Multisensor Video Magnification

DINESH SHRESTHA

A thesis submitted in partial fulfillment for the
Master degree in computer science

to the

Department of computer science and Engineering
University Of Tromsø

Supervisor: Proff. Børre Bang (Department Head Computer Science) And
(Stipendiat) Tatiana Kravetc
June 6

Contents

Declaration Of Own Work

I hereby declare this thesis "Multisensor Video Magnification" is entirely my work and has not been published or submitted elsewhere for the requirement of a degree programme. Any material or work done by other within this thesis has been given due acknowledgement and listed in the reference section as the work is cited and referenced.

Abstract

This study was performed with the three main goals. The first of which includes the implementation of "Eulerian Video Magnification" for the normal videos to observe the subtle changes that cannot be seen through the naked eyes. Similarly, the second goal was to implement interpolation to thermal infrared videos. Finally, the third goal is to combine both the videos to amplify complex variations to reveal important aspects of the world around us.

The first experimental video data is taken by canon 1200D camera where videos were taken from the university with the same camera using tripod for the stability and better resolution and picture quality. Using Eulerian magnification, the experiments were performed for the series of videos that includes different human body parts. The specific region were face, hand and palm. Not only the human behaviour was observed but also the environmental objects was considered to see the similar types of subtle changes and reveal some more important facts and display them in indicative manner. Eulerian magnification amplifies color changes at the pulse rate to make the subtle color variation due to blood flow that are visible to the human eye. Simulated and real video of human skin are processed to reveal blood flow in the face, wrist and hand using a MATLAB implementation of the Eulerian magnification algorithm. From the research, it shows that it is very susceptible to impairment from motion and camera configuration.

The next experiment was done with the videos from thermal camera. The task was to increase the frame rate for the video where Butterflow algorithm was implemented [18]. The algorithm uses the library from OpenCV and uses the concept of "Two-Frame Motion Estimation Based on Polynomial Expansion" by Gunner Farneback [5]. The algorithm takes low frame rate videos and convert it into the required frame rate for the user. Finally, both the tested videos are overlapped in Qt-framework to reveal the changes in object behaviour.

Acknowledgement

I would like to express my sincere gratitude to Proff. Børre Bang (Head of Department of computer science and engineering and Associate. Lecturer Taitana Kravetc for allowing me to undertake this work. I am also grateful to my supervisor for their continuous guidance, advice, effort and invertible suggestions throughout the research and to carry out my thesis project successfully. I would also like to mention, James Pandey, one of my friend on helping me one some part of the project.

Motivation and Novelty of the project

Basically, the general motivation of EVM algorithm is health sector. But it can be implemented in other sector such as civil engineering to analysing subtle vibrations due to sound. There has been some successful effort on the assessment of vital signs, such as, heart rate, and breathing rate, in a contact-free way using a web camera [28], and even a smartphone [29]. Other similar products, which require specialist hardware and are thus expensive, include laser Doppler [30], microwave Doppler radar [31], and thermal imaging [32]. Since it is a cheaper method of assessing vital signs in a contact-free way than the above products, this research work has potential for advancing fields, such as, telemedicine, personal health-care, and ambient assisting living. Despite the existence of very similar products by Philips[33] and ViTrox Technologies [34] to the one proposed on this research work, none of these implement the Eulerian Video Magnification method. Due to being recently proposed, the Eulerian Video Magnification method implementation has not been tested in smartphones yet.

Using the frequencies, robot can be used to detect a human's heart rate, which would then allow the robot to verify whether the subject is a human or not. So, implementation of AI on the project would be next improvement for EVM. Also the subtle signal can be quantitatively analysed to enable other applications, such as extracting a person's heart rate from video, or reconstructing sound from a distance by measuring the vibrations of an object in a high-speed video. This is a multidisciplinary project that combines physics, programming and image processing technique in computer vision analysis. The peculiarity of the project is that the plan of the realization and the algorithm will allow one to tackle different problems related to design and modelling of the project.

