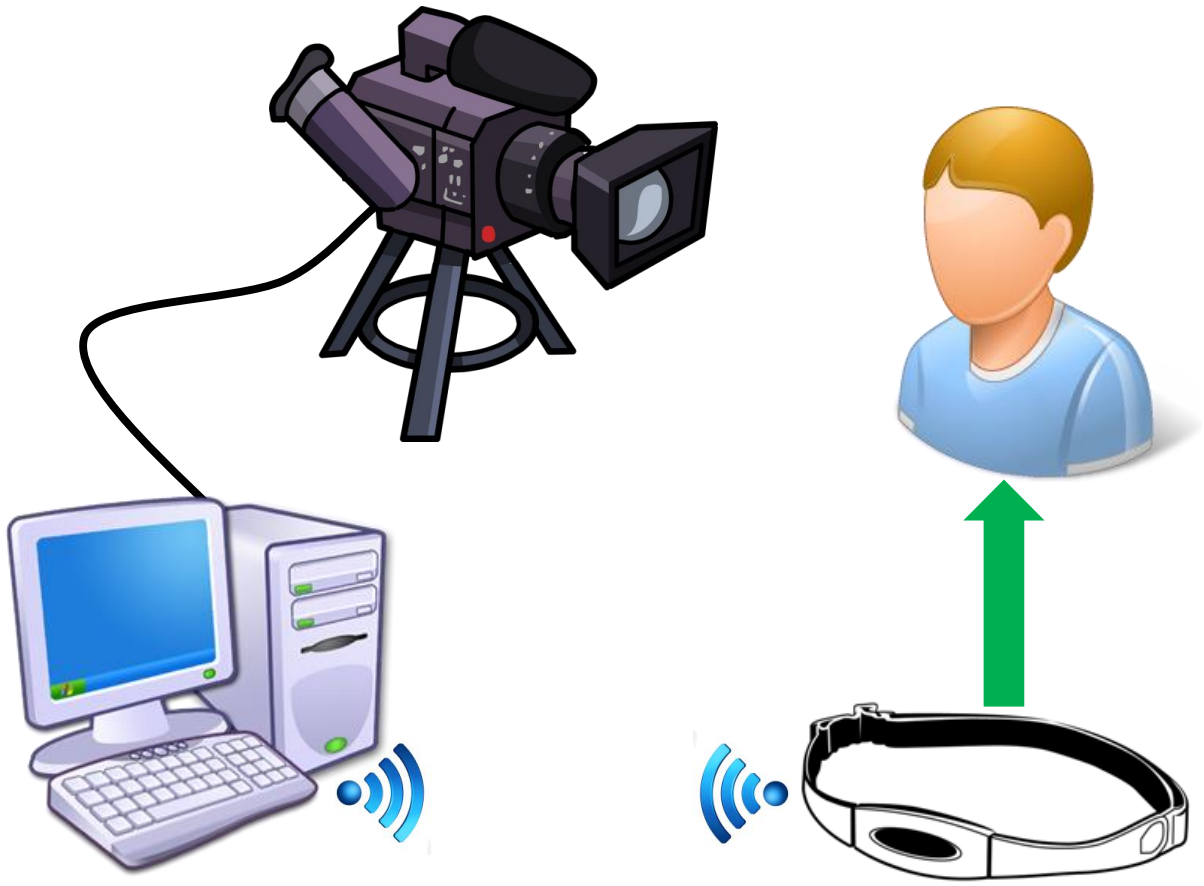

Figure 1.2: A framework for estimation blood flow velocities from videos.

1.2.1 Heart rate detection

The cardiac output refers to the volume of blood being pumped by the heart. To estimate the blood flow velocity field, we need to take the heart rate as the input. In our previous work, the heart rate was determined by a heart rate sensor on the subject's chest. However, it can also be determined automatically by finding a peak in the frequency spectrum as shown in[43] and [53].

In particular, although not the focus of this thesis, we show how we extract heart rate from standard RGB face videos via analysis of the subtle variations of color on the skin caused by blood flow. Following [43] and [53], we are able to detect the heart rate by finding out the maximum amplitude frequency spectrum over the facial region.

1.2.2 Noise removal and classification

As a consequence of the subject's involuntary movements or camera jitter, artifacts can be introduced during video capture. For example, slight movement in and out periodicity equal to the heart rate frequency within a region of interest (ROI) would appear as a fake heart rate (HR) signal, which can be either stronger or weaker than the real HR signal. To leave this artifact out, motion artifacts are assumed to be equally strong in all frequency bands around HR frequency, and there is no such relation of phase shift for adjacent pixels.

The amplitude and phase map of heart rate frequency are extracted and visualized as the initial features. Further features can then be then extracted from them. We exploit two denoising techniques to remove such noise from the involuntary motion on the head and background, respectively. The first technique removes noise by using two thresholds. However, the second technique divides all the spatial positions as blood flow, notion artifacts or non-variation noise positions. The multi-kernel based k-NN classification is applied for segmenting blood flow positions.

1.2.3 Estimation and visualization

The remaining pixels are locations in the face that exhibit blood flow. The phase shift between these pixels gives us information about when this pixel changes its color due to the blood flowing through it relative to its neighbors. The difference between two neighboring phase values (i.e., the phase shift) thus directly corresponds to the velocity of the blood flow at this point.

We compute this difference in horizontal and vertical directions by applying gradient, which corresponds to the desired 2D velocity field of the blood flow.

1.3 Contributions

The goal of this work is measuring blood flow velocities without contact. Specifically, we estimated the velocity field of the blood flow in a human face from videos. We made the

following major contributions:

- Taking a normal video as input, for each point of the face region, we visualize the phase shift (blood flow direction) and the magnitude (Intensity of skin color variation) of the blood flow on 2D maps.
- Noise in this visualization was reduced by employing two denoising techniques and then comparing difference between them.
- The blood flow velocity was computed and visualized by calculating the gradient on the phase map after reducing the noise.
- A database of still face videos was built for qualitative evaluations of our method. The accuracy of the estimated blood flow was evaluated using specifically designed synthetic face videos.

1.4 Scholarly Achievements:

In the process of completing this work, the following publications have been submitted or accepted:

- **Jun Yang**, Benjamin Guthier and Abdulmotaleb El Saddik. Estimating two-dimensional blood flow velocities from videos, International Conference on Image Processing (ICIP), Quebec City, Canada, September 2015. (accepted)
- Shiai Zhu, Samah Aloufi, **Jun Yang** and Abdulmotaleb El Saddik. On the Learning of Image Social Relevance from Heterogeneous Social Network, Elsevier Journal on Neurocomputing, 2015. (submitted)

1.5 Thesis overview

The thesis is organized as follows:

- Chapter 2 introduces the background and related works of this research. The inspiration, Eulerian video magnification[55], is introduced first. Principal component analysis (PCA) for enhancing blood flow information through the skin by exploiting multichannel color space, are stated in this chapter. As the highly related research topic to estimating blood flow, this chapter also presents the heart rate measurement from videos[53][43].

- By giving mathematical solutions, Chapter 3 introduces our approach to visualize the blood flow and estimate a velocity field from videos. Then spatial processing, temporal analysis, and filtering to image sequences are described in this chapter, following by PCA, applied to use all the information from RGB channels. To compute and visualize the blood flow pattern, Chapter 3 also illustrates the blood flow selection by exploiting supervised clustering. Lastly, the final estimation and visualization are presented.

- In Chapter 4, a qualitative and quantitative evaluation of performance of our algorithm is presented. The results from the real face videos allow a qualitative evaluation of the computed velocity fields. To quantitatively evaluate the accuracy of the blood flow detected by our method, this chapter also describes synthesizing face videos with known velocity fields. By using different metrics, our evaluation on synthetic videos shows that our approach is able to segment blood flow positions and estimate the blood flow velocities in noisy synthetic videos.

- Chapter 5 concludes the thesis and highlights the contributions of this research. Future works are also discussed.

Chapter 2

Related Work and Background

In this chapter, the inspiration, Eulerian video magnification[55], is introduced to visualize the skin color variation due to blood flow. As the previous step of our work, heart rate detection is also introduced in this chapter. We will also present he principal component analysis (PCA) for dimensionality reduction. Lastly, the k nearest neighbors algorithm (k-NN) is described at the end of the chapter.

2.1 Eulerian video magnification

Our world is constantly changing over time. Many scenes in our daily lives appear to be static but in fact contain informative variations and change that are too subtle to be visible to us. However, such hidden signals might be valuable and precious for many applications. For example, skin color varies slightly between red and yellow. Similarly, motion with low spatial amplitude is also hidden and invisible in normal videos that are captured from cameras or smartphones.

The purpose of Eulerian video magnification[55] is to reveal the hidden information that falls below the sensitivity of the human visual system. The basic approach is that given the temporal frequency band of interest, we consider the time series of pixel values as 1D signal and amplify the strength of variation for each spatial location. The basic framework of Eulerian video magnification is illustrated in Figure 2.1.

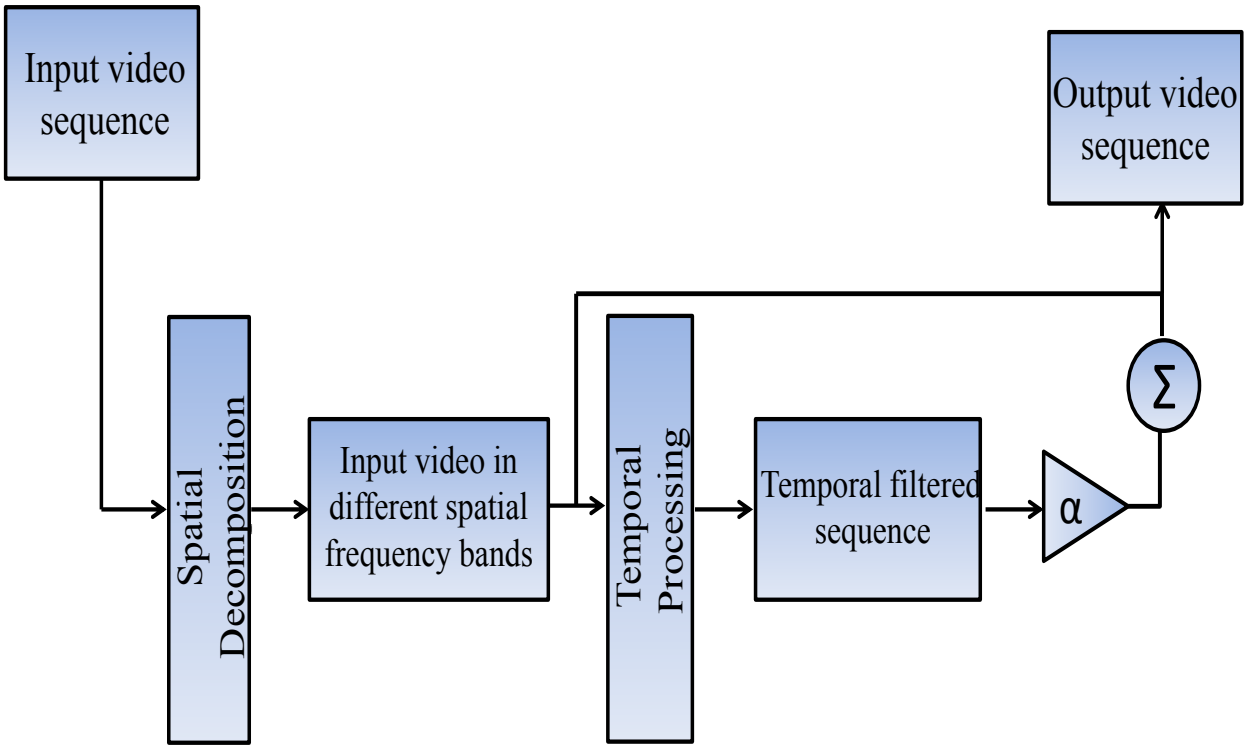

Figure 2.1: Overview of the Eulerian video magnification.

The Eulerian video magnification system processes video as follows:

- The video sequence is taken as input and decomposed into different spatial frequency bands, which is also a full Laplacian pyramid[14] in the general case.
- A band-pass filter with a user-specified frequency band is then defined and applied on each spatial band.
- Given amplification factors α, the filtered spatial bands in the preceding step are amplified by this factor and added back to the original signal.
- The spatial pyramid is collapsed to obtain the final video output.

With such a system, we are able to make some invisible variations visible. For example, in Figure 2.3, by amplifying the band of heart rate frequency, the color variation, caused by blood flow through the face, is revealed.

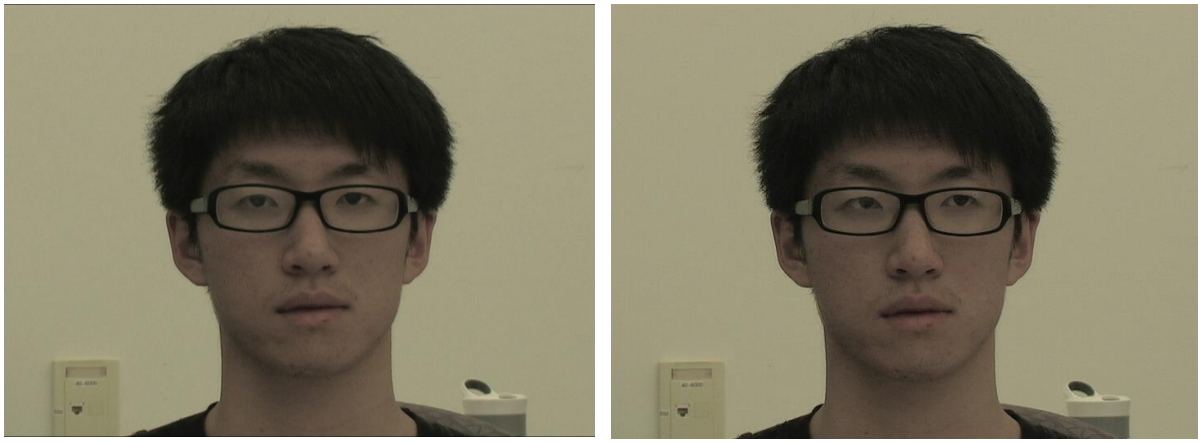

Figure 2.2: Two original frames from one video sequence.

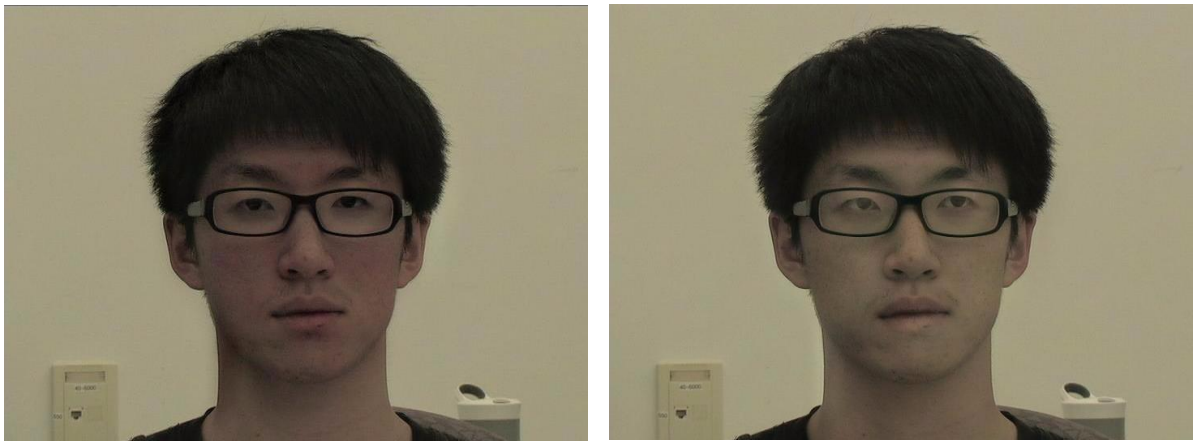

Figure 2.3: Two frames from amplified video sequence in figure 2.2 showing the color variation on the skin due to blood circulation.

The Eulerian video magnification can both amplify the color variation and the low spatial amplitude motion. Unlike Lagrangian motion magnification[38] by tracking motion, Eulerian video magnification approximates the motion estimation, which is similar to the optical flow technique[30][39]. A simple case of a 1D signal undergoing translational motion is illustrated in Figure 2.4.

Let $V(x,t)$ denote the signal value (image intensity in 2D case), where x is the spatial position, and t is the time. The observed signal value can be expressed as:

$$V(x,t) = g(x + f(t)) \tag{2.1}$$

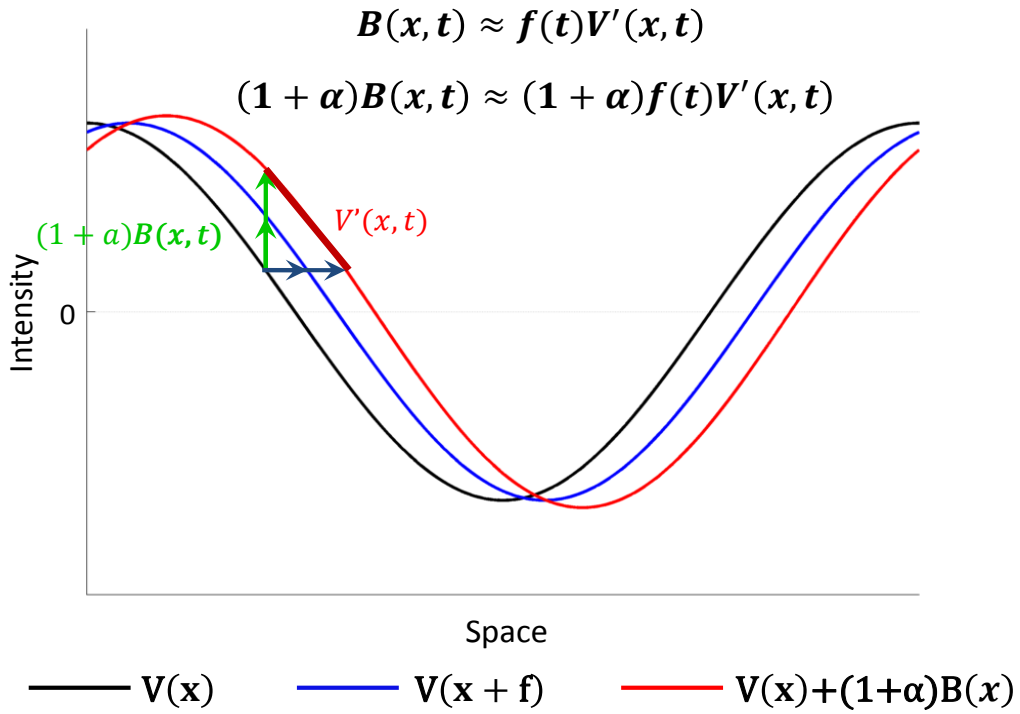

Figure 2.4: Approximation of spatial translation by temporal filtering.

where $f(t)$ is a displacement function, corresponding to the time t. The ideal motion magnification is the synthesizing

$$\hat{V}(x,t) = g(x + (1+\alpha)f(t)) \tag{2.2}$$

where α is the user-defined amplification factor.

Using first-order Taylor series expansion, the 1D signal $g(x+f(t))$ can be expressed by

$$V(x,t) \approx g(x) + f(t)\frac{\partial g(x)}{\partial x} \tag{2.3}$$

For the sake of simplicity, $f(t)\frac{\partial g(x)}{\partial x}$ is substituted for $B(x,t)$, which is also the temporal band-pass filtered signal from $V(x,t)$. In the Eulerian video magnification process, the band-passed signal $B(x,t)$ is then amplified by a given factor α and added back to the

input signal $V(x,t)$, resulting in the following final output:

$$\tilde{V}(x,t) \approx g(x) + \alpha B(x,t) \tag{2.4}$$

The first-order Taylor series expansion performs very well on smooth spatial variation with a relatively small amplification factor α. However, this approximation becomes less and less accurate according to high frequency changes in the spatial domain and a large amplification factor. As a result, the boundary of α is determined by the given video motion $f(x)$ and the image structure spatial wavelength λ. The equation is given as follows:

$$(1+\alpha)f(t) < \frac{\lambda}{8} \tag{2.5}$$

For motion magnification, however, artifacts are introduced significantly when the product of the transition function and factor exceeds the bound.

2.2 Non-contact heart rate measurements from video

Heart rate is a critical indicator in medical diagnosis and cure. Currently, the dominant measurement of cardiac pulse requires patients to wear equipment such as a chest strap, which is both inconvenient and uncomfortable. However, there is growing interest in contact-free heart rate measurements. As illustrated in Figure 2.5, imagine a "magic mirror" exists that which can both reflect the subject's image and also measure the subject's heart rate automatically.

Such remote photo-plethysmographic imaging (PPGI), introduced in the 1930s [29], enables the measuring the heart rate from the variation of color captured by cameras that provide comfortable physiological assessment without electrodes. Several studies have illustrated that PPGI is able to detect variation from such a distance[31] and could be used to measure heart rate [43], [53], [23] and [49]. This work can be considered as a preprocessing of Eulerian video magnification[55], which takes the heart rate as input for amplifying and visualizing skin color variation, caused by heart pulse. A demonstration that shows the color amplification can be found in [5].

Figure 2.5: Video-based heart rate measurement.[3]

Several studies have shown that as a result of the optical-absorbing property, skin color variation caused by blood circulation is much more significant on the green channel than on the red and blue channels [35][53]. However, these two channels also contained plethysmographic information.

In [43], Poh et al. designed such a "magic mirror" according to the subtle signal variations on human skin. The system is based on the principle light is absorbed during blood circulation[53]. Much more light will be absorbed as the blood flow volume increases through vessels, and vice versa. Therefore, the heart rate can be estimated by analyzing temporal variations on videos.

The overview of the general steps is shown in Figure 2.6. First, face detection is applied to detect and locate a subject's face from one video frame, which is the region of interest (ROI) for all of the frames. The MATLAB-compatible version of Open Computer Vision (OpenCV) library is used in this case. With the ROI located, all the pixels in this region are then averaged spatially for each color channel (RGB) to generate three 1-D observations,

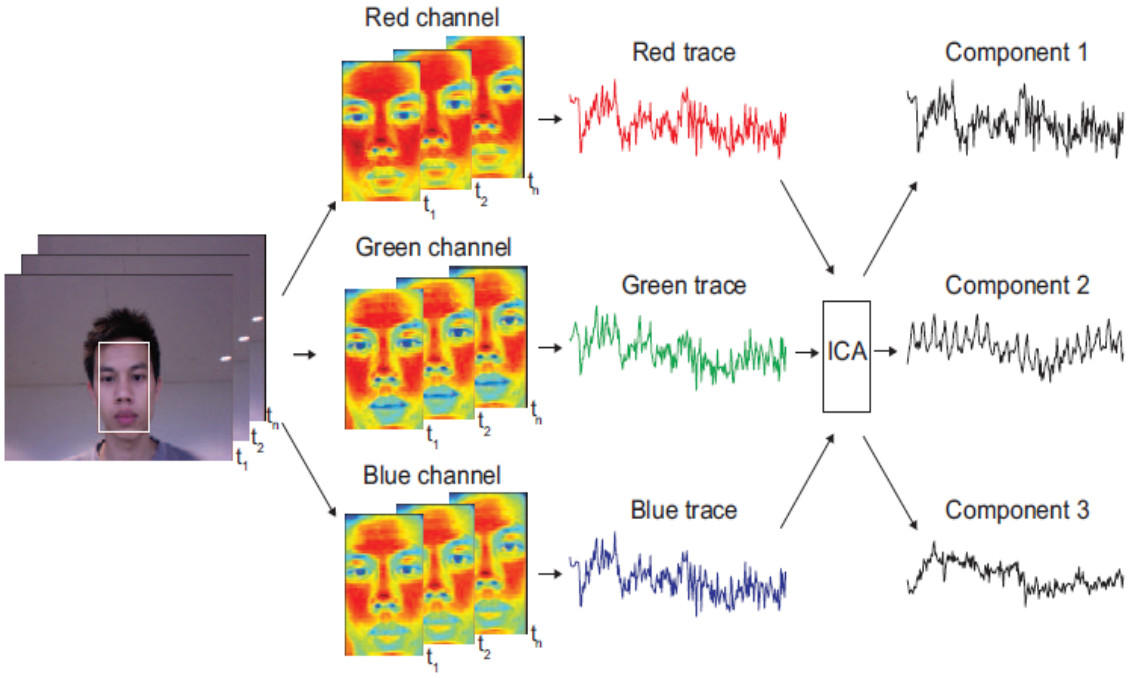

Figure 2.6: Video based heart rate measurement in the work[43].

$x_R(t)$, $x_G(t)$ and $x_B(t)$. ICA is also applied to uncover independent source signals from such a set of observations. This process can be expressed as:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{2.6}$$

where $\mathbf{x}(t) = [x_R(t), x_G(t), x_B(t)]^T$ and $\mathbf{s}(t) = [s_1(t), s_2(t), s_3(t)]^T$. $\mathbf{A}$ is a 3 by 3 square matrix. The inverse of matrix $\mathbf{A}$ is used to separate the mixed source signal. The second component after performing ICA contains the strongest plethysmographic signal. For simplicity, the second component is then selected as the input for the next step. The three 1D signals are now reduced to one.

The last step is exploiting the Discrete Fourier Transform (DFT) on selected signal ($s_2(t)$) and obtaining the power spectrum in the frequency domain. The heart rate frequency is designated as the maximum in the power spectrum within such a fixed range frequency. In the experiment, the authors set this range from 0.75 to 4 Hz, which corresponds to 45 to 240 beats per minute (bpm).

For the evaluation part, subjects were asked to wear a finger blood volume pulse (BVP)

sensor during video recording. The results comparison from proposed method and BVP sensor showed that a high accuracy rate was achieved.

2.3 Principal Component Analysis

2.3.1 Introduction

Principal component analysis (PCA) is a statistical method that converts a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. Assume we are given a dataset comprising a set of **n** observations of **m** variables

$$\begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(n)} \\ x_2^{(1)} & x_2^{(2)} & \cdots & x_2^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ x_m^{(1)} & x_m^{(2)} & \cdots & x_m^{(n)} \end{bmatrix}$$

where n and m are the number of observations and attributes, respectively. For example, the attributes of a soccer player could be technique, speed, physical fitness, strength and so on. However, some different attributes such as physical fitness and strength, $x_i^{(n)}$ and $x_j^{(n)}$, respectively, give a soccer player a general state of well-being and quality of being strong. These two attributes are probably linearly dependent, and only small differences exist between them. Therefore, the data can approximately rely on a $m-1$ dimensional subspace. And the goal of PCA is to remove this redundancy or, in other words, summarize the similar properties with fewer components.

In the soccer player example, let's consider a dataset with two attributes, where $x_i^{(n)}$ measures the physical fitness of a soccer player, and $x_j^{(n)}$ captures the player's technique. In soccer, if player can't pass, receive, or shoot the ball well, he or she will not be a good player. Similarly, a soccer game is a very physically power consuming, requiring a player to make varied types of movements. Thus, these two attributes $x_i^{(n)}$ and $x_j^{(n)}$ are strongly

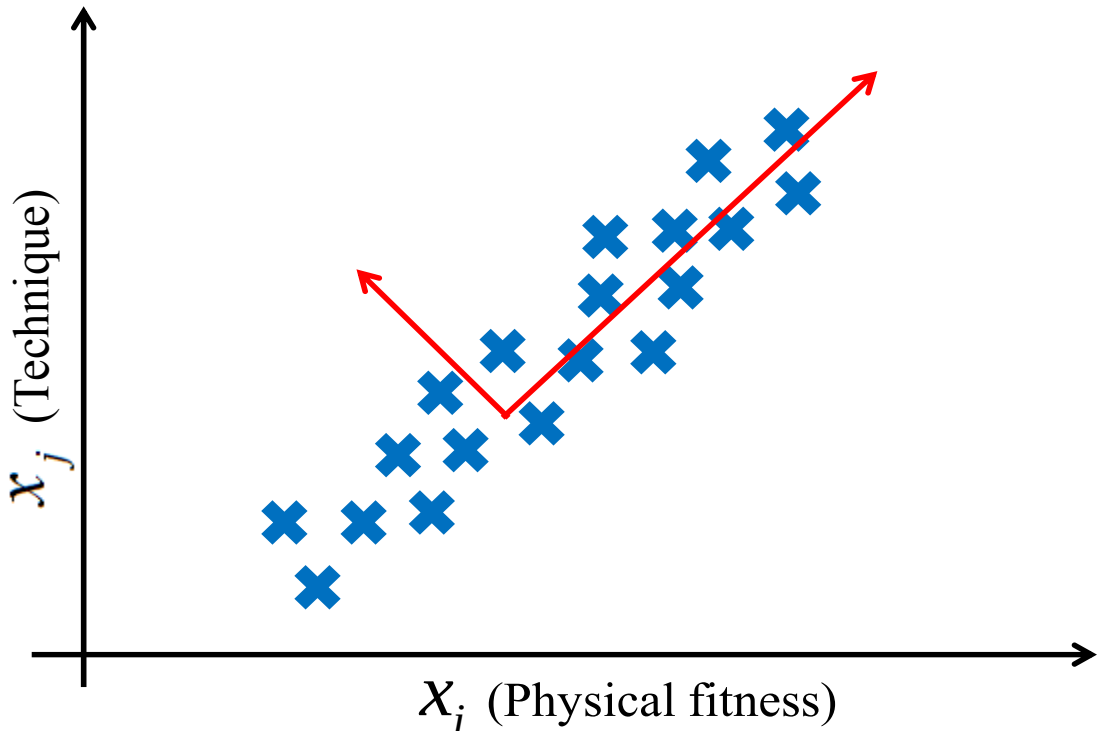

Figure 2.7: A dataset with two attributes of a soccer player.

correlated. Now, we are interested in capturing how good a player is using the data shown in Figure 2.7.

The diagonal axis in the figure captures how good the soccer player is with very little noise lying off this axis. Some other examples are also illustrated in[46] and [47].

2.3.2 Mathematical principles

To describe vector, we need to determine a basis set. The basis of a vector space is defined as a subset of vectors that are linearly independent. For example, in a standard 2D case, the basis is $[0,1]^T$ and $[1,0]^T$, and all the vectors in this space can be represented as the linear combination:

$$z = x \cdot [1,0]^T + y \cdot [0,1]^T \tag{2.7}$$

where z is the expression of all 2D vectors. The object of PCA is to compute the most important basis to re-express the source data. The ideal new basis after PCA should be

able to reveal the hidden structure where all the noise is filtered out.

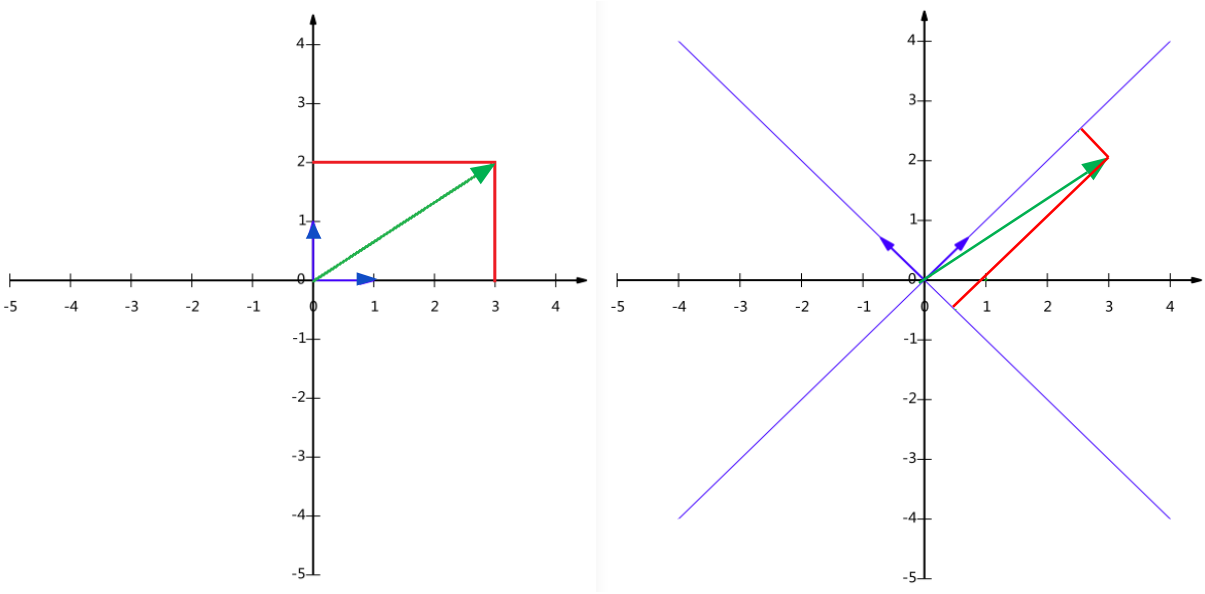

Figure 2.8: Vector $z = [3, 2]^T$ represented by different basis.

In 2D space, for example, with basis set of $[0, 1]^T$ and $[1, 0]^T$, the vector $z = [3, 2]^T$ can be represented as

$$z = 3 \cdot [1, 0]^T + 2 \cdot [0, 1]^T. \tag{2.8}$$

The basis can be also determined as $[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]^T$ and $[-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]^T$. To get the new coordinates of vector $z = [3, 2]^T$, according to the inner product, the linear combination with a new basis is now expressed as

$$z = \frac{5}{\sqrt{2}} \cdot [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]^T - \frac{1}{\sqrt{2}} \cdot [-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]^T. \tag{2.9}$$

This process is illustrated in Figure 2.8. To find the major axis of variation, the solution is to get the unit vector u from the transformed basis so that when the data is projected onto the u direction, the variance of the projected data is then maximized.

Given the dataset shown in Figure 2.9, the projected data has a very large variance in the u_1 direction. Conversely, if direction u_2 is selected, the variance of such projections will be extremely small.

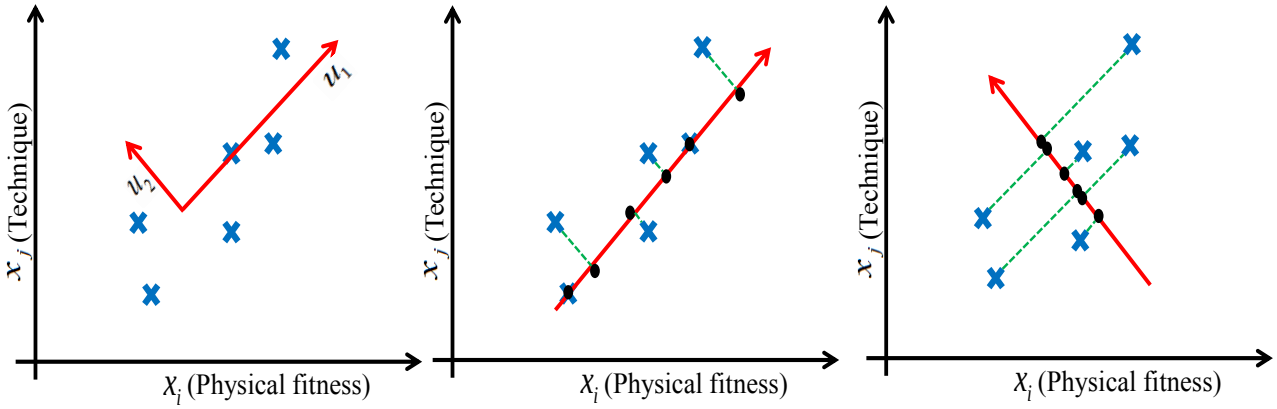

Figure 2.9: Illustration of projection in different direction.

As discussed above, we want to identify the direction u_1 corresponding to Figure 2.9. In other words, the projections are expected to be as dispersed as possible, which is described as variance in mathematics. Typically, we first normalize the mean and variance of input data as preprocessing:

1. Set $\mu_m = \frac{1}{n}\sum_{i=1}^{n} x_m^{(i)}$.
2. Set $\sigma_m^2 = \frac{1}{n}\sum_{i=1}^{n}(x_m^{(i)} - \mu_m)^2$.
3. Replace each $x_m^{(i)}$ with $\frac{x_m^{(i)} - \mu_m}{\sigma_m}$.

In these steps, m and n correspond to variables and observations of data, respectively. Steps (1-2) calculate the mean and variance of input data and step 3 rescaleS each coordinate to have zero mean and unit variance so that different attributes are then treated on the same scale. Now we are given the unit vector u and data points $x_m^{(i)}$, the distance from projected point to origin is $(x_m^{(i)})^T u$ and the target is to maximize the variance of these projections $\frac{1}{m}\sum_{i=1}^{n}[(x_m^{(i)})^T u]^2$, which can be expanded as

$$\begin{aligned} D = \frac{1}{m}\sum_{i=1}^{n}[(x_m^{(i)})^T u]^2 &= \frac{1}{m}\sum_{i=1}^{n} u^T (x_m^{(i)})(x_m^{(i)})^T u \\ &= u^T \left(\frac{1}{m}\sum_{i=1}^{n}(x_m^{(i)})(x_m^{(i)})^T\right) u \\ &= u^T C u \end{aligned} \tag{2.10}$$

where C and D are the covariance matrix of original data $x_m^{(i)}$ and projected data $(x_m^{(i)})^T u$. The transformed space corresponds to the eigenvector of C; in other words, to find a 1D subspace, which represents the approximation of original data, we should choose u to be the principal eigenvector of C. More generally, if we want to represent our input data by a k-dimensional subspace, we should eigenvector u_1 to u_k corresponds to be the biggest eigenvalues λ_1 to λ_k.

An implementation of PCA can be found in [34]. In MATLAB, we could use the following function:

```
COEFF = princomp(X)
[COEFF,SCORE,latent] = princomp(X)
[COEFF,SCORE,latent,tsquare] = princomp(X)
```

The function performs PCA on the n by p original data matrix X, and returns the principal component coefficients. Rows and columns of X correspond to observations and variables, respectively. $COEFF$ is a square matrix in the size p, and each column contains coefficients for one principal component, which is in the order of decreasing variance. $SCORE$ is the representation of X in the principal component space, and *latent* contains the eigenvalues λ of the covariance matrix of X. The detail of PCA on data can be found in [32] and [34].

2.4 Classifying with k-Nearest Neighbors

In pattern recognition and machine learning, the k-Nearest Neighbors algorithm (k-NN) is a non-parametric method used both for regression and classification[7]. As one of the simplest of all machine learning algorithms, the k-NN algorithm has advantages in high accuracy, insensitivity to outliers and no assumptions about training or input data. However, it is computationally expensive and requires a lot of memory. In this section, we will discuss the principle of the k-NN algorithm and method to find the optimal value for k by cross-validation[37].

2.4.1 Classifying with distance measurements

In machine learning tasks, supervised learning infers a function from labeled training data, which consists of a set of training examples. In this process, we have the training data, which is an existing set of example data. All of these data have been labeled, which means we know exactly which class each piece of data should belong to. When we are given a new set of data without a label, we compare each new piece of data to the existing labeled data and determine what class it should fall into. Common supervised learning classification algorithms include: k nearest neighbors (k-NN), support vector machines (SVM), discriminant analysis, nave Bayes classifier, neural networks and decision trees. The details of supervised learning algorithms can be found in [11] and [16].

For the k-NN algorithm, given a piece of new data, we take the most similar pieces with their labels from training date (the nearest neighbors) and choose the top k most similar ones from the training set. The commonly used distance metric of nearness is the Euclidean distance by default. This distance between two vectors x and y with only two features is given by

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \tag{2.11}$$

where x and y are training data and new observations with two elements. Following the distance calculation, the distances between specific observation and all training data are sorted from greatest (can be first or lowest) to worst. Next, the greatest k distances are used to vote on the class of input data. Lastly, we take a majority vote from the top k, and the class of majority is assigned to the new data. In other words, the new data is assigned to the class most common among the training set neighbors, which have the nearest distance to its location in the feature space. This method can be illustrated in the Figure 2.10.

We have a training set consisting of two classes (A and B) with an equal number of instances, as indicated by red squares and green dots. The feature-space is constituted by x_1 and x_2. We are now given an unlabeled observation, indicated by the gold triangle.

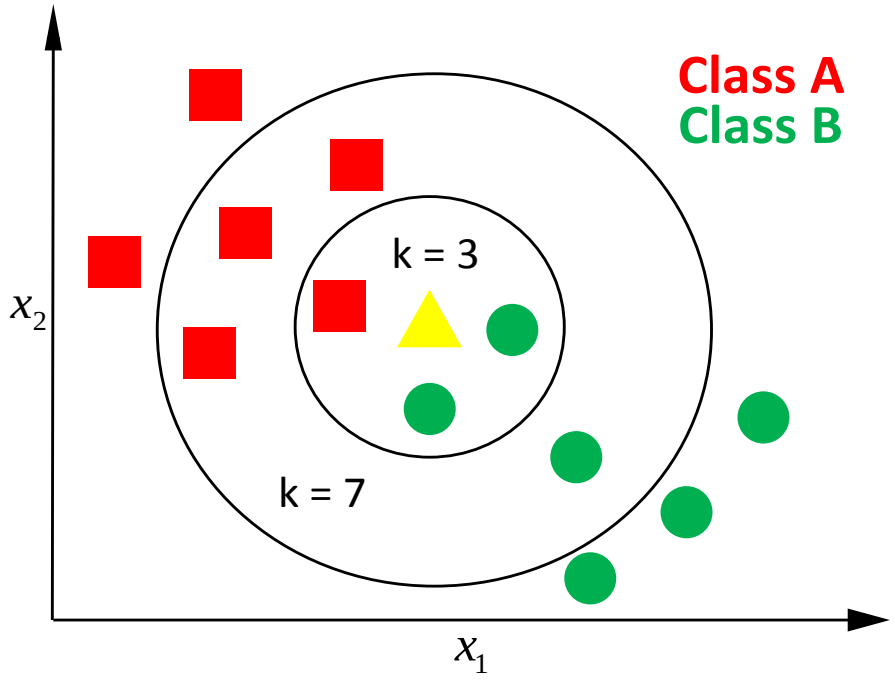

Figure 2.10: A simple example of k-NN with different k.

We do not know the class of this unknown data is, but we can use k-NN to make that determination. First, we calculate the distance to all the data in the training set and then assign the class by a majority vote of the k nearest neighbors. Let us assume $k = 3$; in the determined circle, one out of three neighbors belong to class A, and two belong to class B. Therefore, we forecast that the unlabeled observation is classified into class B.

2.4.2 Cross-validation in k-NN

For k-NN classification, we need to find a value for k. In Figure 2.10, the unlabeled observation is classified into class B by assuming $k = 3$. However, if we widen this circle by increasing the value of k, for example, for $k = 7$, four neighbors are of class A and only three are of class B, the unlabeled observation is then classified into class A instead of B. Therefore, to minimize classification errors, the proper value for k should be chosen. Generally, in the classification, larger values of k reduce the effect from noise but make boundaries between classes ambiguous[18]. In other words, in k-NN classification, k acts as a smoother.

Cross-validation is a technique used for assessing the accuracy of a learning model in either regression or classification problem. We divide training data into a training set and a validation set, the error value for the validation set are then determined and verified by data from the training model. An example of cross-validation in regression problem is illustrated in Figure 2.11. The blue and red cross are the training and validation set, which are used for building model and to evaluate the training model, respectively.