List of abbreviations

EVM——	Eulerian Video magnification
IR———	Infrared
DCT——	Discrete Cosine Transform
IIR———	Immediate impulse response
BF———	Butterflow
NTSC —-	National Television System Committee
FFT——-	Fast Fourier Transform
FT———	Fourier Transform
FLIR——	Forward Looking Infrared«

Chapter 1

Introduction

Over the past few centuries microscope has revolutionized the world. They reveal to us a tiny world of object, life and structures that are too small for us to see through our naked eye and are tremendous contribution for science and technology. Similar to that the thesis work is to reveal us the tiniest motions and color changes in the objects. The changes that are impossible for us to see with our naked eyes.

For example, skin changes its color very slightly when the blood flow under it, that is incredibly subtle which is why when you look at other people you don't see their skin or their face changing color. Similarly, to support this task to next level, concept of thermal imaging is introduced which works on the temperature of the object. With a slight change in temperature, there is fluctuation in color of thermal video imaging which can be very helpful for the research to study new facts about the object behaviour due to change in temperature.

In the example of face in Eulerian Video magnification, the static picture is seen but once the video is processed then completely different image will be seen in video. The video output is the small change in skin color that is magnified 100 times so that they become clear and visible. The human pulse can even be observed and analysed how fast the heart is beating in the object. From this method, the blood flows in the face and other part of body can be observed.

Not just to visualize the pulse but also to recover the actual heart rates without touching the patient can be possible with the implementation of EVM method. The video can be taken with regular DSLR video camera and the result will be as accurate as the standard monitor in the hospital. And it doesn't have to be rerecorded and we can do it to other videos also. It is basically to analyse the changes in the light that are recorded at every pixel in the video over time and analyse those changes.

While seperating the signal and those subtle changes from the noise, one must careful because noise always exist in the videos. Therefore, image processing technique must be used to get the very accurate measurement of color at each pixel in the video and the way the color changes over the time to amplify those changes. It can be made bigger to create those types of enhanced videos or magnified videos that shows those changes.

However, it turns out that it is not only to show tiny changes in color, but also tiny motions because the light that gets recorded in the cameras will change not only if the color of the object changes, but also if the object moves. The veins and arteries that are pulsing in our bodies can also be recorded. Furthermore, our eyes are constantly moving in the wobbly motion. Even

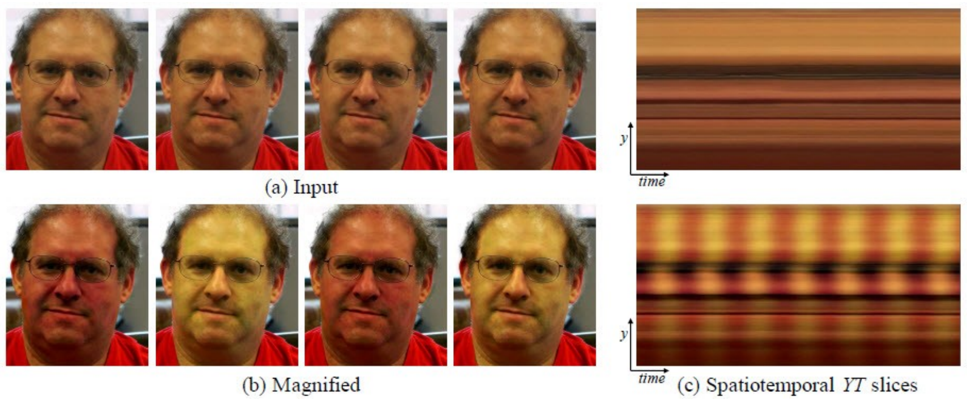


Figure 1.1: An example of Eulerian Video Magnification for visualizing human pulse. (a) Four frames from the original video sequence(face). (b) The same four frames with the subject's pulse signal amplified. (c) A vertical scan line from the input(top) and output(bottom) videos plotted over time that shows amplification of the periodic color variation [9]

when a person is sitting still, there's a lot of information that can extracted from their breathing patterns, small facial expressions and the small movements. Small mechanical movements can also be magnified, such as vibrations in engines that can help engineers detect and diagnose machinery problems, or see how building and structures sway in the wind and react to the forces. Measuring those motions is one thing and actually seeing those motions as they happen is the whole different thing. This tool not only allows us to look the world in the new way but also redefines limits of video cameras. And one of the interesting phenomena related to this topic is sound. As the known fact, the sound basically changes in the air pressure that travel through the air and those pressure waves hit objects and they create small vibrations in them, which is how hearing and recording of the sound is done. But it turns out that the sound also produces visual motions. Those are motions that are not visible to us but are visible to a camera with the right processing.