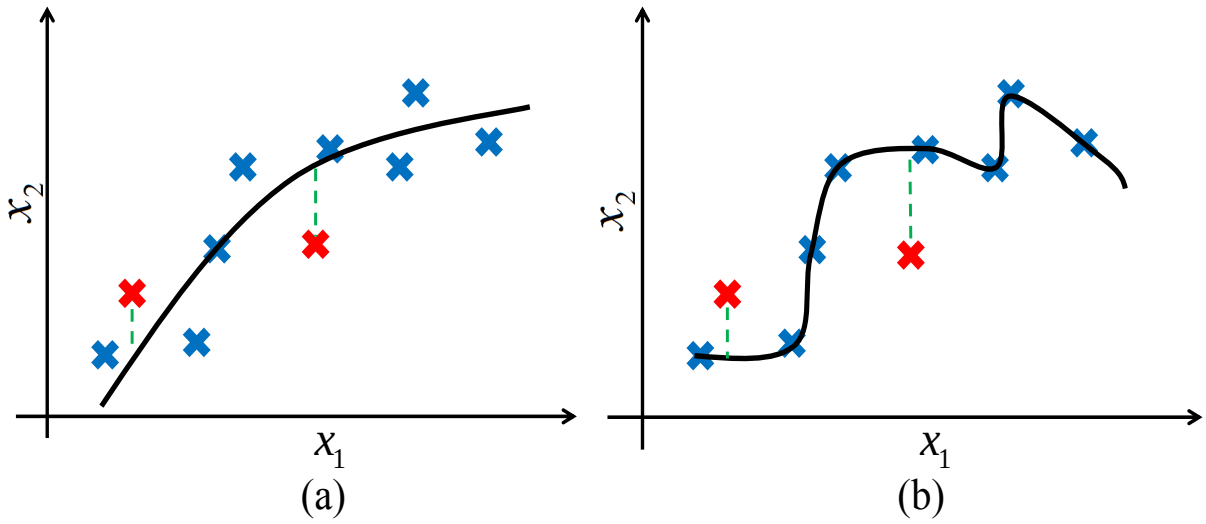

Figure 2.11: An example of cross-validation in regression problem.

As discussed above, we need to find a value for k in the k-NN classification, which minimizes prediction error by a defined loss function. Firstly, we divide training data into two sets: a training set $(80 - 95\%)$ and a validation set $(5 - 20\%)$, the class labels for the validation set are then determined and verified by data from the training set. Lastly, we choose the number of neighbors k that maximizes the classification accuracy.

In summary, the k-NN algorithm is a straightforward and effective way to classify data. With cross-validation, it can provide a very accurate result. However, this algorithm has to iterate the full dataset, which implies, for large datasets, it will take a large amount of memory. In addition, for each piece of unlabeled observation, the distance must be calculated for every piece of data from the training set, which is very time-consuming. In our case, all the input features are extracted from a video; as a result, the spent storage and time is acceptable.

Chapter 3

Methodology

The color of human skin varies slightly over time as a result of blood circulation. This variation can be visualized by Eulerian video magnification [55]. Each face pixel in a video thus constitutes a noisy 1D sinusoid signal over time with a frequency corresponding to the heart rate. The phase of this signal expresses the relative time that it takes for the blood to reach this part of the face. Our approach exploits this fact by taking a short video of a person's face as input and producing an estimate of the velocity field of the blood flow from the relative phase shifts between the pixels. An overview of the entire process is given in Figure 3.1. The input video is first low-pass filtered and downsampled to improve the signal-to-noise ratio (SNR) and computational efficiency. To make the best use of the three color channels, a principal component analysis (PCA)[34] is performed, and the component containing the largest color variation over time is selected. These preprocessing steps are contained in Section 3.1 and Section 3.2. Section 3.3 describes noise reduction from our previous work, followed by multi-kernel k-NN classification for segmenting blood flow positions in section 3.4. Finally, section 3.5 illustrates the estimation and visualization of blood flow. A temporal discrete Fourier transform (DFT) over the 3D video cube is performed, and the amplitude and phase map corresponding to the known heart rate are extracted via a band-pass filter. Two noise reduction techniques further process the maps. As the last step, the desired velocity field is computed as the gradient of the phase map.

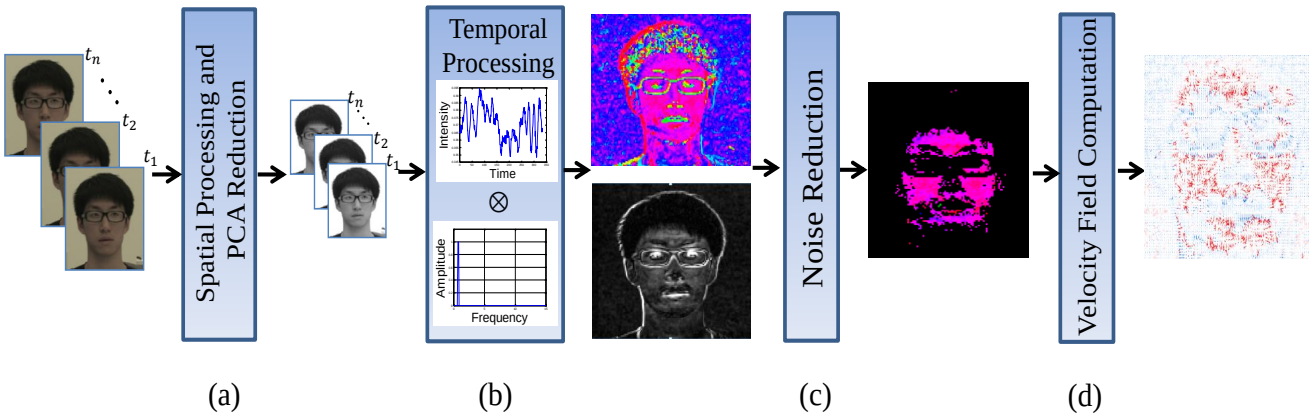

Figure 3.1: Overview of the Non-contact blood flow estimation framework.

3.1 Space-time video processing

Our approach combines spatial and temporal processing to reveal two dimensional velocity field blood flow in the temporal frequency band of heart rate (HR). The goal of spatial processing is to increase the temporal signal-to-noise ratio (SNR) and improve computational efficiency by pooling multiple pixels together. We then perform temporal processing on each position from downsampled 3D color signals (video cube). The variation of a pixel's value is considered as the time series, and we apply a band-pass filter to extract the frequency bands of heart rate (HR).

3.1.1 Spatial averaging to improve SNR

As a result of the strong intensity of noise inherent in the video, the variation of heart rate signals is always invisible. In the general case, a full Gaussian pyramid [14] is computed to enhance these subtle signals. In our case, however, we apply proper spatial filtering and downsampling. The level of spatial downsampling is a trade-off between SNR and resolution. Sufficient spatial pooling reveals the signal of the heart rate; however, we will lose too much information if the spatial filter applied is too large. Figure 3.2 shows five levels of Gaussian pyramid for one frame of an example video. The original image, on the far left, measures 720 by 640. This becomes level 0 on the pyramid. At each higher level, the image is a quarter of its previous level due to reduced sample density.

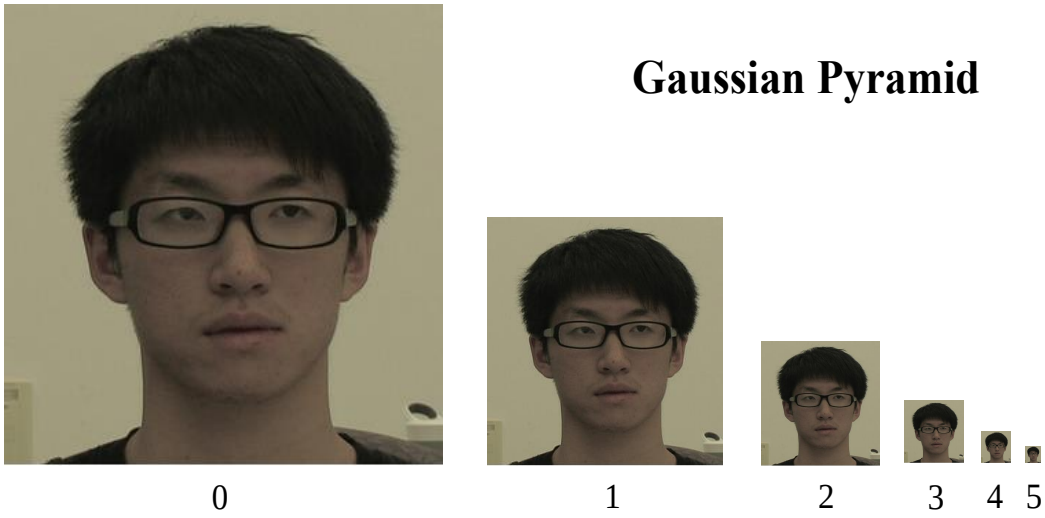

Figure 3.2: A 5-level Gaussian pyramid of one frame from the face video.

The effect of the Gaussian low-pass filter is shown on the rightmost photo, over-spatial processing results in the lost of face information. However, if the spatial filter applied is not sufficient enough, the signal of pulse will not be revealed(Figure 3.3).

Figure 3.3(a) shows a frame from the face video with white Gaussian noise added. (b) and (c) presents the trace obtained when the (noisy) sequence is processed with the same spatial filter used to process the original face sequence but with a different spatial processing level (1 and 4, respectively). The pulse signal is not visible in (b), as the noise level is higher than the power of the signal, while in (c) the pulse is clearly visible (the periodic peaks about one second apart in the trace).

The input face video provides a 3D color signal (video cube) with 24 bits of color information for each sample. We first perform low-pass filtering and downsampling on each frame to get a new 3D color signal with reduced spatial size. This spatial processing not only increases the SNR of each pixel but also reduces the workload for processing. The result is a video cube $I_c(x, y, t)$, where x and y are the horizontal and vertical pixel position, respectively, and t is the frame number. $c \in \{R, G, B\}$ represents the three color channels.

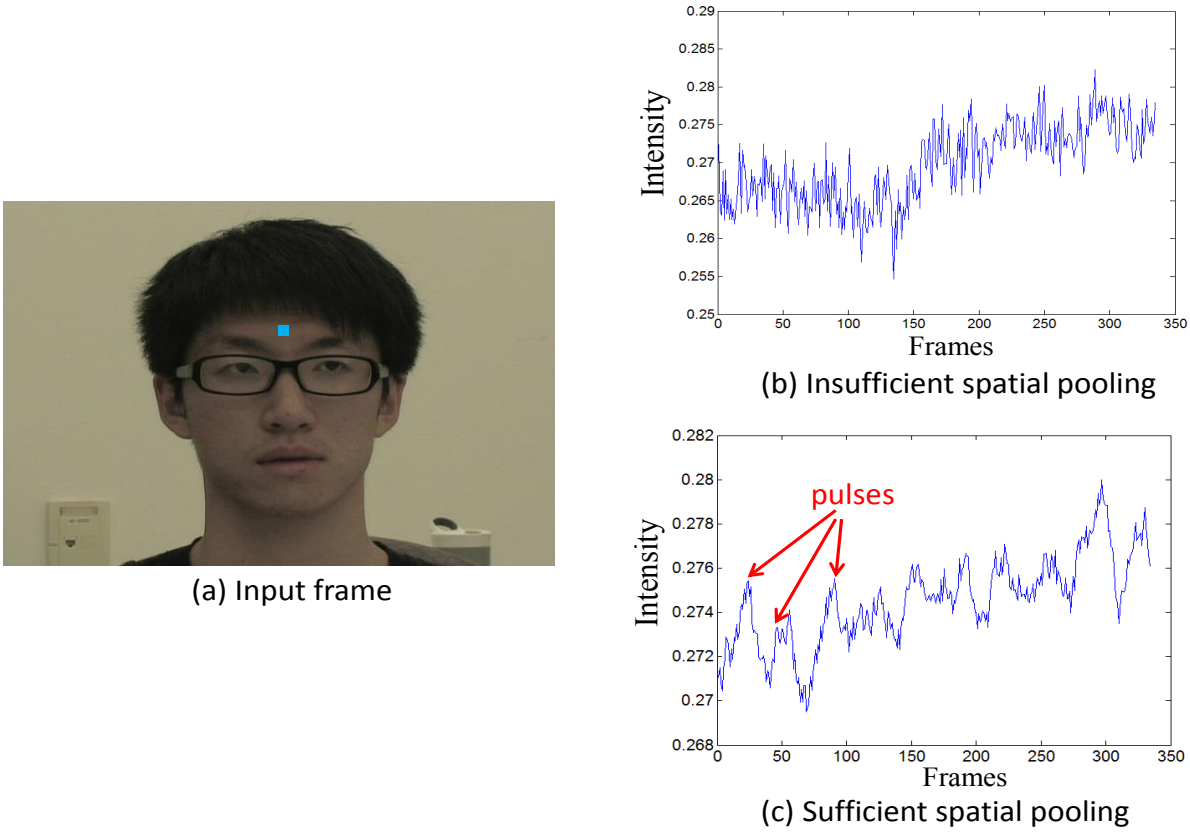

Figure 3.3: The illustration of proper spatial pooling for revealing the signal of interest.

3.1.2 Digital filtering and spectral analysis

It was shown in[53] that compared with the red and blue channel, the green channel contains the strongest color variation due to blood circulation. As mentioned in the preceding section, a new 3D color signal $I_c(x, y, t)$ with reduced spatial size is obtained by performing low-pass filtering and downsampling on each frame.

For a fixed pixel position (x', y'), the video gives us a 1D signal $I_G(x, y, t)$ that varies over time t. We apply the discrete Fourier transform (DFT) on such a 1D signal over time. Fast Fourier transforms (FFT in MATLAB) were used to determine the amplitude and phase spectra for $I_G(x, y, t)$. When performed for all pixel locations, the video is transformed into the frequency domain. We denote this by $F_G(x, y, k)$, where k is the index of temporal frequency after the DFT. Note that I_G and F_G have the same size, but each element in F_G has a complex value.

An ideal band-pass filter is then applied to F_G that filters out all frequencies except for the heart rate of the subject in the video. This is implemented by discarding everything but the selected index k' corresponding to the heart rate: $F_G(x, y) := F_G(x, y, k')$. In our previous work, the heart rate is determined by a sensor on the subject's chest. However, it can also be determined automatically by finding a peak in the frequency spectrum as was shown in [43] and [53].

Pulse amplitude mapping

The complex-valued signal $F_G(x, y)$ is index k' of the Fourier transform of each pixel. The amplitude of each position is then calculated as

$$|F_G(x, y)| = \sqrt{Re(F_G(x, y))^2 + Im(F_G(x, y))^2} \tag{3.1}$$

where $|F_G(x, y)|$ is the real-valued magnitude map. It indicates the amplitude of the sinusoidal skin color variation at the frequency of the heart rate. Figure 3.4(b) shows the magnitude map of an example video. The magnitude is strongest in areas of exposed skin and weaker in occluded areas and the background.

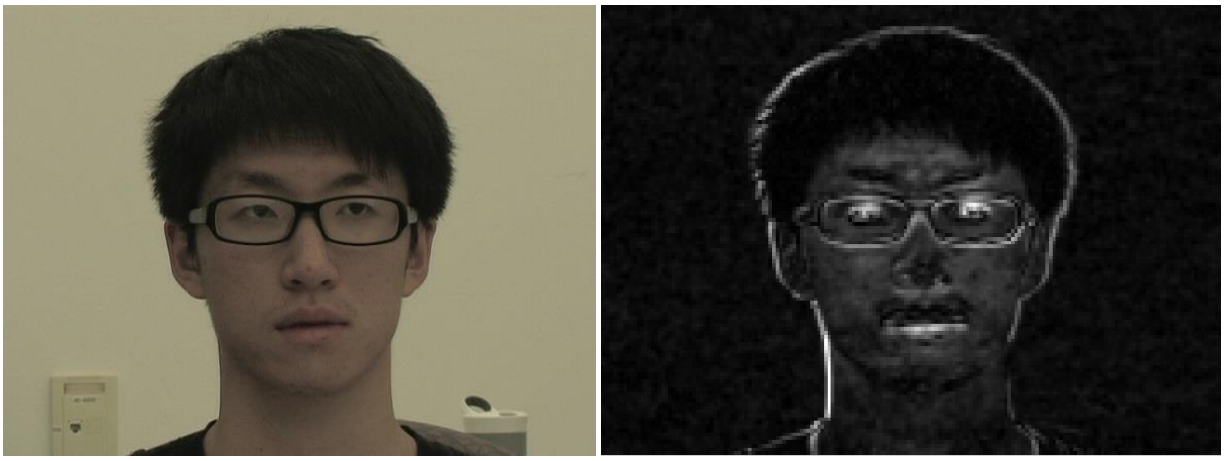

Figure 3.4: Magnitude map of the video in the heart rate frequency; the spatial processing level is 2.

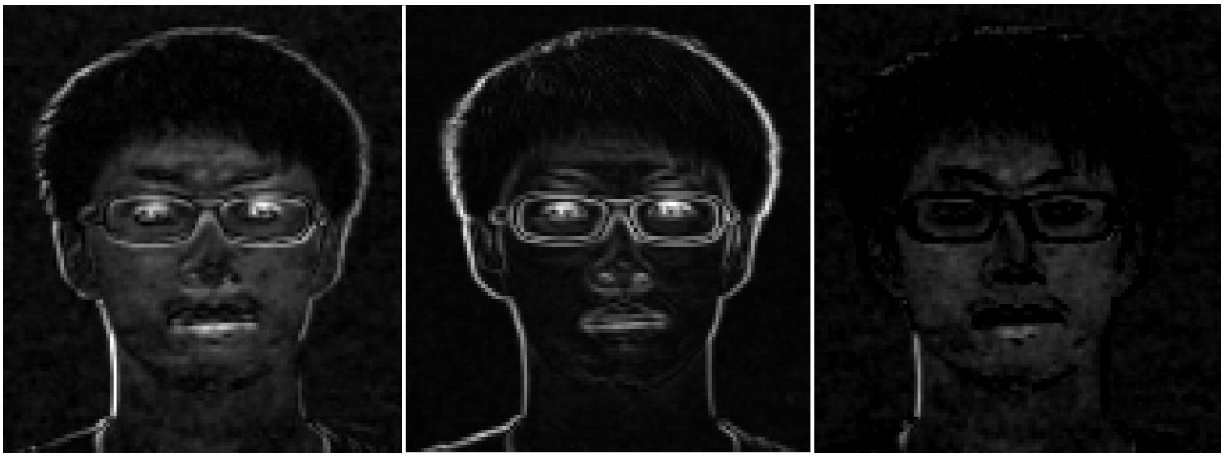

Figure 3.5: Phase map of the video in the heart rate frequency; the spatial processing level is 2.

However, the magnitude is also strong at the edges inside and around the face. This is a consequence of involuntary movements of the subject's head at random frequencies. To remove these movement artifacts, we employ a technique that has been proposed similarly in [53]. The assumption is that motion artifacts mostly induce "white noise" and are equally strong in all frequency bands, while the high magnitude due to blood circulation only appears in band k'(0.92 Hz of this subject).

A "movement artifact map" is defined by averaging the magnitude maps of neighboring bands around the heart rate frequency. As shown in Figure 3.5, such a map indicates that as the result of movement, the magnitude is strong at the edges inside and around the face.

By subtracting the averaged magnitude maps of neighboring bands, the strength of the artifacts can thus be decreased. We obtain a denoised magnitude map $|F'_G(x,y)|$ in the green channel as

$$|F'_G(x,y)| = |F_G(x,y)| - \frac{1}{2|\Omega|}\sum_{c\in\Omega}|F_G(x,y,k'\pm c)| \tag{3.2}$$

where Ω contains the relative indices of neighboring frequency bands to be considered. Figure 3.5(b) indicates that noise areas with high contrast are indeed associated with edge

positions. Figure 3.5(c) shows the example magnitude map after subtracting the neighbor frequency bands. The resulting map $F'(x, y)$ now correctly indicates the strength of the blood flow at each pixel location.

Phase mapping

Similarly to the magnitude map, the phase of the complex valued $F_G(x, y)$ can be computed at each pixel location, leading to a phase map $\phi_G(x, y)$:

$$\phi_G(x, y) = atan2(Im(F_G(x, y)), Re(F_G(x, y))) \tag{3.3}$$

The entries in the phase map are real-valued, periodic and limited to the interval of $(-\pi, \pi]$. Because the value of the hue is also periodic (from 0 to 2π), we visualize these periodic values by mapping them to the periodic component hue of the HSV color space while keeping saturation and value at maximum. This is illustrated in Figure 3.6.

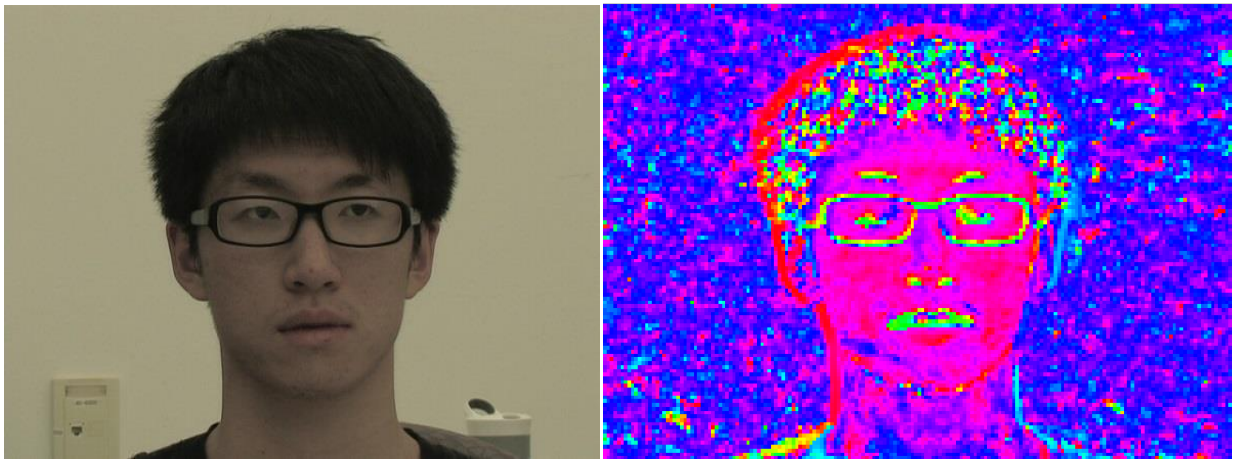

Figure 3.6: Phase map of the example video in the HSV color space; the spatial processing level is 2.

It can be seen that phase varies smoothly at adjacent positions inside the facial region, whereas it appears more random in non-skin pixels. This can be exploited as follows. If

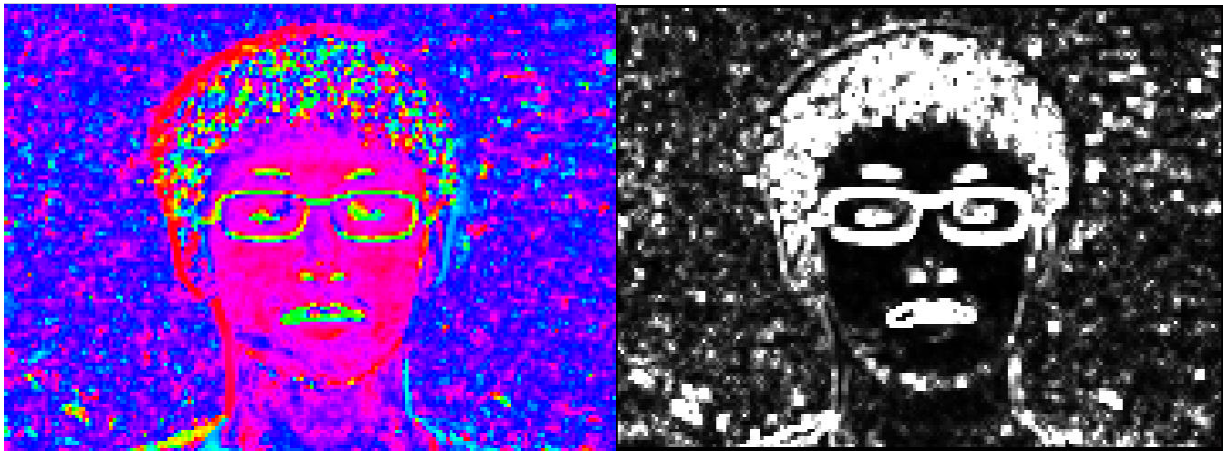

Figure 3.7: Phase variance map from the phase map; the spatial processing level is 2.

there is a large phase shift in adjacent pixel locations, we can assume that this is due to noise, and such locations should be ignored during the estimation of blood flow.

Similarly, phase varies smoothly at adjacent positions inside the facial region as the result of blood flow, whereas it appears more random in non-skin pixels (like background and hair). We assume that large phase shifts in adjacent pixel locations are the result of noise; these shift are unrelated and should be ignored during the estimation of blood flow. To express this numerically, we define an energy function E as

$$E_G(x,y) = \sum_{i,j=-1}^{1} (\phi_G(x,y)) \ominus \phi_G(x+i, y+j))^2, \tag{3.4}$$

which evaluates the variance of phase values in a small neighborhood around each pixel. The minus operator $\ominus$ is defined in such a way that it takes the periodicity of the phase into account. For example, for two phase values 0.9π and -0.9π, the difference is -0.2π instead of 1.8π. Figure 3.7 shows the energy map of the phase shown in Figure 3.6. Areas with highly random phase variations mainly in the hair and near edges appear in white.

3.2 Heart rate measurement and enhanced spectral analysis

In the last section, the heart rate is detected by using a heart rate sensor to obtain corresponding amplitude and phase map. However, to design a non-contact blood flow estimation system, we propose a method similar to [43] to determine the heart rate automatically from the green channel. Meanwhile, for the sake of obtaining a stronger blood flow signal, PCA is applied to remove redundancy information.

3.2.1 Heart rate measurement

Instead of using a sensor, we detect the subject's heart rate from the video automatically. In particular, although not the focus of this thesis, we will show how we extract the heart rate from standard RGB face videos via analysis of the subtle variations of color on the skin caused by blood flow.

The currently predominant techniques for measuring heart rate require patients to wear inconvenient equipment that can be uncomfortable or even painful. Non-contact heart rate measurement can provide a comfortable physiological experience for patients. As a result, Verkruysse et al.[53] and Poh et al.[43] extract pulse rates from video by utilizing the subtle color changes on the skin due to blood circulation. Balakrishnan et al.[9] extract the heart rate and beat lengths from videos by measuring subtle head motion caused due to the Newtonian reaction to the influx of blood at each beat.

Following [43] and [53], the heart rate is determined automatically by finding a peak in the frequency spectrum in the green channel. As we discussed earlier, the skin color constantly changes very slightly over time as the result of blood flow. This subtle signal can be extracted and revealed in the frequency domain by using the Fourier Transform. Our process is illustrated in Figure 3.8.

An automated face tracker is first used to detect faces within the first frame in the video and to localize the region of interest (ROI) for each video frame. We utilized the Open

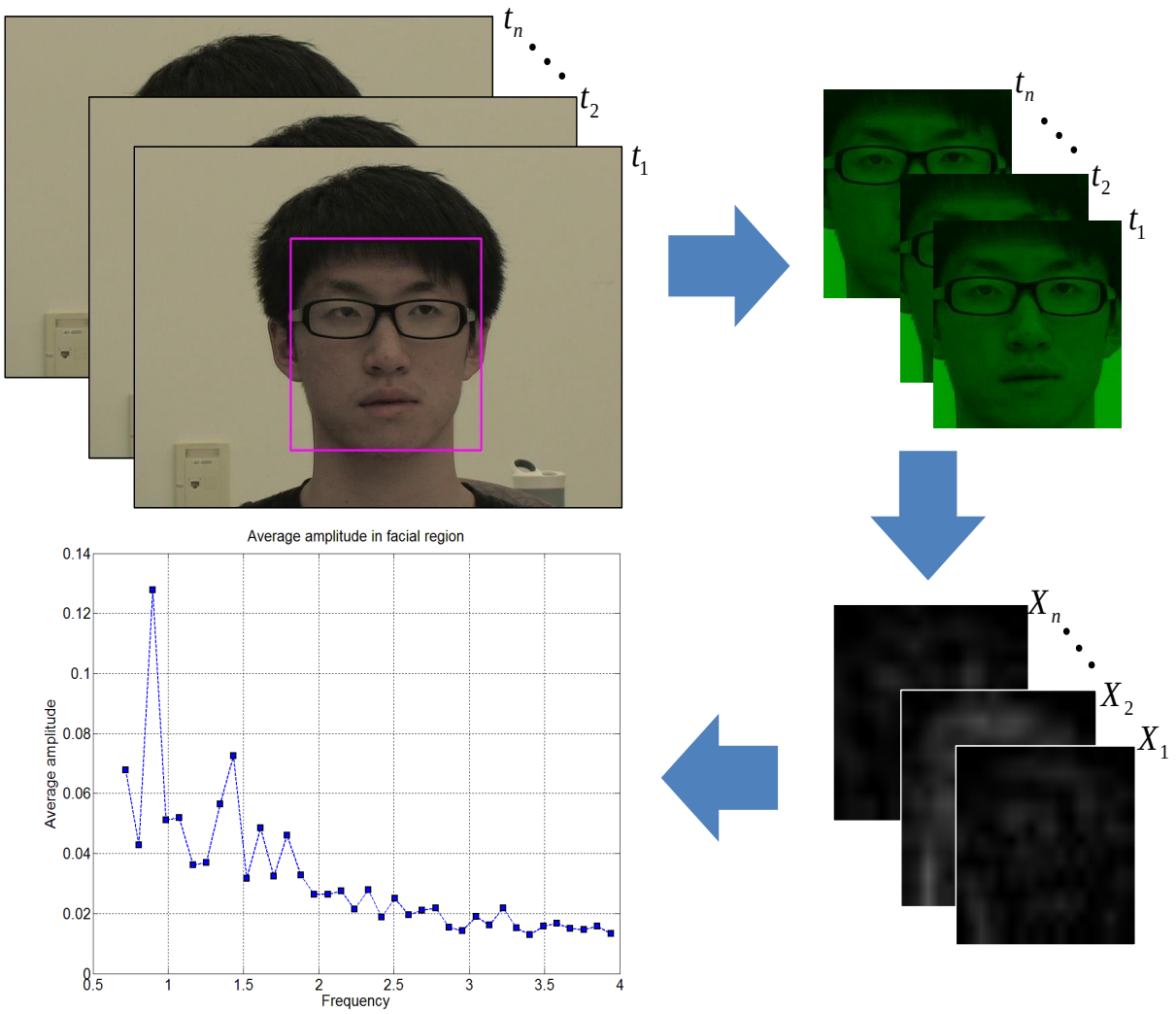

Figure 3.8: Heart rate detection by using only green channel.

Computer Vision (OpenCV) library to obtain the face location and size of it. In OpenCV, the face-detection algorithm is based on Haar feature-based cascade classifiers, which is proposed by Viola and Jones[54]. It is a machine learning-based method where a lot of positive images (images of faces) and negative images (images without faces) are captured to train the classifier. For one frame from the original video, the algorithm returns the spatial positions (x- and y- coordinates) along with the size that defines a rectangle on the face. The outputs are then switched into scaled facial location, which is adaptive to the size of the downsampled 3D color signal.

We then apply the same spatial processing described in Section 4.1.1 using a Gaussian

low-pass filter and downsampling of three or four levels, and filter each pixel sequence selecting frequencies between 0.67 and 4Hz (corresponding to 40-240 beats per minute). The spatial processing is simply used to reveal the subtle variation, and the wide band-pass filter produces a more noise-free result. With the complex-valued signal $F_G(x, y, k)$, the amplitude of each coefficient k at spatial location (x, y) is then calculated as follows:

$$|F_G(x, y, k)| = \sqrt{Re(F_G(x, y, k))^2 + Im(F_G(x, y, k))^2} \tag{3.5}$$

As we discussed above, because of involuntary movements of the subject's head, the artifacts are strong at the edges inside and around the face. To get rid of effect of the motion artifacts, we exploit edge detection techniques (e.g., Canny filter, Sobel operator, Laplacian derivative, etc.) to remove this information and make our estimation more robust. The details of edge detection can be found in [15][40].

Lastly, We average the value of amplitude over the entire face region of the subject in the green channel; the edge positions determined by edge detection are also left out. The heart rate is eventually determined by finding a peak from the band-passed signal in the frequency spectrum.

3.2.2 Enhanced spectral analysis by PCA

It was shown in [52][43][53] that the green channel contains the strongest color variation due to blood circulation. However, to obtain a stronger blood flow signal, all three channels should be considered. We perform PCA to achieve a reduction of dimensionality and to remove redundancy from the three channels, as illustrated in Figure 3.9.

The good blood flow positions in the facial region are chosen as the training data for PCA. To achieve this, the face detection and edge detection are first applied to locate the facial region from the downsampled video cube $I_c(x, y, t)$ and remove the most-likely motion artifacts, respectively. All the spatial positions outside the facial region or with edge information will be ignored in further steps. These can be synchronized with heart rate measurement by using the same level of downsampling.

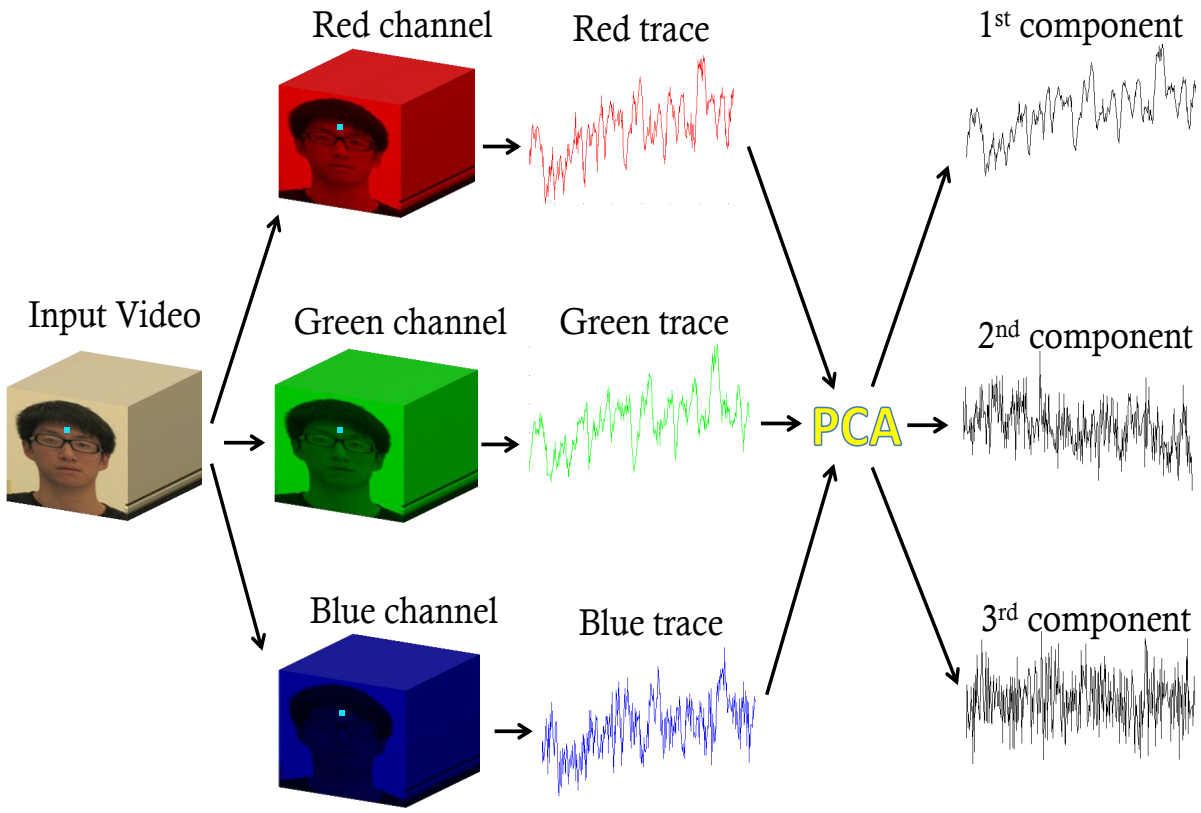

Figure 3.9: PCA projects most of the variation due to blood flow onto the first component, while the other two mainly contain variation from noise and motion.

Following the heart rate measurement, we now end up with $F_G(x, y, k)$ after transforming the signal from time domain to frequency in the green channel. The signals in heart rate frequency $F_G(x, y)$ are then obtained by disgarding everything but the index k' corresponding to the peak. The positions of training data for PCA are selected

$$T(x, y) = w_1 \times |F_G(x, y)| - w_2 \times |F_G''(x, y)| - w_3 \times E_G(x, y) \tag{3.6}$$

where $T(x, y)$ represents the training data score. $|F_G(x, y)|$ and $|F_G''(x, y)|$ are the scaled amplitude map in the HR frequency band and its averaged neighboring bands; $E_G(x, y)$ is the scaled variance from the phase map in the HR frequency band. All the data are scaled into the range [0,1]. w_1, w_2 and w_3 are the weights of the three maps, respectively, and are all set as 1 by default. The training data scores are only calculated on remaining positions from face location and edge detection and then ordered in descending order, the

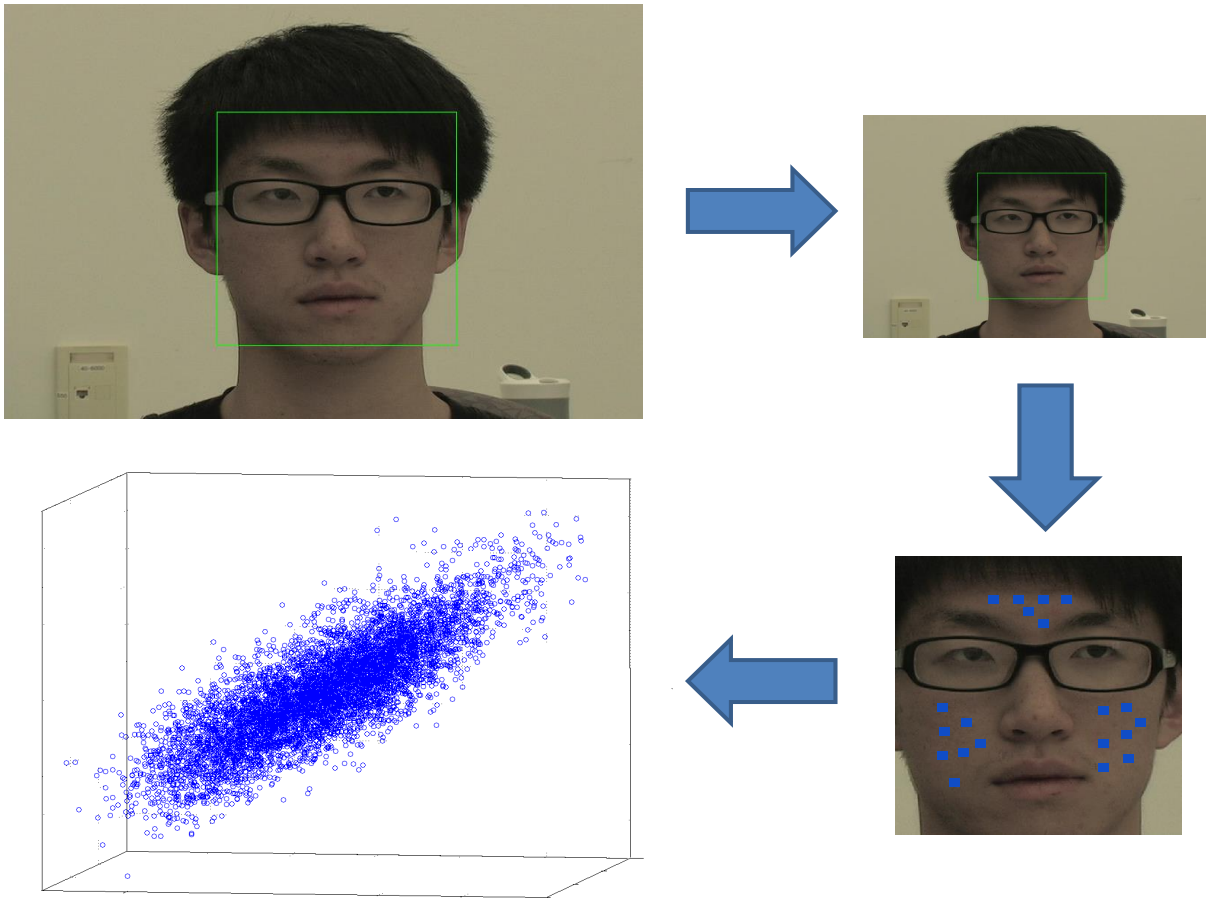

Figure 3.10: Face pixels with strong color variation are used for the PCA.

first n scores are eventually selected. This process is shown in Figure 3.10.

The PCA calculates a transformation matrix that projects most of this variation onto a single component. In conventional PCA, the number of recoverable components is as same as the number of observations, thus we assumed three components, represented by $I_1(x, y, t)$, $I_2(x, y, t)$ and $I_3(x, y, t)$. The whole process can be illustrated by

$$I_{COM}(t) = AI_{COM}(t) \tag{3.7}$$

where the column vectors $I_{COM}(t) = [I_1(t), I_2(t), I_3(t)]$, $I_C(t) = [I_R(t), I_G(t), I_B(t)]$ and A forms a 3×3 transforming base, and the column of $I_{COM}(t)$ are in order of decreasing component variance. Before training, the raw RGB traces, which are selected in the last step are first normalized as follows:

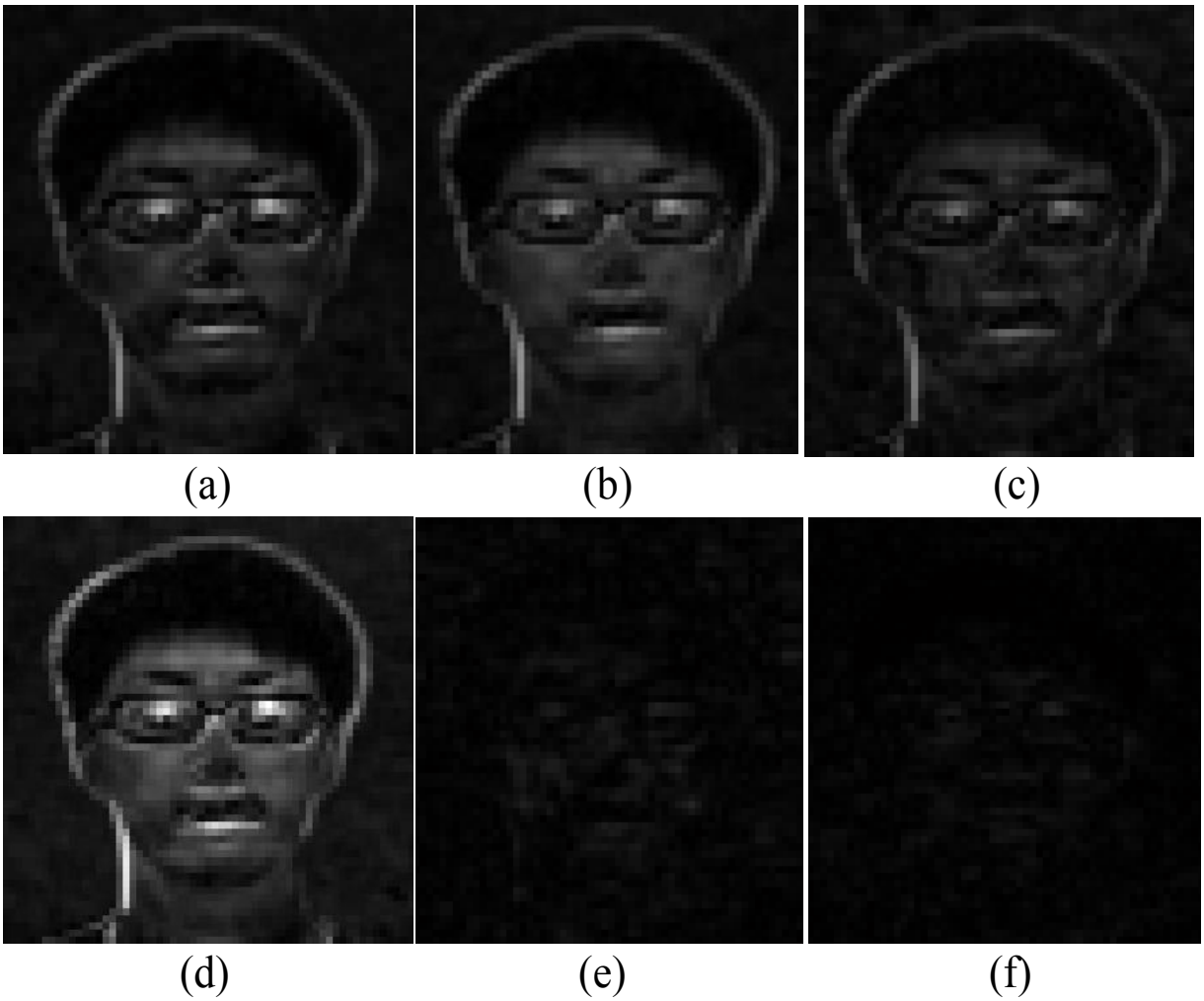

Figure 3.11: A comparison of amplitude mapping before and after PCA in HR frequency; the spatial processing level is 3.

$$I_i^{'}(t) = I_i(t) - \mu_i \tag{3.8}$$

For each $i = 1, 2, 3$, μ_i is the mean of $I_i(t)$. The normalization transforms $I_i(t)$ to $I_i(t)^{'}$ which is zero-mean. As a result, we obtain a resized single-channel video cube $I(x, y, t)$. We calculate the pulse amplitude mapping $|F_{COM}(x, y)|$ after performing PCA and compare it with the mapping on the RGB channels $|F_C(x, y)|$. An example is shown in Figure 3.11.