For example, we know the singers can break a wine glass with the correct note that's in the resonance frequency of that wine glass. The experiment can be performed with the wine glass and the magnify the motions 250 times and from that the vibrations in the glass can be seen and resonates in response to the sound. From that it can be analyzed that the process can be reverted and recover the sound from the video by analysing the tiny vibrations that sound waves create in objects, and essentially convert those back into sounds that produced them.

Since the project is Multisensor there will be two or more hardware. In the project, reuglar camera and FLIR camera are two sensor to capture the same videos from both the cameras. Both the camera or the sensors works at different frame rates and also one works on optical flow and that of thermal camera works on temperature of an object. For the visible camera to capture the video it must be more than 24 frames or images per second. It is the convention that is also applied for movies. However, these days there are cameras that can capture the videos at 60 Fps or more. It is different for the case of thermal camera or the infrared camera as it can capture the videos at lower frame rate i.e. from 7 Fps to 9 Fps. Similarly, thermal camera works on temperature of the object. So, it is effected with the environment temperature of the object. Different types and quality of thermal camera are available in the market.

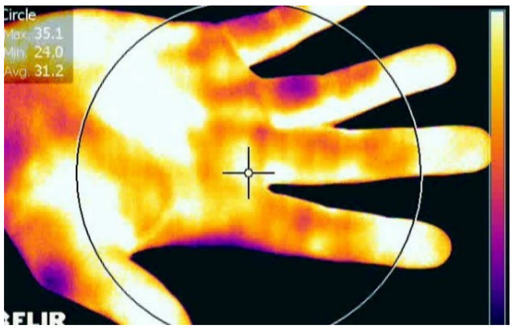


Figure 1.2: Thermal image of hand with difference in color pattern

Major difference of thermal camera with visible camera is its focusing lenses that cannot be made of glass because the glass blocks long-wave infrared light. So, special material is used such as sapphire crystal to make glass of thermal camera. Images from infrared camera are monochrome as the camera generally uses an image sensor that doesn't distinguish different wavelength of infrared radiation. Warmest part of the image are white colored, intermediate temperatures are red and yellow and the dimmest or coolest part is black. Thermal cameras can be used by firefighter to see through the smoke to localize hotspots of fires, power line maintenance to locate overheating joints and parts to eliminate potential hazards and many other applications that work on principle of heat and temperature.

To overlap the two video sources with different frame rates and resolution, it is quite challenging task as both the videos run at different frame rates. To solve this, interpolation of the videos frame is done so that it gets matched with the frame rate of the visible camera. After that with the use of the necessary blending techniques the frames are overlapped and create the videos are made out of it.

1.1 Problem Statement

The problem description includes the framework for testing and examining the possibilities to reveal and amplify the invisible signal in the given temporal frequency range of interest. Thesis report first describes the application of spatial and temporal filtering to amplify color changes in human skin at the pulse rate through a process called EVM. The process involves spatial filtering using the concept of image pyramid, as described in section 5.1.2. For motion magnification Laplacian pyramid is used because even subtle motion would be apparent in high pass filtered portion of a frame. However, Gaussian pyramid is used at a selected level to find regions of color change when filtering for color variation at a selected level to find regions of color change and minimize the high frequency effects of motion.

Similarly making the set up that consists of both the regular visible video and IR-video and combine them to see the important changes in the object behaviour. The outline for the work is as follows:

- Reimplementation of MITs "Eulerian video magnification" where the task is to reduce the noise and use spatial filtering method to boost the power of the specific signal.

- Using spatial decomposition that is followed by temporal filtering to the same video frames
- Amplification of the filtered spatial bands and video reconstruction
- Extension of the video analysis to overlap the two video sources with different video frame rate and resolution.
- Also examine the correlation between both the color and temperature over the time and interpolate the thermal infrared video in order to make its frame equal to visible camera video frame rates.

1.2 Research Objectives

The objectives of this research are to re implement the code for "Eulerian Video Magnification" that was developed at MIT and Quanta Research Cambridge, Inc. for different videos to see the important changes in the behaviour of the object. Analyse those subtle changes to use it for the research project in the examples taken for the project. Implementation are first done where few examples from their original sites has been taken. Further improvement has been done over Eulerian video magnification i.e. phase based approach to make it more applicable in many areas.

Thermal camera works on temperature of the object at low frame rate. Frame rates per seconds defines how many images are needed to make an one second video. Therefore, technique is needed so that framerate can be made similar to the framerate of the visible camera video and easily overlap them. Therefore, the main objectives of this thesis project include the part from "Eulerian video magnification" and the part from thermal video imaging to combine them to see the important changes in the behaviour of the object.

Chapter 2

Status of Knowledge

Status of knowledge includes the what has been done earlier and what is needed to be further implemented and how to improve the work. Video Information for the normal video camera

Frame width	1280px
Frame height	720px
Data rate	585kbps
Total bit rate	745kbps
Frame rate	30 fps
File format	.mp4
Audio	type
Bit rate	160kbps
Channels	2 stereo type
Audio Sample rate	48kHz

Video Information for the Flir video Camera

Frame width	640px
Frame height	480px
Frame rate	7 or 8 Fps
Bitrate After video interpolation	30 Fps

2.1 Existing Theories

There are millions of pixels on a digital camera sensor where each pixel has a photo site(cavity) that is uncovered when you press the shutter release button. After that the camera closes each photosite and works out how many photons fell in each cavity at the end of the exposure. Depending on the number of photons in the photosite, the camera determines the intensity of each pixels. However, each small cavity cannot distinguish how much of each color has fallen, so the sensor can only record grey scale images. A filter called a Bayer mosaic filter is placed over the sensor to record the color pictures. The filter consists of three colors of small filter i.e. red, green and blue that only allows light of a certain color to reach each cavity. It contains twice as

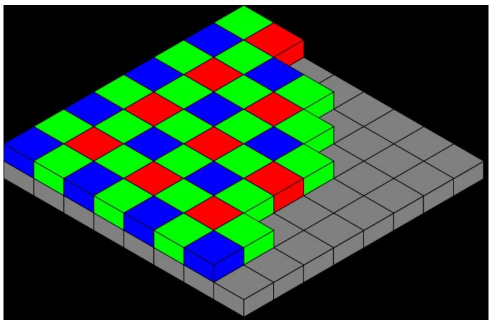


Figure 2.1: Bayer arrangement of color filter on pixel array of image sensor [35]

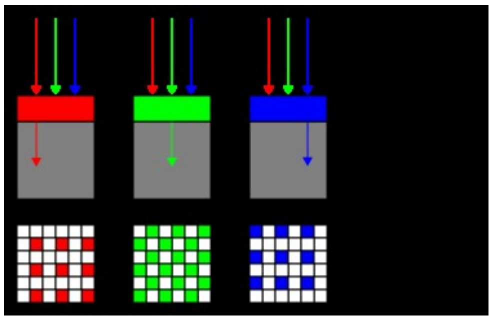


Figure 2.2: profile and cross section of sensors[35]

much green filters in Bayer's mosaic to accurately reflect the way the human eye sees color. It means that it is more sensitive to green light. The array only allows the intensity of one of the three colors in each cavity to the sensor. Finally, an image is made, once all the colors are put together.

Filters are a piece of glass that is attached to the front of a lens through which all incoming light has to pass [35]. Some of the filters are clear and specially designed to protect the front of the lens, but some are coated with special chemicals to serve for the various other purpose. However, IR filter is for IR photography. Infrared filter can be categorized into two different types, one that block IR light while passing visible light and one that block visible light while passing infrared light. Digital video and still cameras use IR blocking filter to prevent unwanted IR light from reaching the sensor, which is sensitive to near infrared. There are several types of infrared passing filters, also called low-pass filters.