We see that compared with the green channel (Figure 3.11(a)), which contains the strongest amplitude on the facial region, the amplitude mapping in the red and blue chan-

nels (Figure 3.11(b) and (c)) also provide abundant blood flow information, which is valuable in our case. Figure 3.11(d)-(f) illustrates the amplitude map in the first, second and third component (in the order of decreasing component variance). The first component of the amplitude map is not much stronger than the other two components, but it is also stronger than the amplitude in each channel before PCA.

Similarly, the corresponding phase mapping in the heart pulse frequency is visualized in Figure 3.12. The phase value in the RGB color channels and first to third components are shown from (a)-(c) and (d)-(e), respectively.

As the result of blood circulation, the values of the phase shift on the facial region in different channels are very close; however, the value is completely different on the background positions. After applying PCA, most of the blood flow information are ended up in the first component. Therefore, the value of phase from the second and third components varies a lot, which means there is no such useful information (blood flow).

It can be seen that performing PCA achieves a reduction of dimensionality and removes redundancy from the three channels. We only consider the first component, and all the information from other two will be ignored in further processing.

3.3 Noise reduction

To estimate blood flow information, artifacts such as background and hair region should be removed in preprocessing, and we only perform our estimation of blood flow on purified positions. For the sake of removing these artifacts, in our previous work, we proposed a simple method by using two thresholds on the subtracted amplitude and phase energy map.

Similar to Section 3.2, after performing PCA reduction, we obtain the denoised magnitude map $|F'(x, y)|$ by the following equation:

$$|F'(x,y)| = |F(x,y)| - \frac{1}{2|\Omega|} \sum_{c \in \Omega} |F(x,y,k' \pm c)| \tag{3.9}$$

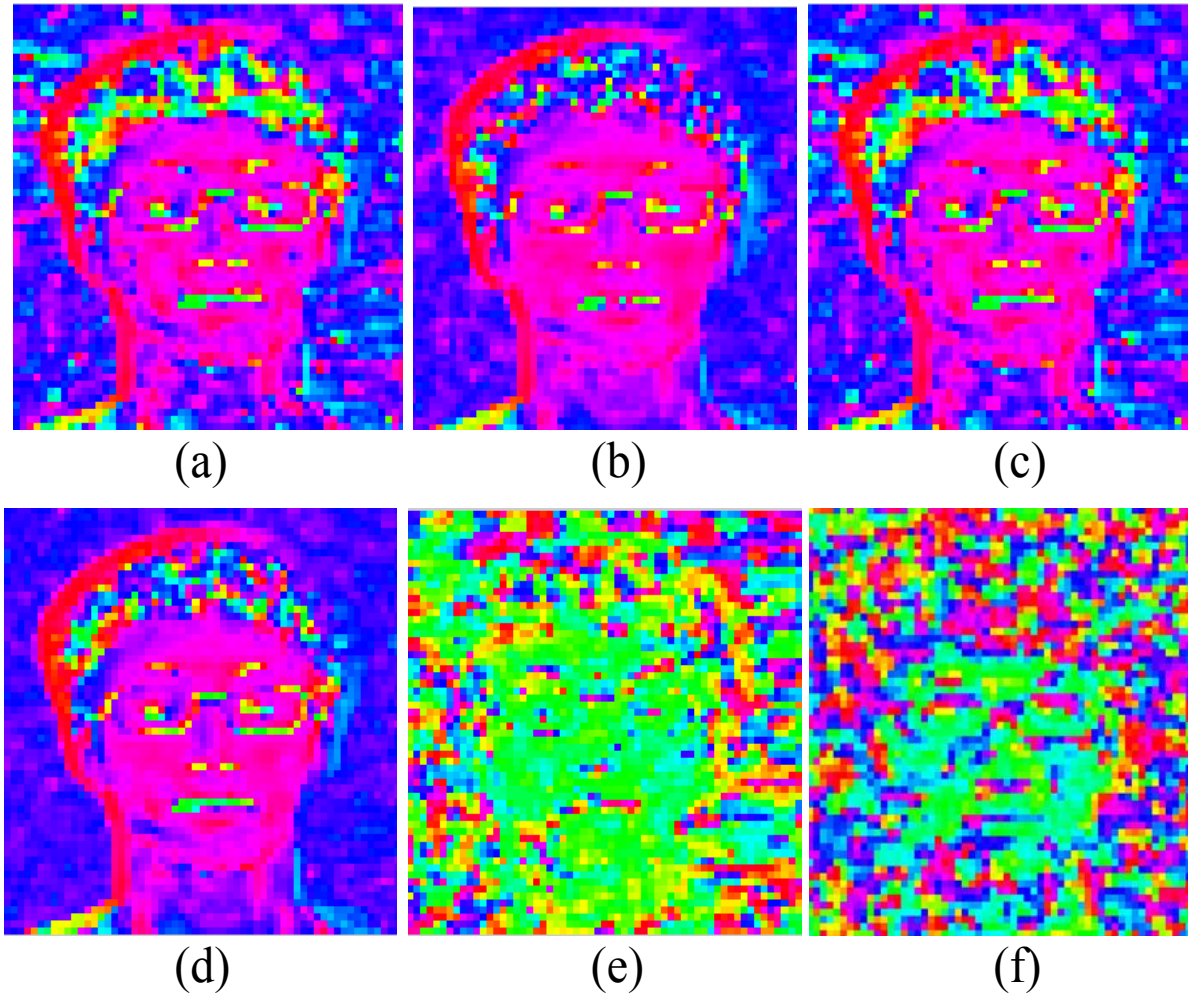

Figure 3.12: A comparison of phase mapping before and after PCA in HR frequency; the spatial processing level is 3.

For the sake of removing motion artifacts, only the locations where the magnitude is higher than a user-specified threshold τ_1 are considered during the estimation of the blood flow velocity.

Similarly, the phase energy map $E(x, y)$ are then defined by the energy function E as

$$E(x,y) = \sum_{i,j=-1}^{1} (\phi(x,y) \ominus \phi(x+i, y+j))^2 \tag{3.10}$$

Again, a user-defined threshold τ_2 is applied. Every location with an energy value that exceeds the threshold is labeled as noise and ignored during the blood flow estimation

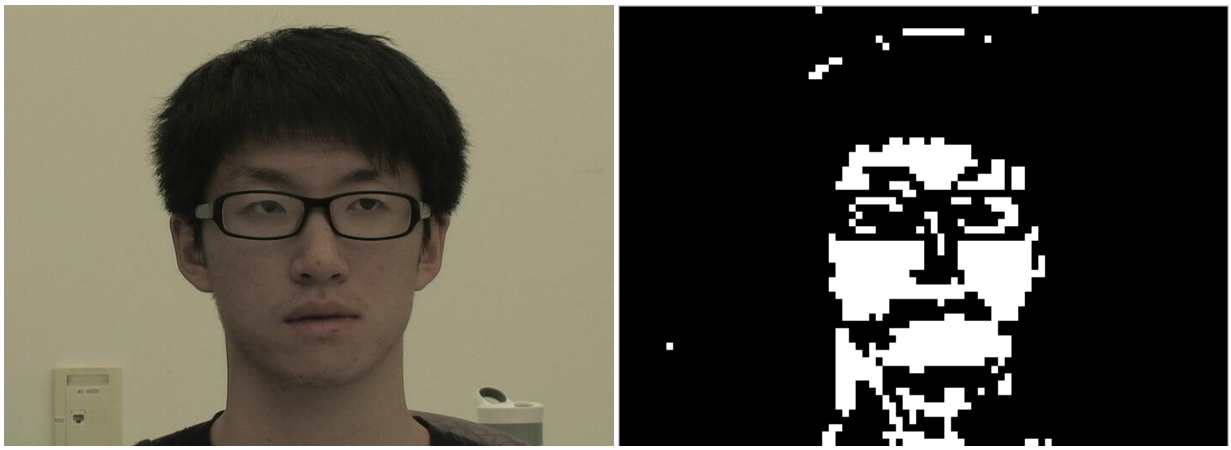

Figure 3.13: Binary mapping by using two thresholds in corrected amplitude and phase variance map.

process.

From the previous processing steps, we obtain a denoised magnitude map $|F'(x, y)|$ and the energy map $E(x, y)$ calculated from the phase map. In addition, we have two user-defined thresholds τ_1 and τ_2. Only pixels with a strong amplitude in the frequency band of the heart rate should be considered. This is expressed as $|F'(x, y)| > \tau_1$. Also, the phase should vary smoothly in an area around the pixels, the positions with energy $E(x, y)$ greater than the threshold will be labeled as noise. This condition can be written as $E(x, y) < \tau_2$.

Only locations that satisfy both conditions are kept in the next processing step, while all other locations are ignored. This process is illustrated as a binary mapping in Figure 3.13, where only white positions will be processed in further stages and black positions are labeled as noise and ignored.

3.4 Position segmentation by multi-kernel k-NN

In Section 3.3, noise is reduced significantly by applying subtraction and energy on the magnitude and phase map, respectively. The blood flow region is then segmented by thresholding, which is a popular tool and well-known technique in image segmentation[44][42].

However, in our case, the thresholding method requires the user to determine the value for global threshold τ_1 and τ_2 on the subtracted amplitude and phase variance map, which is inconvenient and sensitive to the value of selected threshold.

To improve our system, instead of using user-defined thresholds, we reduce noise and segment blood flow pixels by using a classifying or clustering technique. Our algorithm is based on the insight that, the properties of blood flow positions are very close because they are from the same mechanism. Equally, the most noise also possesses the similar attributes.

In this section, we present a simple algorithm for blood flow pixel extraction based on k-NN decision rule. The feature vector for each spatial location (x and y) is from input video. Because the learning model of k-NN classifying only depends on the value k, we utilized cross-validation to find the optimal value for k for our learning model for each video.

3.4.1 Feature selection

In both supervised and unsupervised learning, feature extraction and selection are important to improve learning performance[12][25]. In our case, We extract different features from the input video, which forms a feature vector for the training classifier in next step.

We used following five features:

1. **Amplitude in the HR frequency band** - This feature indicates the amplitude of the sinusoidal skin color variation at the frequency of the heart rate. The magnitude is strongest in areas of exposed skin and weaker in occluded areas and the background. The difference of value of magnitude is capable of distinguishing the blood flow pixels from background and other non-blood flow information areas.

2. **Amplitude around the HR frequency band** - As illustrated in Figures 3.4 and 3.5, the amplitude of movement artifacts in HR are close to blood flow artifacts, which are less distinct. However, this artifact is not HR related and equally strong in

all frequency bands. We averaged the amplitude at bandwidths around the HR and described it as one feature, which is able to identify movement artifacts from blood flow pixels.

3. **Phase in the HR frequency band** - The entries in the phase map are real-valued, periodic and limited to the interval of $(-\pi, \pi]$. Phase values are very close in blood flow positions, whereas it appears more random in non-skin pixels, which can be used to classify most noisy pixels into non-blood flow information clusters.

4. **Variance from the phase map in the HR frequency band** - As shown in Figure 3.6, phase varies smoothly at adjacent positions inside the facial region, whereas it appears more random in non-skin pixels. This feature can be used to purify blood flow pixels by removing artifacts whose phase value is close to real value.

5. **Spatial location** - The spatial location is useful for a variety of image processing problems, such as image segmentation[22][26][42] and image parsing[6][13][50]. Given the input video, spatial location ($\mathbf{x}$ and $\mathbf{y}$) can provide important information for its label. Instead of a universal classifier, we are able to learn a location-constrained classifier. Because each local position is learned by using the pixels in a local neighborhood system, the result is expected to fit local distribution better.

3.4.2 Learning from weak labels

Machine learning requires the input of labeled samples. However, it is sometimes very difficult and expensive to obtain such resources. Alternatively, in the process of learning, utilizing such weak but reasonable labels arise in different scenarios[51][48]. We divide the data into the following three groups:

1. **Blood flow positions** - In areas of blood flow positions, the magnitude is strong in the frequency of HR but relatively weaker in neighbor bands around the HR. Moreover, the phase variance on the blood flow position is much lower than the

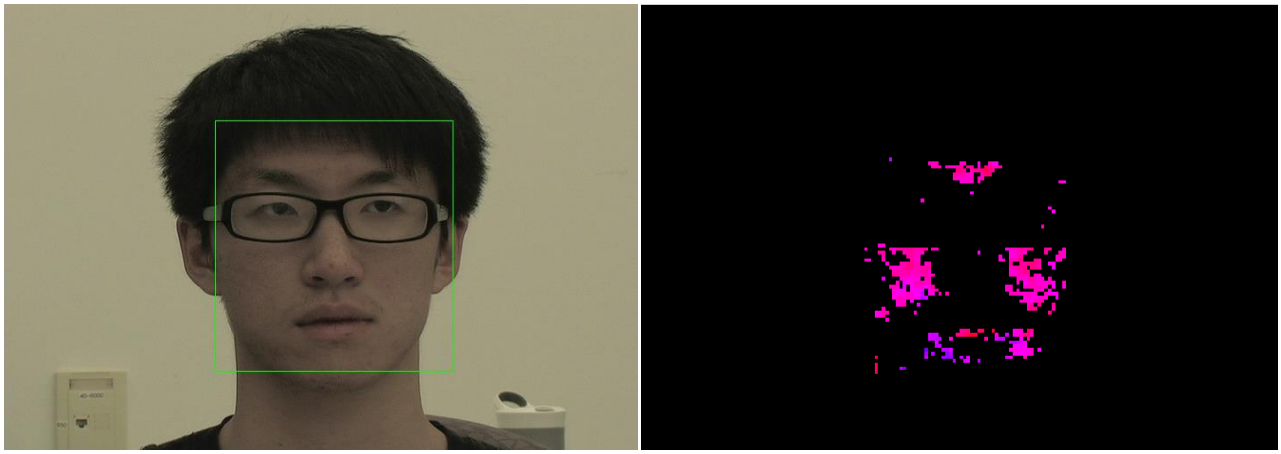

Figure 3.14: The training data of blood flow positions.

region of non-skin pixels. We select the training data of the blood flow positions as

$$S(x,y) = w_1 \times |F_S(x,y)| - w_2 \times \ |F_S^{''}(x,y)| - w_3 \times E_S(x,y) \tag{3.11}$$

where $|F_S(x,y)|$ and $|F_S^{''}(x,y)|$ are the scaled amplitude map in the HR frequency band and its averaged neighboring bands, and $E_S(x,y)$ is the scaled variance from the phase map in the HR frequency band. All the data are scaled into the range [0,1]. w_1, w_2 and w_3 represent the weights of the three maps, respectively, and are all set as 1 by default. After performing face detection, the final scores $S(x,y)$ are calculated inside the facial region and ordered in descending order; the first n scores are selected as training data of blood flow information. Figure 3.14 illustrates the selection of blood flow positions with the color white.

2. **Non-variation noise positions** - The magnitude is weaker on noise positions (e.g., hair, background) in both the HR frequency band and its averaged neighboring bands. And phase variances are higher due to the highly random phase variation. With the face-detection technique, the majority of positions outside the facial region are noise ones, as a result, we can get the noise training data by selecting random positions outside the facial region, which is illustrated in Figure 3.15.

3. **Motion artifact positions** - The property of motion artifacts is very unique. Due to involuntary movements, the motion artifacts appear equally in all frequency bands in-

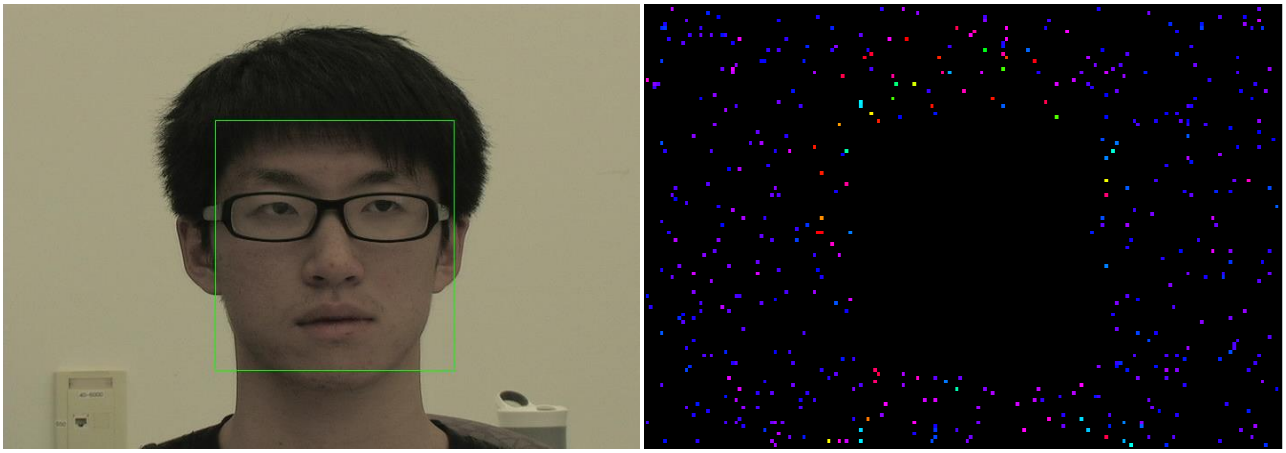

Figure 3.15: The training data of non-variation positions.

stead of in the dominant one. Therefore, we select n positions with the first strongest magnitude from the amplitude map in averaged HR neighboring bands as training data. This selection is shown in Figure 3.16.

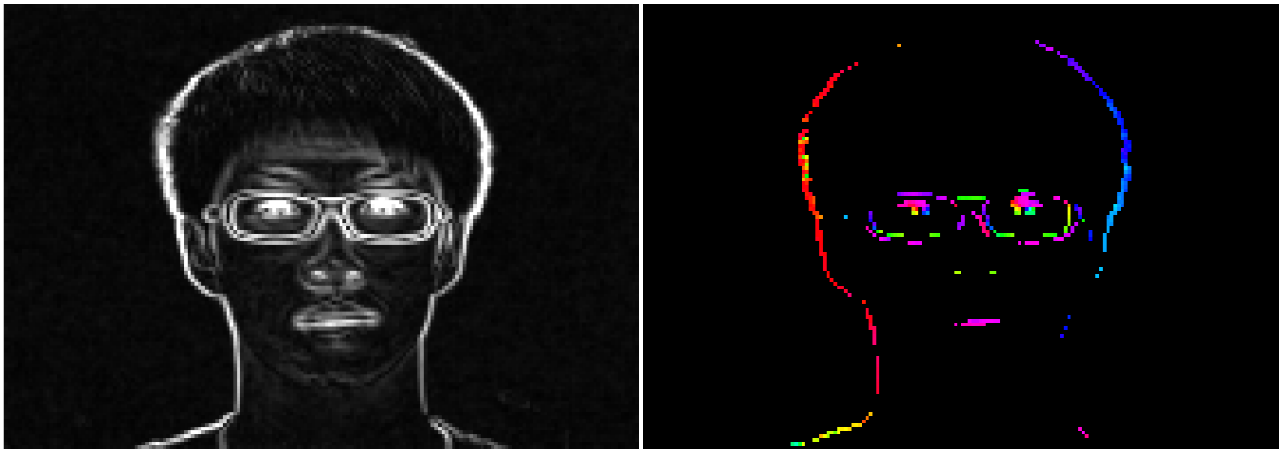

Figure 3.16: The training data of motion artifact positions.

3.4.3 Multiple kernel-based k-NN classifiers for blood flow position segmentation

The kernel learning methods[19][41][45] have been gradually becoming important tasks in the area of pattern recognition and machine learning. The kernel method is known best

by its member support vector machine (SVM). In recent years, a variety of methods have been proposed to utilize multiple kernels instead of a single one[24]. In our case, the final segmentation is performed by using multiple kernel-based k-NN algorithm, which improves the classification accuracy significantly.

The object of the kernel is to map data from the original space into the feature space. As single kernel on two samples x and x', represented as feature vectors from some input space, is defined as $K(x, x')$. Multi-kernel, however, refers to the method that uses a predefined set of kernels and learns an optimal combination of kernels. This combination can be either linear or non-linear. The basic approach of multi-kernel is to map data from the original feature space to the Hilbert space; because of the additive property, the combination function is still a kernel. In our case, we use a supervised learning of a linear combination of N kernels, which is defined as:

$$K'(x, z) = \sum_{i=1}^{N} \beta_i K_i(x, z) \tag{3.12}$$

where β_i is the coefficient for each kernel with $\beta_i \geq 0$, and $\sum_{i=1}^{N} \beta_i = 1$, x and z are the training and test in the original feature space.

For each feature, the distance measurement with each kernel between the training sample in x and the test sample z is defined as:

$$K_i(x, z) = exp(\gamma_i ||x_i - z_i||_1) \tag{3.13}$$

where $||x_i - z_i||_1$ is the L_1 norm, the simply absolute value between training data x and test data z for feature i. γ_i is a free parameter, which is used to smooth and scale the data in the group. Because the value of each kernel is scaled into the range from 0 (in the limit) to 1 (when $x_i = z_i$), it has the explanation as the similarity measure. One thing to note is that because of the periodicity of the phase, post-processing should be performed after getting the norm 1 on the feature of the phase value. We perform cosine distance here, which is used for the complement in positive space as:

$$d_p(x, z) = 1 - cos(||x_p - z_p||_1) \tag{3.14}$$

where d_p refers to the cosine distance with the feature of the real phase value between the training data and test data. For multi-kernel k-NN, the kernel distance measurement in the feature space between training samples and test samples, is thus defined as:

$$d'(x, z) = \sum_{i=1}^{N} \beta_i \times exp(\gamma_i d_i(x, z)) \tag{3.15}$$

where $d(x_i, z_i)$ represents cosine distance and norm 1 for the feature of the phase value and other features, respectively. β_i is set as $\frac{1}{N}$ by default and equal to $\frac{1}{6}$ in our case.