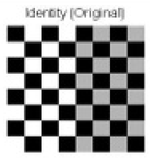
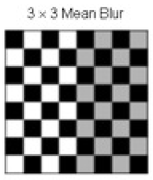
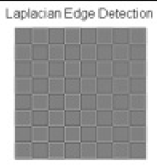
The use of computer algorithm to perform image processing on digital images allows a much wider range of algorithms to be applied to the input data and can avoid the problems such as build-up of noise and signal distortion during processing. Filtering are used to blur and sharpen the digital image and can also be performed on the spatial domain by convolution with specifically designed kernels (filters array) or in the frequency (Fourier) domain by masking specific frequency regions.

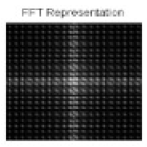
Below are some of the filter types with mask and examples:

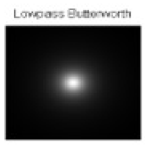
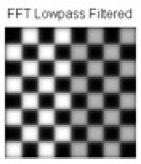
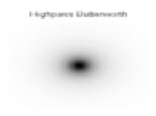
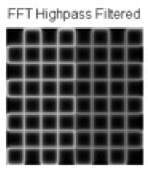
2.2 Thermal Imaging

Thermal image has advantages over visible images because of its illumination invariant property. It is less sensitive to the variation in object appearance caused by illumination changes because thermal infrared sensor measures the heat energy radiation emitted by the object rather than the reflected light. It is represented by heat patterns emitted from an object where object emits different amount of IR energy according to their temperature and characteristics. Similarly, it generates imaging features that reveals thermal characteristics of the object pattern and it utilizes anatomical information of the object which is unique that can be measured at distances using passive IR sensors.

The facial images contains less important texture boundaries, extracting the object boundaries in thermal images is a challenging task, because of its amorphous nature and lack of sharp

Filter type	Kernel or mask	Example
Original image	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Spatial lowpass	$\frac{1}{9} X \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Spatial High pass	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	

Fourier Representation	Pseudo-code:
	Image =checkboard
	F= Fourier Transform of image
	Log(1+Absolute Value(f))
		

Fourier Lowpass		
Fourier High pass		

[H]

Figure 2.3: Table. Different types of filter with mask [36]

boundaries in images. One of the important problem with thermal imaging is that with the change in object temperature, the characteristics of thermal object also gets changed significantly. Thermal infrared video can provide essential information about the temperature distribution of the human body or the object that works on heating principle. It is a radiation free technique that gives relevant information about the pathology and patho-physiology of the human body thermoregulation. It is the fact, that surface temperature depends of a complex set of thermal exchanges and functionality of tissues, vasculature, metabolism and environmental conditions. Therefore, it is well suited to pick up changes in skin temperatures that might occur due to inflammation, or due to other health hazards.

In the same way, computer vision, computerized image processing and pattern recognition technique has always been useful in getting and evaluating medical thermal images and is being an important tool for clinical diagnosis. When the thermal image is combined with visual images with the aim of relating the specific skin surface temperature distribution to the human anatomy and also for monitoring the efficacy of any treatment. Therefore, overlaying two images from different sources and creating the video out of it in an automatic computer vision method will provide a useful tool for improving medical diagnosis. In order to see the important changes in the human and also in the environmental objects, examples on plant has been taken and also on human body part i.e. hand, palm and face.

The main task is to simplify the model as much as possible, without losing the precision. While choosing the best method one should remember that the modelling occurs on a small scale.

2.3 Existing implementations of EVM

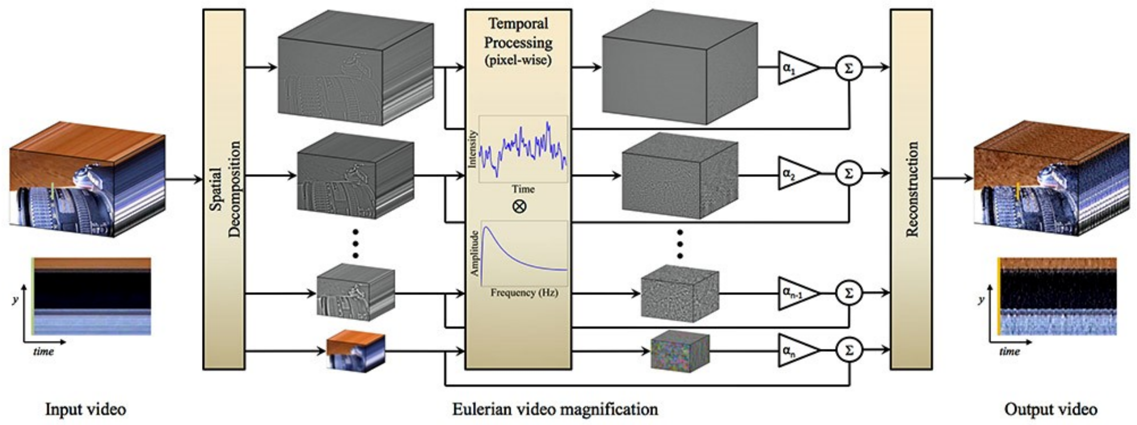


Figure 2.4: Overview of Eulerian Video Magnification where system first decomposes the input signal into different spatial frequency bands and applies the same temporal filter to all the bands. The filtered spatial bands are then amplified by amplification factor ∞, and added back to the original signal and collapsed to generate the input output video. The choice of temporal filter and amplification factors can be tuned to support different applications. [9]

There are four steps to process an input video by EVM:

- Select a temporal bandpass filter
- Select an amplification factor alpha
- Select a spatial frequency cut off (specified by spatial wavelength,) beyond which an attenuated version of alpha is used.
- Select the form of attenuation for α- either force α to zero for $\lambda < \lambda_c$

Amplification factor and cut-off frequencies are all customizable by the user in the real-time application.

First-order Motion To support the theory and explain the relationship between temporal processing and motion magnification, 1D signal is supposed that undergoes translational motion. Let $I(x,t)$ be the image intensity at position x and time t. As the image undergoes translational motion, the observed intensities can be expressed with respect to a displacement function $\delta(t)$ such that $I(x,t) = f(x + \delta(t)) and I(x,0) = f(x)$. The goal of the motion magnification is to synthesize the signal from the source.

$$I(x,t) = f(x + (1+\alpha)\delta(t)) \tag{2.1}$$

Here, α is the amplification factor. Now, let's assume that image can be approximated by a first order Taylor series expansion. Then we write the image at time t, in first order taylor expansion as,

$$I(x,t) \approx f(x) + \delta(t)(\frac{\partial f(x)}{\partial (x)}) \tag{2.2}$$

Let B (x, t) be the result after applying temporal band pass filter to I(x, t) at every position in x. Let us assume that the motion signal,$\delta(t)$is within the passband of temporal bandpass filter. Then we get,

$$B(x,t) = \delta(t)(\frac{\partial f(x)}{\partial (x)}) \tag{2.3}$$

After that the bandpass signal is amplified by α and add it back to I (x,t), that gives the resulting processed signal,

$$I(x,t) = I(x,t) + \alpha B(x,t) \tag{2.4}$$

Combining the equations 2,3, and 4 we get,

$$\check{I}(x,t) = f(x) + (\alpha+1)\delta(t)(\frac{\partial f(x)}{\partial (x)}) \tag{2.5}$$

Assuming the Taylors expansion holds for amplified perturbation $(\alpha + 1)\delta(t)$ we can relate amplification of the temporally bandpass signal to motion magnification. The final output is