We now have extended classical k-NN by incorporating multi-kernel. The proposed algorithm for classification is designed to find the similarity between each test data and training data of each cluster by using multi-kernel. With the given value k, the algorithm is applied on the extracted feature space, and the class of each test data is then determined by the majority voting scheme.

3.4.4 Cross-validation for choosing optimal k

In the k-NN case, the training set is constrained to a certain number of data for each class, and we need to find a value for k that is capable of minimizing prediction error. As a result, we need to measure the number of prediction errors by defining a loss function and counter. That is, we take the truth as input, make the prediction and the counter returns a value that is 0 when it perfectly matches and large when the prediction is far from the truth.

We represent our data as pairs $(x^{(i)}, y^{(i)})$ for $i = 1 \cdots n$, where $x^{(i)}$ represents the i^{th} data, and $y^{(i)}$ is the label of class. When new data comes, we want to predict the class this data belongs to. For this coming data, we classify it into one class if the majority of its k-NNs among $x^{(i)}$ are belong to this class. We used a zero-one loss function L, which is defined as

$$L(\hat{y}, y) = I(\hat{y} \neq y) \tag{3.16}$$

where I is the indicator notation, and this zero-one loss function returns 0 if the prediction is correct and 1 if it is correct. For each k, the result of loss functions are then added up and assigned to the error counter.

We first split the original training set into two parts, a training subset $T(80\%)$ and a validation subset $V(\ 20\%)$. The learning models are then built from the training subset, and the loss error number is measured by using the validation subset. To find the optimal k^* for our learning model, we choose the argument of the minimum value as

$$k^* = \arg \min_{1 \leq k \leq K} err(k) \tag{3.17}$$

where $err(k)$ is the counter of the error classified number. The whole progress of cross-validation is illustrated here:

Input: The paramater k in k-NN, the number of cross validation set n.

for $k = 1, 2, ..., K$ **do**
    $err(k) = 0$;
    for $i = 1, 2, , ..., n$ **do**
        Predict the class label $\hat{y}_i$ for data validation set x_i ;
        $err(k) = err(k) + 1$;
    end
end

Output: $k^* = \arg\min_{1 \leq k \leq K} err(k)$

Return k^*;

Algorithm 1: Cross validation in k-NN.

3.5 Estimation and visualization

From this final section, we obtain the spatial positions, which are classified into blood flow, non-variation and motion artifacts clusters. These three labels can be visualized by mapping into the Red, Green and Blue channels as maximum, respectively. To visualize only blood flow positions with phase information, we also map phase values into the periodic hue component of the HSV color space while keeping saturation and blood flow position's value at maximum with non-variation and motion artifact values at zero.

The remaining pixels are locations in the face that exhibit blood flow. The phase shift between these pixels gives us information about when this pixel changes its color due to the blood flowing through it relative to its neighbors. The difference between two neighboring phase values (i.e., the phase shift) thus directly corresponds to the velocity of the blood flow at this point. We compute this difference in the horizontal and vertical directions by applying the 2D Sobel operator (Figure 3.17).

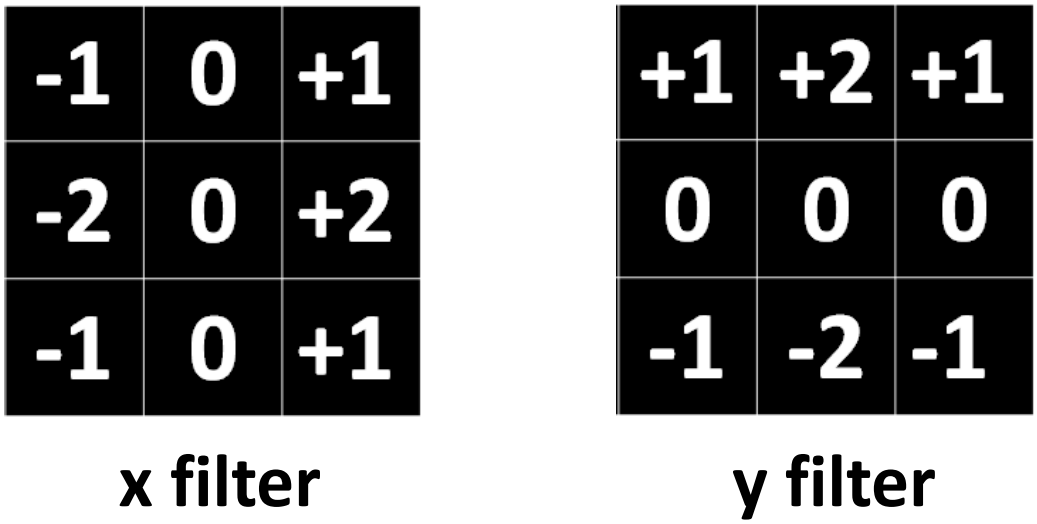

Figure 3.17: Sobel Filter in horizontal and vertical direction.

Special care must be taken to consider the periodicity of the phase when implementing Sobel filtering in this case. The result is the gradient of $\phi(x, y)$, which corresponds to the desired 2D velocity field of the blood flow. Only locations that were labeled as blood flow positions are calculated while others were discarded due to the invalid property.

Chapter 4

Experiment Results

The goal is to compute an approximated two-dimensional blood flow field from an image sequence. Without doubt, a fundamental problem is to measure the accuracy of the estimated velocity field. We evaluated our algorithm on real face videos from volunteers as well as on synthetic image sequences for which the 2D blood flow fields were known in advance. In both cases, we chose sequences that are close to the Eulerian video magnification scenario, which are not severely affected from environment.

The real videos allow a qualitative and the synthetic videos a quantitative evaluation, respectively. For the synthetic videos, we report the results of two different noise-reduction techniques and different levels of spatial low-pass filtering and downsampling. Our comparisons are concentrated on the accuracy on both magnitude/angular error, and density of the velocity field. All of our synthetic and real videos, as well as our evaluation results, are available on our website.

4.1 Real videos

For the real videos, to illustrate that our technique works on more than just the face, the results will be shown on hands of the subjects as well. Results with different durations will be also illustrated. We can only show the segmented positions and computed velocity fields, discuss the qualitative properties and leave the reader to judge.

Figure 4.1: Experimental setup.

4.1.1 Experimental materials and setup

A simple digital cameras were used to capture videos for analysis. For the experiments, all videos were recorded in 24-bit RGB color (3 channels $\times$ 8 bits/channel) at 30 frames per second (fps) with a pixel resolution of 720 $\times$ 480.

As illustrated in Figure 4.1, the experiments were taken indoor and volunteers were asked to sit down in front of the camera at a distance of 2-3 meters. For this research, 10 face videos of 6 volunteers with an approximate duration of 10 seconds each were collected. The videos were taken from people of both genders, with different ages and varying skin colors (Asians, Africans and Caucasians).

A wireless Bluetooth heart rate sensor (HxM BT) was used to measure the volunteer's

heart rate for all experiments. Using a mobile phone or laptop, the sensor provides real-time heart pulse feedback. This measurement was synchronized with the video recording. The experiments were conducted indoors with fluorescent light overhead as the source of illumination. During the video recording, after setting the camera into movie mode, participants were asked to sit down and keep their head as still as possible. Their faces were recorded with a camera on a tripod at a distance of 2 m from the subjects.

Using MATLAB or the OpenCV Library, 8-bit pixel values (0-255) in red (R), green (G) and blue (B) channels were read from the frame sequence, which provided a set of raw 3D color signal $S(x, y, t)$, where x and y are horizontal and vertical positions, respectively, and t is the index of the frame number corresponding to the video's sampling rate.

4.1.2 Amplitude and phase map

Figure 4.2 shows the pulse amplitude and phase map for the green channel, extracted from a 10s (335 frames) video of a volunteer. The blood flow information is represented here. The positions in areas of blood flow possess much stronger magnitudes than non-blood flow positions(e.g.,occluded areas and background). The motion artifacts are also illustrated in both Figure 4.2 (a) and (b).

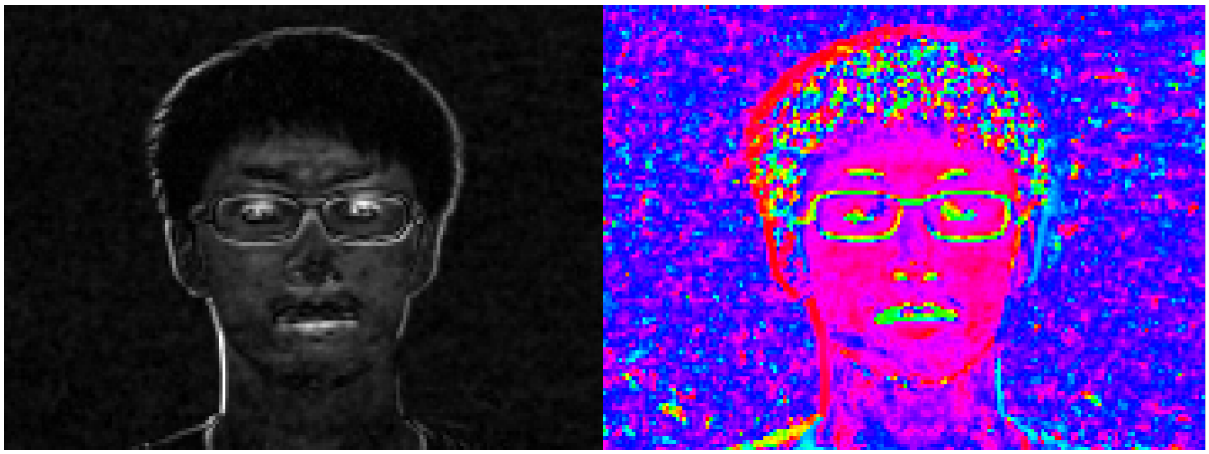

Figure 4.2: Magnitude and phase in the green channel with the original number of frames.

For heart rate detection or blood flow estimation, the duration when capturing video is always a problem. In [53], duration varied from 30 seconds to several minutes. Given

the heart rate, Figure 4.3 and Figure 4.4 illustrate the amplitude and phase map for the green channel from the same video but with only the first 200 and 100 frames, respectively. Compared with pulse mapping from the video with the original time, the reduction of time results in much weaker magnitudes and more random phase variation in exposed skin areas. It indicates that the reduction of video duration reduces the intensity of the interest signal (here, blood flow). On the contrary, the target signal can be revealed sufficiently with a longer duration. However, the problem of longer duration is that subjects cannot keep their head still or minimize the movement according to the increasing time. Therefore, the choice of duration is a trade-off between a stronger signal of interest and minimizing movement.

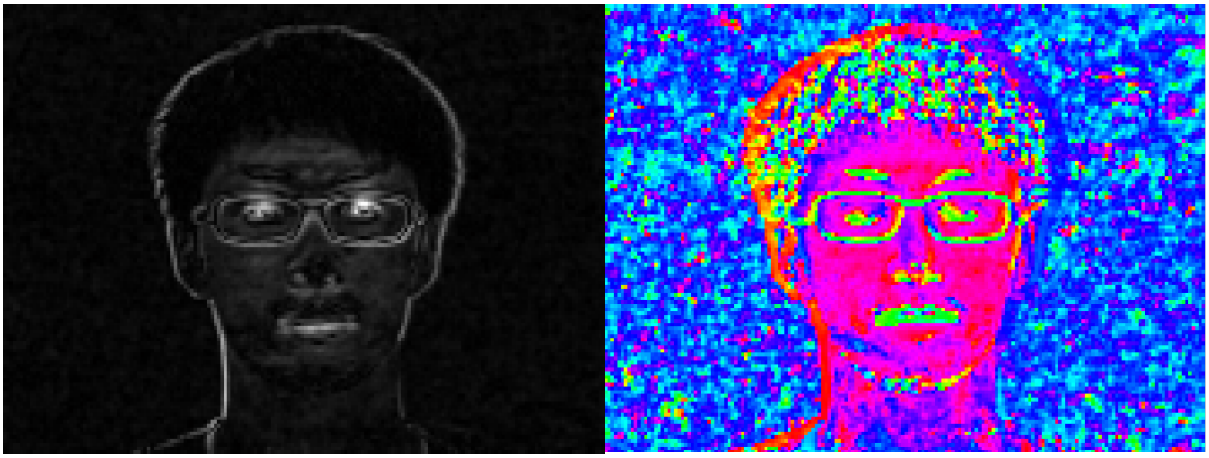

Figure 4.3: Magnitude and phase in green channel with the first 200 frames.

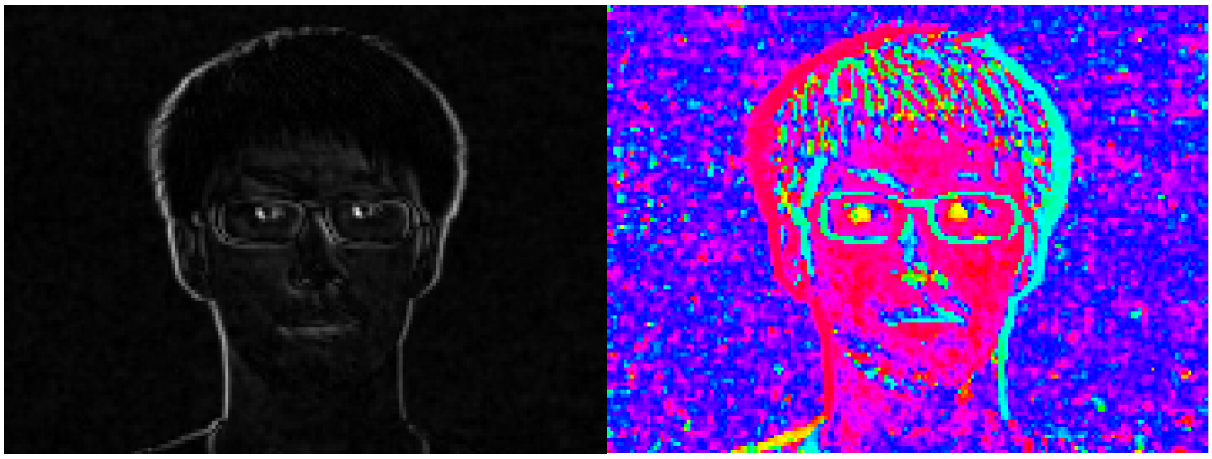

Figure 4.4: Magnitude and phase in green channel with the first 100 frames.

Another interesting discussion is the blood flow region. Our method so far has focused on the facial region due to abundant blood vessels under the skin. However, our method also works on the hand. Figure 4.5 illustrates the amplitude and phase map on both the face and hand regions. Compared with the back of the hand, the palm region reveals much more blood flow information. It can be also seen that, for blood flow positions, the real phase values on the face and the hand are very close, which again demonstrates the smooth flow of blood across the entire human body.

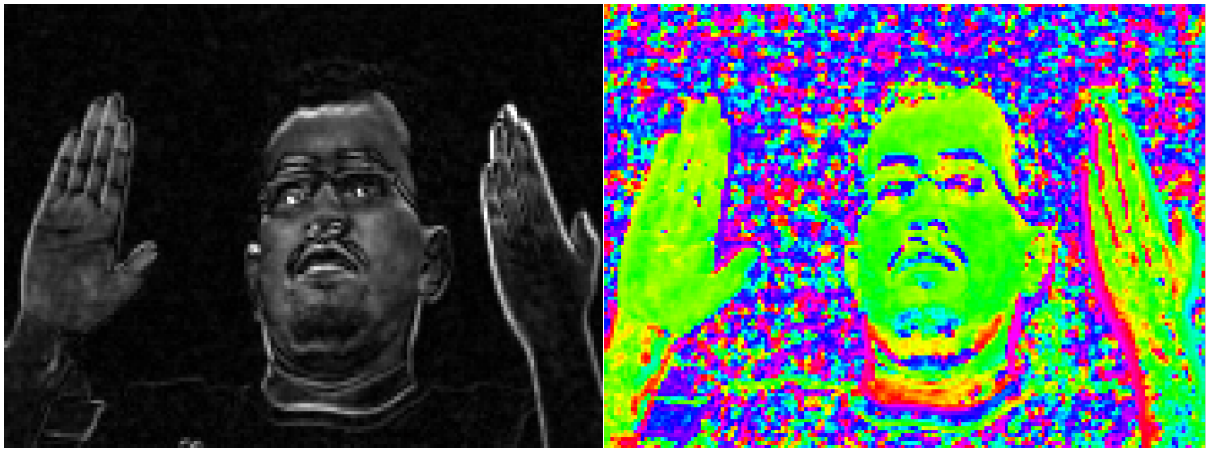

Figure 4.5: Magnitude and phase of both face and hand positions in green channel.

For dimensionality reduction, we can select the exact same training data from the facial region after face detection and then perform PCA for extracting the first component. The majority of blood flow variation will fall into the main axis, which contains the strongest signal of interest. For the bare hand, the hand detection algorithm can be applied first, and we can then use the same method as the facial region for choosing PCA training data as well as achieving dimensionality reduction.

4.1.3 Segmentation

We first present the experimental results of the segmentation on real videos. For all of our tests, we show segmentations by our algorithm from the second level to the fourth level. We chose this range of Gaussian smooth and downsampling experimentally by noting the increased signal-to-noise ratio (SNR) and computational efficiency.

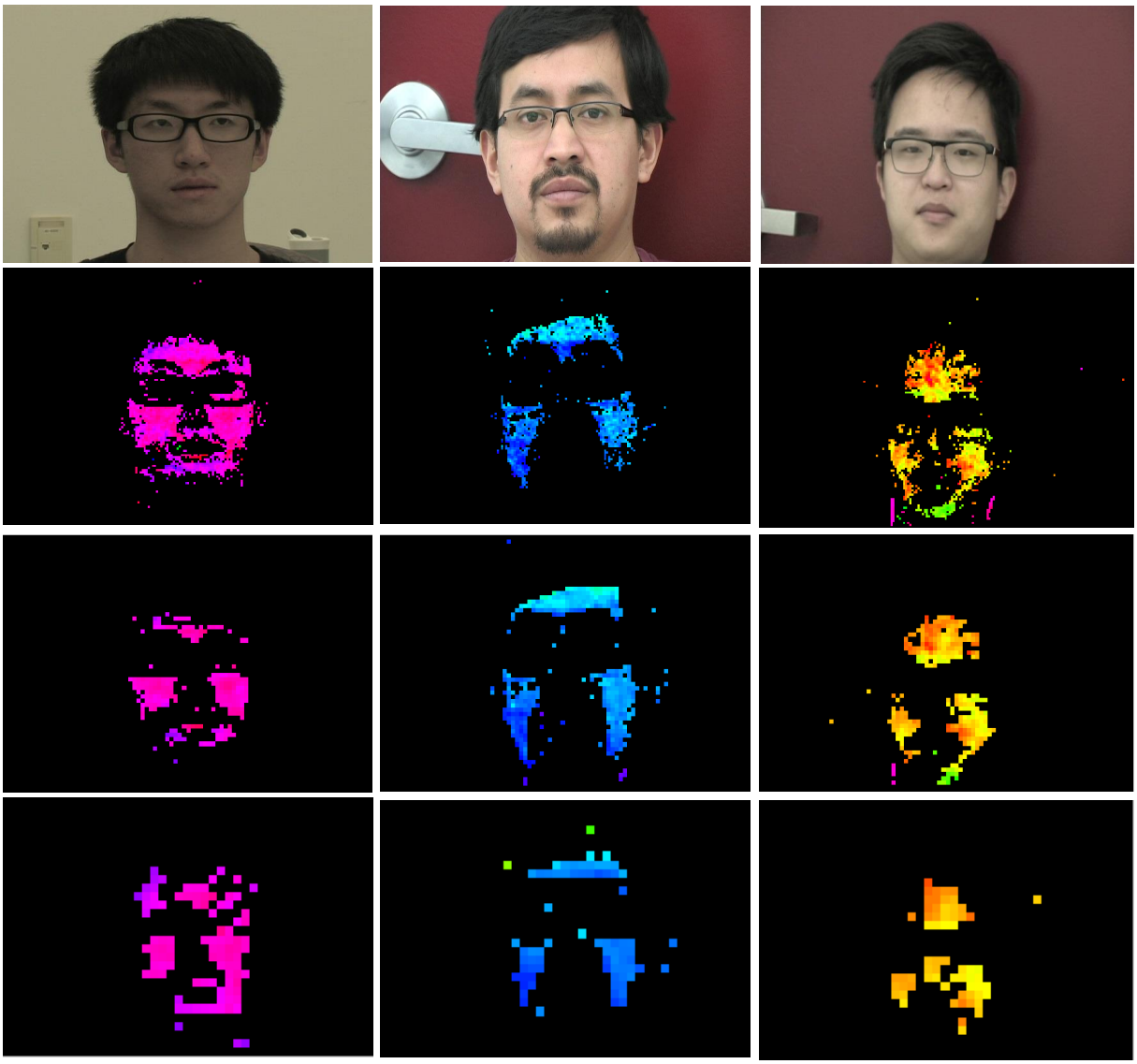

Figure 4.6: Segmentation results for blood flow positions; k is chosen by cross-validation. From top to bottom: There are original frames from videos with segmentations produced at level 2, 3 and 4. From left to right, the three volunteers are in the same spatial size of 720×480.

Figure 4.6 shows the results for different subjects. Segmentations on different spatial processing levels are shown for comparison purposes. As we discussed in the previous chapter, the k-NN classification requires the user-defined constant k. Cross-validation are applied for each test video on each level for choosing this optimal parameter.

From the results shown here, all the results are visualized by mapping the phase com-

ponent to the periodic hue component of the HSV color space while keeping saturation at a maximum. In the meantime, the segmentation (classification) results are also mapped into the value component by setting blood flow positions at the maximum while keeping others at the minimum. The original frames from the videos, segmentations produced with level 2, 3 and 4, are shown from top to bottom. From left to right, the three volunteers are illustrated in the same resolution. It is also easy to understand that increasing the resolution leads to a decrease in computational efficiency.

As can be seen, some areas in the background were not properly marked as noise. One possibility for removing these remaining artifacts in the background is to reuse facial location we obtained in preprocessing to locate the face in the video and to ignore all other regions.

4.1.4 Computed velocity fields

The results from the real face videos allow a qualitative evaluation of the computed velocity fields. After the segmentation in the preceding step, positions with motion artifact and non-variation are labeled as noise and ignored. Finally, Figure 4.7 shows the velocity of the remaining areas of interest in saturated red. Different spatial resolution from three different real videos are illustrated as well.

With real image sequences, it is almost impossible to see differences among the different levels because errors of 10% or 20% are not easily discerned at different resolutions. Moreover, other errors are not always controllable, such as the computed velocity mistaken from segmentation.

4.2 Synthetic videos

The real videos in the previous section offer qualitative evidence that our segmentation algorithm performs well at different spatial processing levels. Following the visual comparison, the quantitative study of segmentation quality on synthetic video will be presented

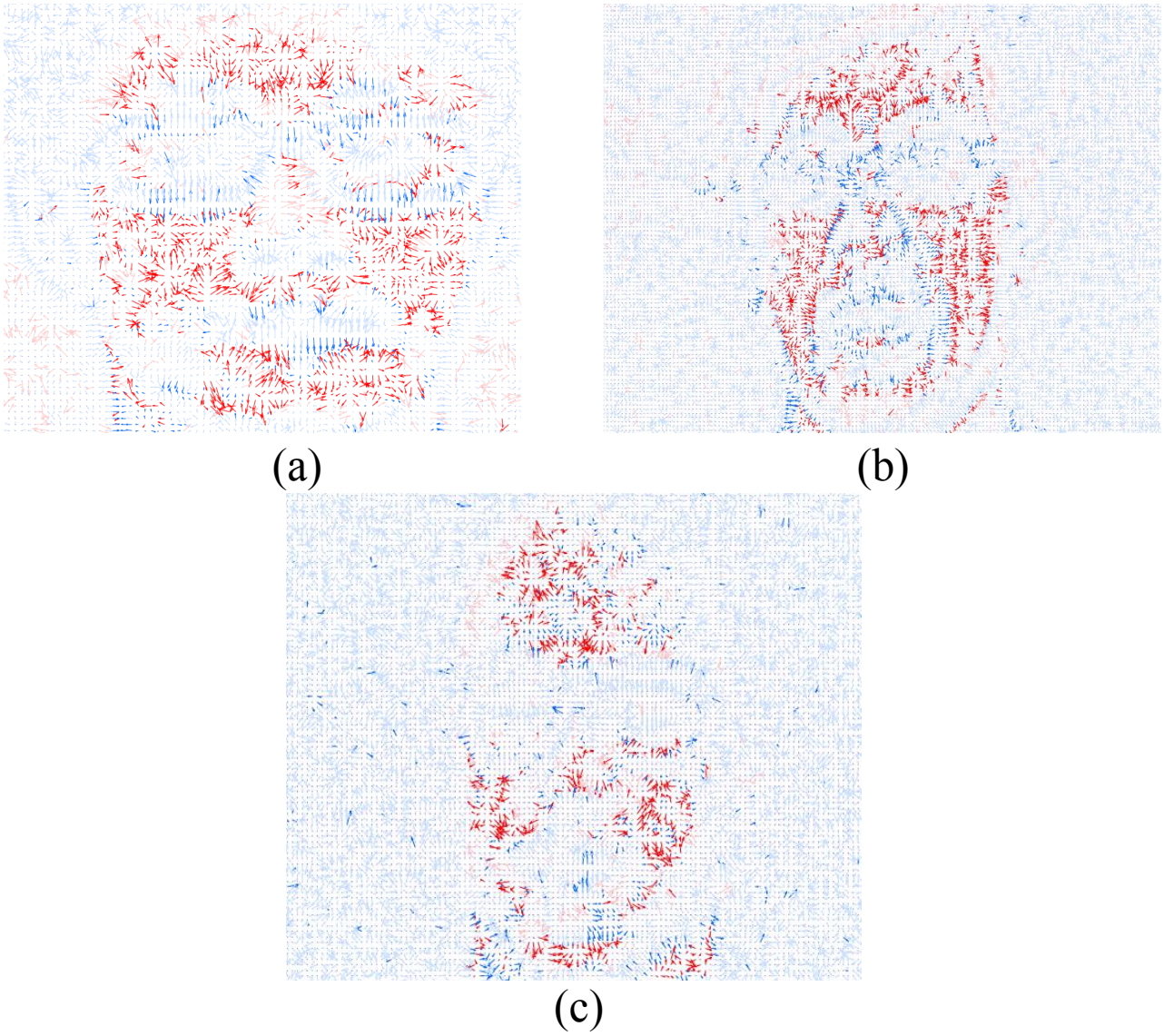

Figure 4.7: Blood flow velocity fields from three different real face videos with different spatial resolutions.

here. In reporting the performance of our blood flow estimation method applied to the synthetic sequences, for which 2D motion fields are known, we concentrate on the segmentation (recall and precision) and error statistics on the flow vector (magnitude and angular).

4.2.1 Synthesizing image sequence

To quantitatively evaluate the accuracy of the blood flow detected by our method, we created synthetic face videos with known velocity fields. The main advantages of synthetic inputs are that the 2-D blood flow field properties are controllable and can be tested in a methodical way. To be more specific, we know the input of the flow field and can thus evaluate each performance corresponding to different techniques and parameters. Therefore, we must to make sure the input signals are clean and cannot be affected or destroyed. Our synthetic videos include the following:

- **Sinusoidal blood flow:** The synthetic videos contain a central region that represents the face in the real video. Heart rate and blood flow velocity are chosen to match the real values as closely as possible. Ideally, the sinusoidal input for each spatial location is represented as the sinusoidal waves:

$$y = Asin(2\pi f + \phi) \tag{4.1}$$

  where A is the amplitude of the sinusoidal signal on the central region, and f and ϕ represents the frequency and phase shift of the signal on the specific location, which are also heart rate frequency and blood flow, respectively. Figure 4.8 shows the phase shift from the bottom right to the upper left.

  The results illustrated above are based on a temporal length of 300 points (frames) with an orientations of 135° and a range of $[0, 2\pi]$. Although the resulting blood flow pattern translates with an unreasonably fast velocity, we can just shrink the range to make the flow velocity realistic.

  To show color variation that simulates the blood flow in the central region, a mathematical function is fit to the magnitude and phase spectrum of the color variation of face pixels in a real video. The fitted function plus Gaussian distributed noise are used to model the synthetic color variation in the frequency domain. Afterwards, the synthesized signal is transformed back into the time domain to create the video. Figure 4.9 shows the fitting signal of real blood flow.

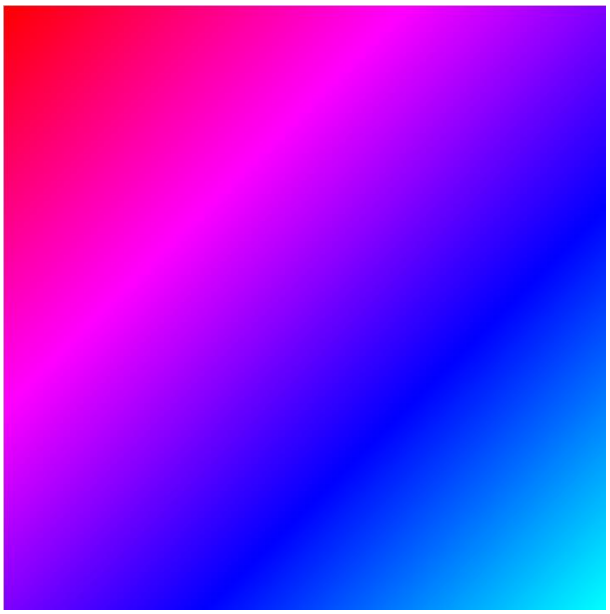

Figure 4.8: Phase mapping in the region of interest; the phase varies from 0 to 2π.

- **Noisy background:** The noisy background is also included in the synthetic video. The assumption is that there is no such information in heart rate frequencies, therefore the amplitudes and phase values of these positions are low and completely random, respectively. Similar to sinusoidal blood flow, we synthesize the noisy background by only generating Gaussian-distributed noise in the frequency domain and then transforming it back to the time domain. It is important to note that no such user-specified sinusoidal wave occurs in the frequency of pulse.

  We see that compared with an central region, the phase values of background pixels are completely random, which means, for each position, there is no such relationship between it and its neighbors.

- **Motion artifacts:** As we discussed in Chapter 4, the artifacts are strong at the inside edges and around the face due to involuntary movements of the subject's head. Under the assumption that motion artifacts induce white noise, which is not dominant in this specific frequency band, the artificial motion should be random, and the amplitude around motion positions needs to be equally strong in all frequency bands.

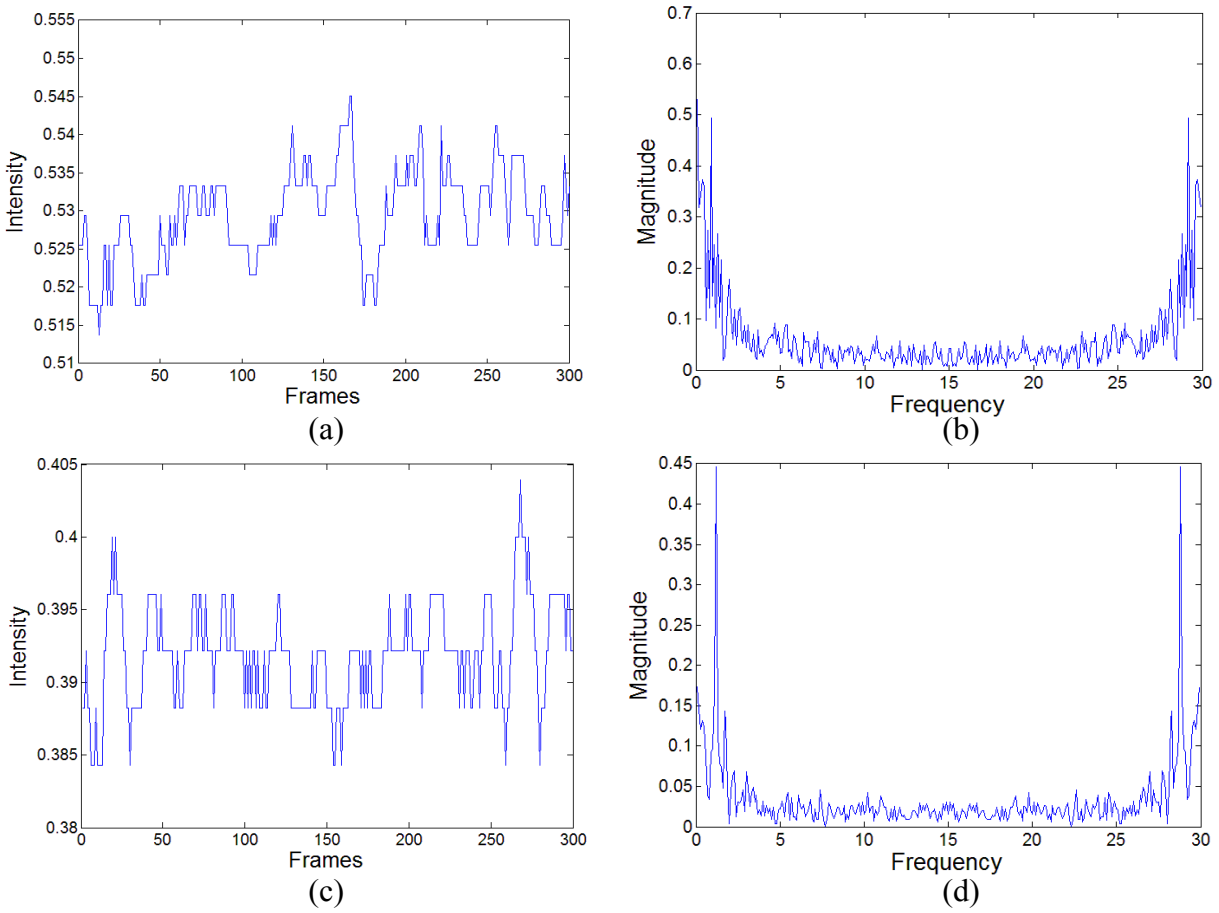

Figure 4.9: The comparison between real and synthetic blood flow signal in time and frequency domains.

To fit this sort of noise, we add moving edges around the central region and several artificial objects (e.g., glasses, mouth, etc.) with slight motion inside the region. The motion artifacts we created can be explained in amplitude and phase map, which are around and in the interest frequency band (heart rate), respectively. These two illustrations are shown in Figure 4.11.

The synthetic videos were created by the authors. OpenCV with the C++ interface was used for implementation. The introduction and documentation of OpenCV can be found in [2] and [1]. As illustrated in Figure 4.12, we used an artificial sequence that simulates translational blood flow on the region of interest (square region). The blood flow moves with orientations $-45°$ and speeds of $\frac{\pi}{n}$ and $\frac{\pi}{n}$ pixel/pixel, respectively.

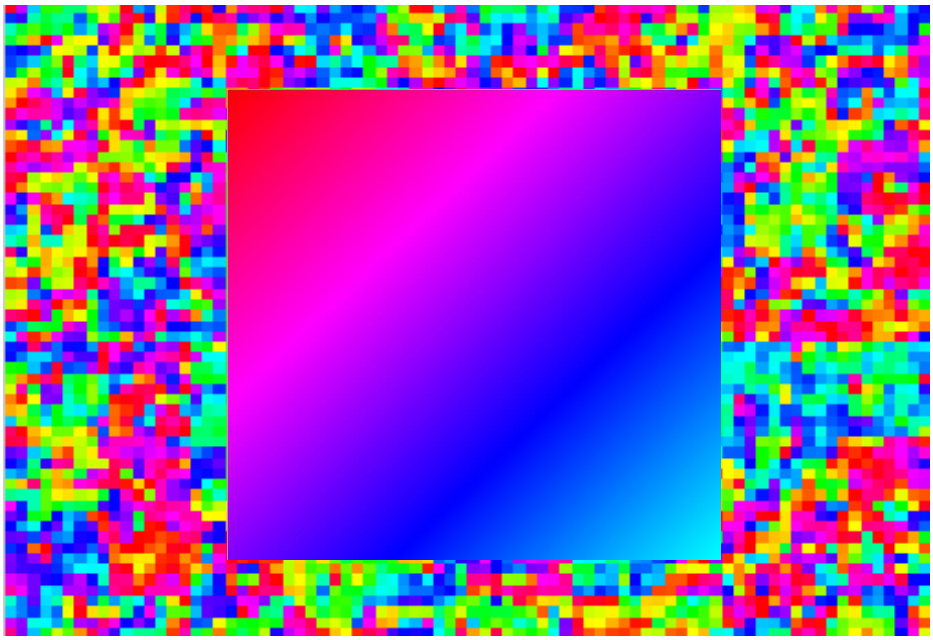

Figure 4.10: The illustration of pulse signal in the noise background.

4.2.2 Error measurement

To evaluate our algorithm on synthetic videos, we proposed two methods to measure the segmentation and velocity, which are known in advance. In the segmentation problem, we use precision as the measurement of the quality of the blood flow positions obtained, whereas recall is used to measure its quantity. For velocity evaluation, we apply two different error metrics, average angular error (AAE) and average magnitude error (AME), to compare the velocity field to the ground truth on every blood flow location in the video.

Precision and recall of segmentation

In Section 3.4, to classify all blood flow and noise information, we performed k-NN classification based on multiple kernels. After obtaining the blood flow pixels, the following nagging question remained: "Have we found the most of them or are we missing some important positions?" Obviously, we hope there is no noise in the blood flow cluster. Un-

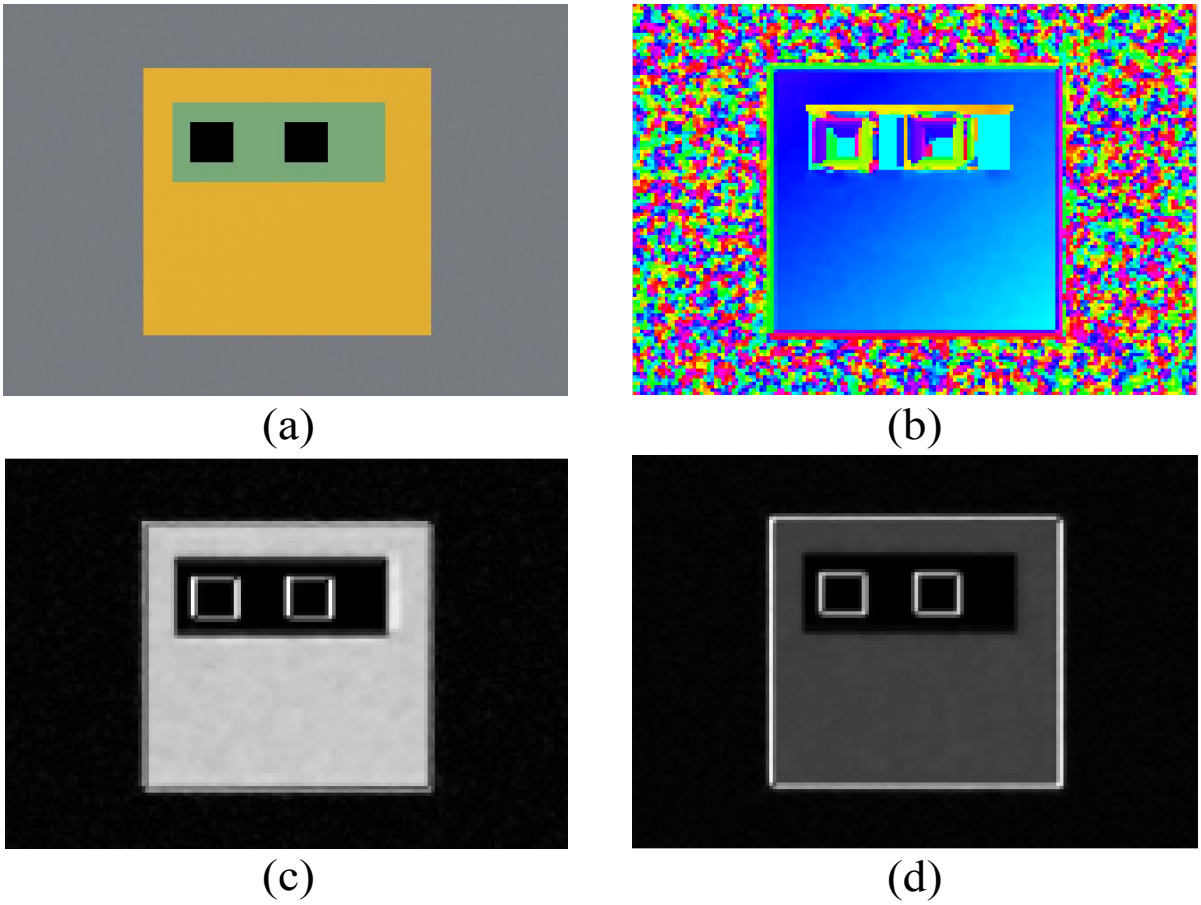

Figure 4.11: One frame from a synthetic video and the corresponding 2D motion fields.

fortunately, getting "everything" while avoiding "noise" is almost impossible. However, it is possible to measure how well our classifier performed by using precision and recall.

In pattern recognition, information retrievals and search engines, the basic measurements used for evaluating a classification model are precision, recall and accuracy. Precision is the fraction of all retrieved instances that are relevant, while recall is the fraction of all relevant instances that are retrieved. Accuracy, the proportion of true instances among the total number of instances, is also used as a statistical measure. However, as the result of the bias problem, we leave it out and only consider precision and recall here.

According to the cluster property, our target is to obtain blood flow positions. As a result, we call the blood flow positions as "positive class" while noise and halo artifacts as "negative class".

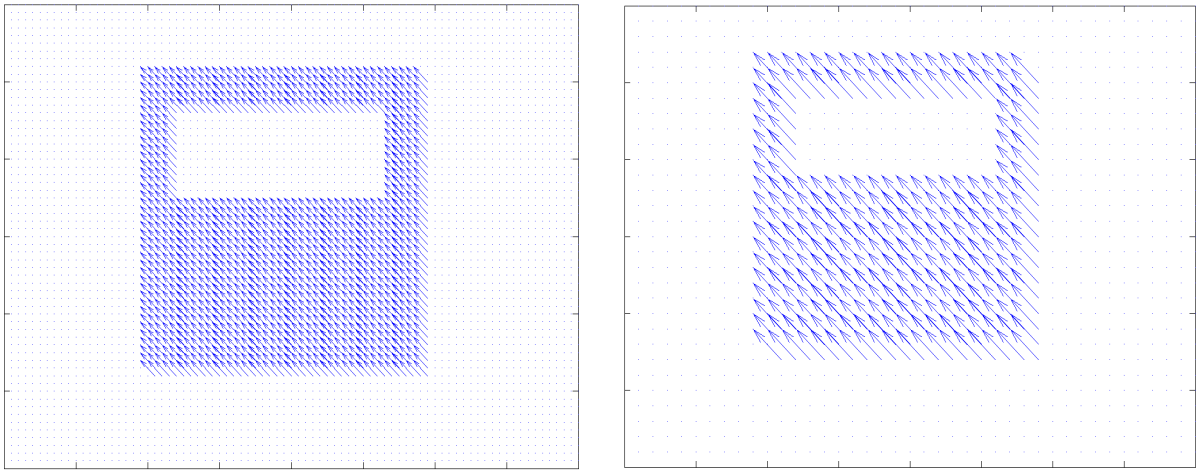

Figure 4.12: Ground truth of 2D motion fields in level 3 and 4 from the same synthetic video.

	Relevant	Non-Relevant
Retrieved	True Positives (TP)	False Positives (FP)
Not Retrieved	False Negatives (FN)	True Negatives (TN)

Table 4.1: Definition of recall and precision.

According to the table, given true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), we can get the precision and recall easily by using the following equation:

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

In our case, true positives (TP) and false negatives (FN) are the relevant blood flow positions that are retrieved and not retrieved, respectively. False positives (FP) and true negatives (TN), however, are irrelevant positions (e.g., noise and motion artifacts) that are retrieved and not retrieved. This measurement is shown in Figure 4.14.

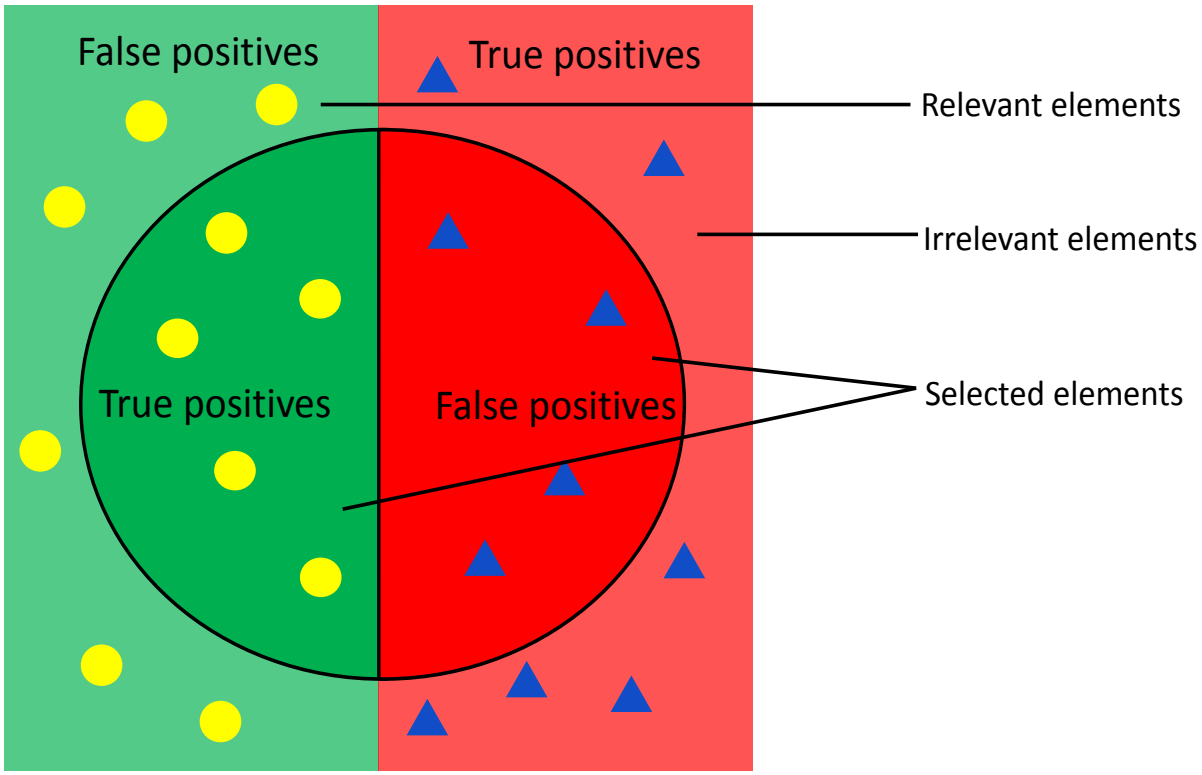

Figure 4.13: The illustration of precision and recall.

The result varies according to the amount of downsampling. To consider both the precision and recall, the F score F_β can be calculated to measure the classifier's accuracy with

$$F_\beta = (1+\beta^2)\cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{4.4}$$

where β is a non-negative parameter. We are using balanced $Fscore(F_1score)$ where β is equal to 1 in our case.

Gradient on velocity field

Error behavior can be reported with many methods. For the synthetic image sequences, we apply two different error metrics to compare the velocity field produced by our algorithm to the ground truth: average angular error (AAE) and the average magnitude error (AME).

For every non-noise location in the video, our algorithm estimates a 2D velocity vector $\overrightarrow{v_e} = (u_e, v_e)$. For the same location, the true velocity $\overrightarrow{v_t} = (u_t, v_t)$ is available from the synthetic video. The angular error AE between these two vectors is calculated as:

$$AE(\overrightarrow{v_e}, \overrightarrow{v_t}) = \arccos\left(\frac{\vec{v_e} \cdot \vec{v_t}}{\|\vec{v_t}\| \cdot \|\vec{v_e}\|}\right) \tag{4.5}$$

Close to [10] and [8], the goal of the AAE is simply to provide a relative measurement. However, instead of adding 1 to the denominator (avoiding the divide by zero problem for zero flows), we create a non-zero flow field as ground truth and calculate the absolute value of AAE.

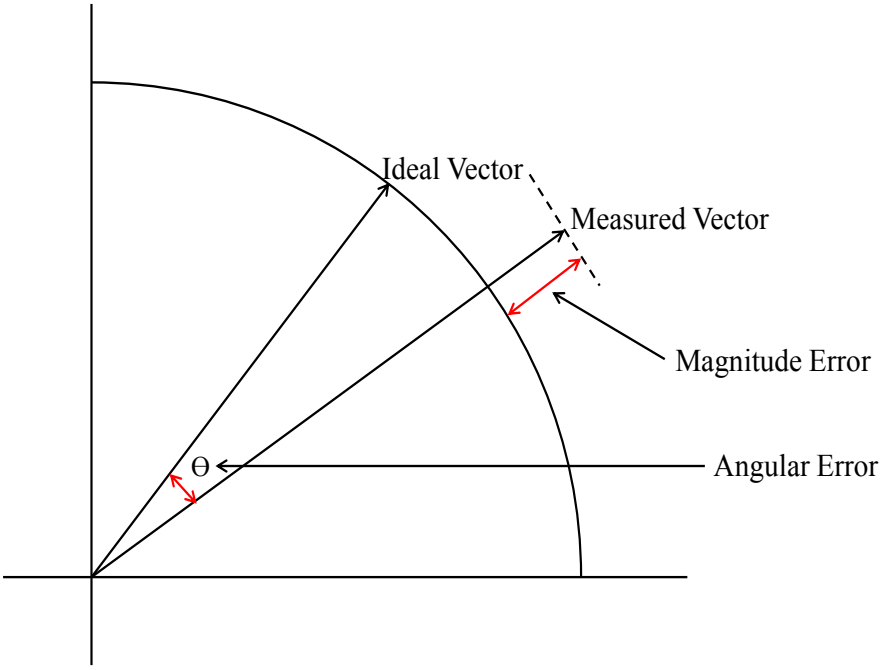

Figure 4.14: The illustration of AAE and AME error measurement.

Similarly, the AME is then computed as the average of the magnitude errors over all non-noise locations. The magnitude error ME is defined as:

$$ME(\overrightarrow{v_e}, \overrightarrow{v_t}) = \frac{\|\vec{v_e}\| - \|\vec{v_t}\|}{\|\vec{v_t}\|} \tag{4.6}$$

Note that this is a percentage relative to the length of the ground truth vector, whereas

the angular error is given in degrees. The average magnitude error is again obtained by averaging ME over all non-noise locations.

4.2.3 Precision and recall of segmentation

We began by comparing the segmentation result and ground truth, which are visualized the same as the results from the real videos in HSV color space(Figure 4.15). The second level of spatial processing was selected here for better resolution.

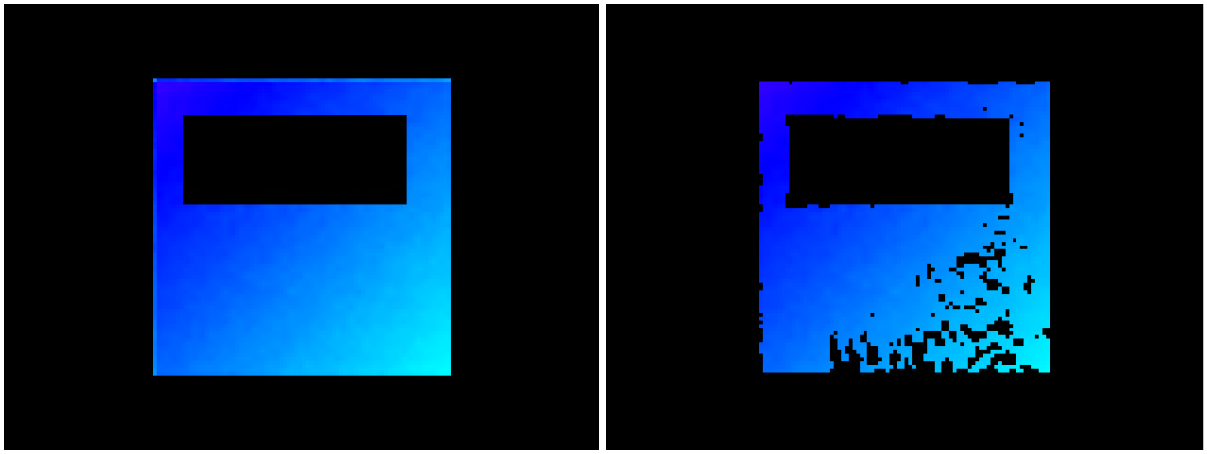

Figure 4.15: Ground truth and result of segmentation.

As discussed above, we only have the spatial processing level for the input parameter. And finally, the precision and recall were calculated at different levels from the same synthetic video. For training data of non-variation noise positions, we selected positions outside the facial region randomly. As a consequence, we calculated it n times and then averaged the precision and recall for all the results over the same level. The result is shown in the following table.

Both recall and precision decrease drastically when the amount of downsampling is increased. This is due to the increased SNR when averaging pixels together. At very small frame sizes, the noise positions were merged into blood flow ones.

	Precision	Recall	F_1-measure
Level 1	0.997	0.900	0.946
Level 2	0.988	0.843	0.906
Level 3	0.960	0.775	0.858
Level 4	0.755	0.647	0.712

Table 4.2: Results of segmentation on a synthetic video according to different spatial processing level.

4.2.4 Gradient on velocity field

Figure 4.16 summarizes the main results of the quantitative evaluation of our technique applied to synthetic video 1, whose velocity field is in the range $[0, 2\pi]$ in the direction of 135°. In our experiment, we varied the amount of downsampling before processing the videos. In the Figure 4.16, a downsampling level of n is equivalent to scaling the video down by a factor of 2^n.

It can be seen that both errors increase drastically when reducing the amount of downsampling. This is due to the increased SNR when averaging pixels together. At very small frame sizes, the angular error increases again as a result of the lower resolution. Downsampling by an effective factor of eight (level 3) gives the best results for this video. The achieved AAE is 20.13°, and the AME is 29.45%.

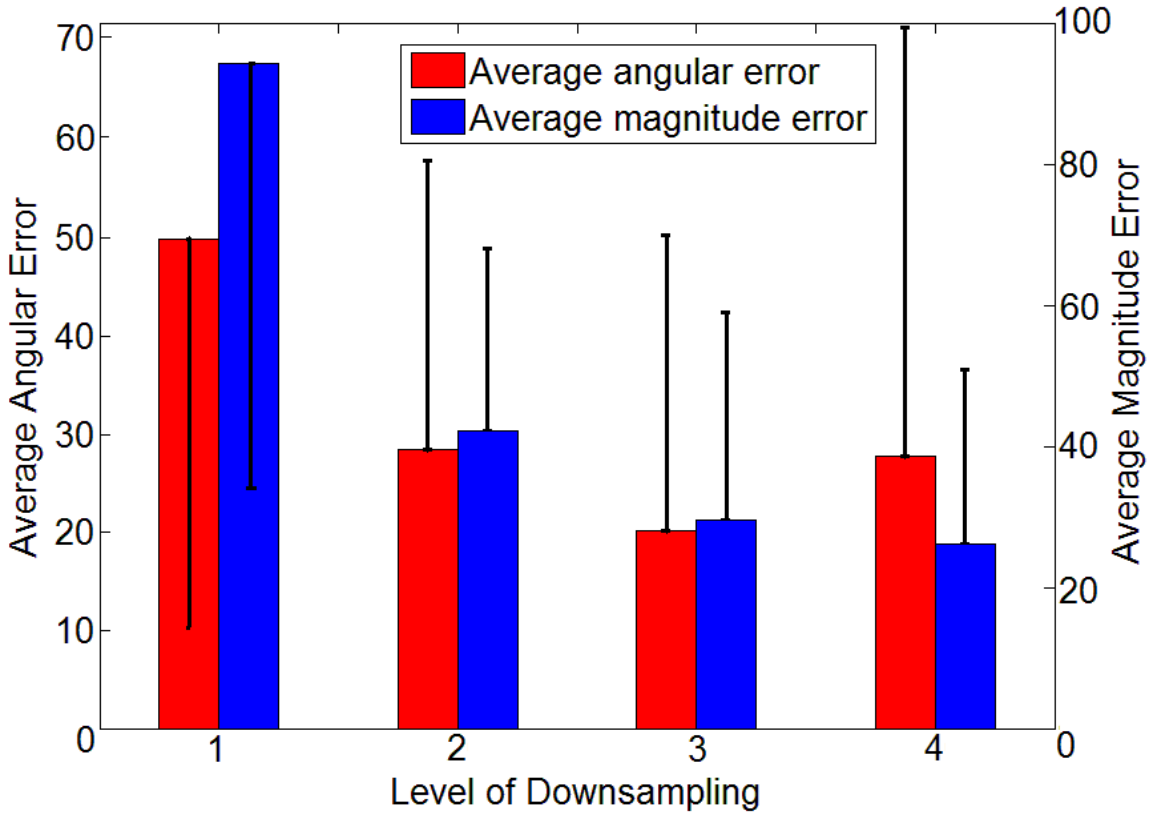

Figure 4.16: AAE (in degrees) and AME (in percentage) of the estimated velocity field for different amounts of downsampling. The black lines indicate one standard deviation.

Chapter 5

Conclusion and future work

5.1 Conclusion

We described a novel non-contact method for estimating blood flow velocities from videos of human. Our method takes video as input and uses the information in green channel to extract heart rate from the color variation in facial region, caused by blood circulation. A combination of downsampling and PCA are used to remove redundancy from the video. Multiple kernel based k-NN classification algorithm allows us to reduce noise significantly. The desired velocity field is finally computed by 2D Sobel filter horizontally and vertically as the gradient of the phase map.

Our results on real videos (10 videos from 6 subjects) show it is possible to obtain blood flow velocity measurements from skin at different positions (e.g. face, hand). Our evaluation of segmentation result on synthetic videos with recall and precision suggested that the increasing of spatial averaging result in the decreasing of segmenting result and the 1^{st} level performed best. The two different metrics (AAE and AME) on the evaluation of velocity field showed that both errors increase drastically when reducing the amount of spatial processing and at very small frame sizes, the angular error increases again as a result of the lower resolution. Our approach estimates the blood flow velocities in noisy synthetic videos with an angular error of 20% and an error in magnitude of 29%.

5.2 Future work

Several factors affected our results. First, our camera has a sampling rate of 30Hz. The low rate of sampling will cause the poor performance of electrocardiogram (ECG) used for blood flow analysis, such as signal to noise ratio. Spatial averaging will only partially address this issue. Second, the current work has been done only for indoor and with fluorescent light overhead as the source of illumination. The varied and suboptimal lighting conditions can probably affect the blood flow detection. The improvement of accuracy and efficiency of system in outdoor and more complicated environment (e.g., normal sunny day light conditions) are planned to be done to the study for future work. Finally, our video were only with around 10 seconds long. The increasing of video duration will improve our output but result in stronger motion artifacts due to subject's involuntary head movements. Other techniques like image stabilization (IS) can be used to reduce it.

For the future direction, one important is to develop approaches for moving subjects. This is complicated because, as we illustrated above, even subtle and smooth head motion cause strong artifacts in both amplitude and phase map. Some existing technique such as face detection and template tracking will even ruin the blood flow positions. Clearly with other more motions (e.g. talking), more sophisticated methods and filtering will be needed. Another important direction is to develop more applications based on our current blood flow detection system. As contact-free measurement is being more and more important in medical diagnosis, it would be valuable to develop related applications such as blood pressure detection and age detection in non-contact way.

References

[1] Opencv documentation. http://docs.opencv.org/.

[2] Opencv library. http://opencv.org/.

[3] Principle and implementation of eulerian video magnification. http://hahack.com/codes/eulerian-video-magnification/.

[4] The rule of the artery osteopathic reflection. http://www.osteopathyny.com/rule-artery-osteopathic-reflection/.

[5] Video magnification. https://people.csail.mit.edu/mrub/vidmag/.

[6] A. Akselrod-Ballin, M. Galun, M. J. Gomori, R. Basri, and A. Brandt. Atlas guided identification of brain structures by combining 3d segmentation and svm classification. In *Medical Image Computing and Computer-Assisted Intervention*, pages 209–216. Springer, 2006.

[7] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[8] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011.

[9] G. Balakrishnan, F. Durand, and J. Guttag. Detecting pulse from head motions in video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3430–3437. IEEE, 2013.

[10] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International journal of computer vision*, 12(1):43–77, 1994.

[11] C. M. Bishop. *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[12] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1):245–271, 1997.

[13] Y. Bo and C. C. Fowlkes. Shape-based pedestrian parsing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2265–2272. IEEE, 2011.

[14] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.

[15] L. S. Davis. A survey of edge detection techniques. *Computer graphics and image processing*, 4(3):248–270, 1975.

[16] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

[17] B. L. Dunmire, K. W. Beach, K. H. Labs, M. I. Plett, and D. E. Jr. Strandness. Cross-beam vector doppler ultrasound for angle-independent velocity measurements. *Ultrasound in medicine & biology*, 26(8):1213–1235, 2000.

[18] B. S. Everitt, S. Landau, M. Leese, and D. Stahl. Miscellaneous clustering methods. *Cluster Analysis, 5th Edition*, pages 215–255, 2011.

[19] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. In *Journal of Machine Learning Research*, pages 615–637, 2005.

[20] J. A. Finegold, P. Asaria, and D. P. Francis. Mortality from ischaemic heart disease by country, region, and age: Statistics from world health organisation and united nations. *International journal of cardiology*, 168(2):934–945, 2013.

[21] B. C. Fry, J. Lee, N. P. Smith, and T. W. Secomb. Estimation of blood flow rates in large microvascular networks. *Microcirculation*, 19(6):530–538, 2012.

[22] K.-S. Fu and J. K. Mui. A survey on image segmentation. *Pattern recognition*, 13(1):3–16, 1981.

[23] M. Garbey, N. Sun, A. Merla, and I. Pavlidis. Contact-free measurement of cardiac pulse based on the analysis of thermal imagery. *IEEE Transactions on Biomedical Engineering*, 54(8):1418–1426, 2007.

[24] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

[25] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[26] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. In *Technical Symposium East*, pages 2–9. International Society for Optics and Photonics, 1985.

[27] X. He, R. A. Goubran, and X. P. Liu. Using eulerian video magnification framework to measure pulse transit time. In *Proceedings of IEEE International Symposium onMedical Measurements and Applications*, pages 1–4. IEEE, 2014.

[28] I. Hein and W. D. O'brien Jr. Current time-domain methods for assessing tissue motion by analysis from reflected ultrasound echoes-a review. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 40(2):84–102, 1993.

[29] A. B. Hertzman and C. R. Spealman. Observations on the finger volume pulse recorded photoelectrically. *Am. J. Physiol*, 119(334):3, 1937.

[30] B. K. Horn and B. G. Schunck. Determining optical flow. In *Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, 1981.

[31] M. Huelsbusch and V. Blazek. Contactless mapping of rhythmical phenomena in tissue perfusion using ppgi. In *Medical Imaging*, pages 110–117. International Society for Optics and Photonics, 2002.

[32] J. E. Jackson. *A user's guide to principal components*, volume 587. John Wiley & Sons, 2005.

[33] J. A. Jensen. Estimation of blood velocities using ultrasound: a signal processing approach. 1996.

[34] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[35] Y. Kanzawa, Y. Kimura, and T. Naito. Human skin detection by visible and near-infrared imaging. In *Proceedings of Conference on Machine Vision Applications*, 2011.

[36] C. Kasai, K. Namekawa, A. Koyano, and R. Omoto. Real-time two-dimensional blood flow imaging using an autocorrelation technique. *IEEE Transaction on Sonics Ultrason*, 32(3):458–464, 1985.

[37] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of International Joint Conference on Artificial Intelligence*, volume 14, pages 1137–1145, 1995.

[38] C. Liu, A. Torralba, W. T. Freeman, F. Durand, and E. H. Adelson. Motion magnification. *ACM Transactions on Graphics*, 24(3):519–526, 2005.

[39] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of International Joint Conference on Artificial Intelligence*, volume 81, pages 674–679, 1981.

[40] R. Maini and H. Aggarwal. Study and comparison of various image edge detection techniques. *International journal of image processing*, 3(1):1–11, 2009.

[41] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[42] N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern recognition*, 26(9):1277–1294, 1993.

[43] M.-Z. Poh, D. J. McDuff, and R. W. Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics express*, 18(10):10762–10774, 2010.

[44] P. K. Sahoo, S. Soltani, and A. K. Wong. A survey of thresholding techniques. *Computer vision, graphics, and image processing*, 41(2):233–260, 1988.

[45] B. Schölkopf, C. J. C. Burges, and A. J. Smola. *Advances in kernel methods: support vector learning*. MIT press, 1999.

[46] J. Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[47] L. I. Smith. A tutorial on principal components analysis. *Cornell University, USA*, 51:52, 2002.

[48] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou. Multi-label learning with weak label. In *AAAI Conference on Artificial Intelligence*, 2010.

[49] C. Takano and Y. Ohta. Heart rate measurement based on a time-lapse image. *Medical engineering & physics*, 29(8):853–857, 2007.

[50] Z. Tu, X. Chen, A. L. Yuille, and S.-C Zhu. Image parsing: Unifying segmentation, detection, and recognition. *International Journal of computer vision*, 63(2):113–140, 2005.

[51] R. Urner, S. Ben-David, and O. Shamir. Learning from weak teachers. In *International Conference on Artificial Intelligence and Statistics*, pages 1252–1260, 2012.

[52] E. J. Van Kampen and W. G. Zijlstra. *Determination of hemoglobin and its derivatives*. Academic Press, 1965.

[53] W. Verkruysse, L. O Svaasand, and J. S. Nelson. Remote plethysmographic imaging using ambient light. *Optics express*, 16(26):21434–21445, 2008.

[54] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–511. IEEE, 2001.

[55] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. T. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Transactions on Graphics*, 31(4), 2012.

[56] T. Xu and G. R. Bashford. Lateral blood flow velocity estimation based on ultrasound speckle size change with scan velocity. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 57(12):2695–2703, 2010.