$$\check{I}(x,t) = f(x + (\alpha+1)\delta(t) \tag{2.6}$$

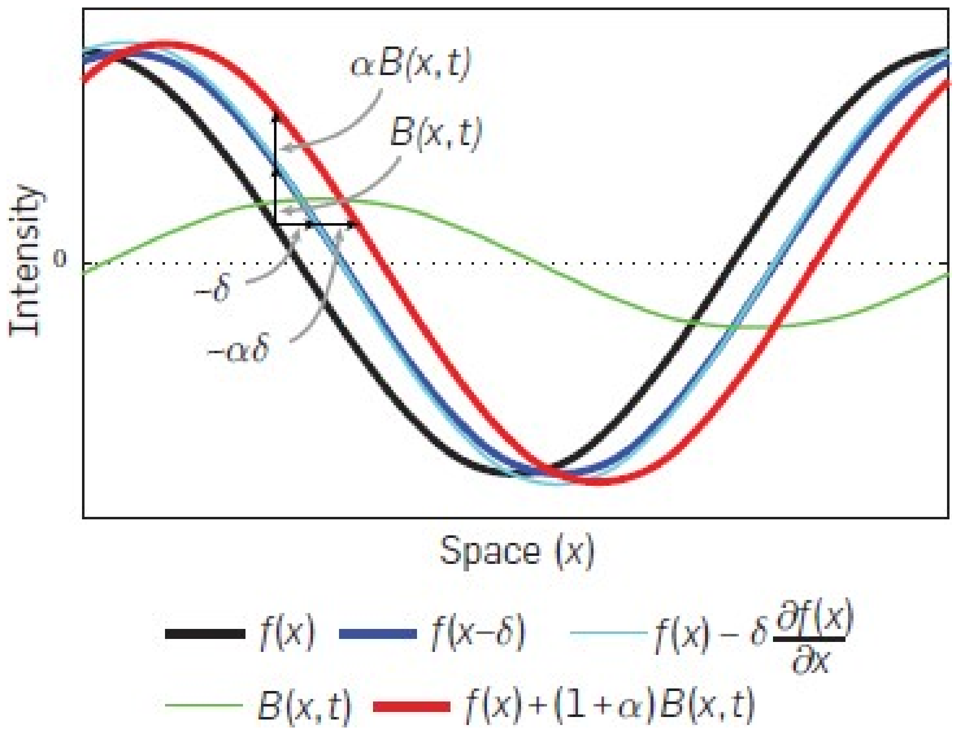


Figure 2.5: The input signal is shown at two time instants: *I(x,t)=f(x)* at time t and $I(x,t+1) = f(x+\delta)$at time t+1. The first order Taylor series expansion of I(x,t+1) about x approximates well the translated signal. The temporal bandpass is amplified and added to the original signal to generate a larger translation. In this example infinity =1, magnifying the motion by 100%and the temporal filter is a finite difference filter, subtracting the two curves. [9]

Result shows that the processing applies motion magnification i.e. the spatial displacement $\delta(t)$ of the local image f(x) at time t, is amplified by $(\alpha+1)$. The process can be illustrated for a single sinusoid in figure.

Figure explains the process illustration for a sinusoid signal i.e. for a small displacement $\delta(t)$and a low frequency cosine wave, Taylor series expansion provides good approximation for the translated signal at time t+1. When boosting the temporal signal by amplification factor α and adding back to I(x,t), we can approximate the wave translated by $(1+\alpha)\delta$. Now let us consider more general case for $\delta(t)$ that is not entirely within the pass band of the temporal filter. So, for this situation let $\delta_k(t)$ indexed by k. Each $\delta_k(t)$ will be attenuated by the temporal filtering factor γ_k. Now, the result in bandpass signal will be

$$B(x,t) = \sum_k \gamma_k \delta_k(t) (\frac{\partial f(x)}{\delta(x)}) \tag{2.7}$$

This temporal frequency dependent attenuation can equivalently be interpreted as a frequency-dependent motion magnification factor, $\alpha_k = \gamma_k \alpha$, giving motion magnified output,

$$\tilde{I}(x,t) \approx f(x) + \sum_k (1+\alpha_k)\delta_k(t) \tag{2.8}$$

The modulation of the spectral components of the motion signal becomes the modulation factor in the motion amplification factor, α_k for each temporal sub band, δ_k for the motion signal.

2.4 Butterflow Algorithm

Butterflow algorithm is used in the project to make motion interpolated videos. It works by increasing the video frame rate by rendering the intermediate frames based on the motion using the combination of pixel wrapping and blending technique. It makes the video smoother by simply blending between frames. Rendering the intermediate frames between the existing frames using the process called motion interpolation is utilized by BF algorithm. Given the two existing frames, A and B, this algorithm generate frames C., C.2………C.n positioned between the two. It wraps the pixels based on motion to generate new ones in contrast to other tools that can blend or dupe frames. The perception of more fluid animation commonly found in high frame rate videos can be obtained from the additional interpolated frames. Source file is written in python with using optical dense flow algorithm. BF algorithm uses OpenCV library and an algorithm called "Two Frame Motion Estimation Based on Polynomial Expansion" by Gunnar Farneback [5].

2.4.1 Polynomial Expansion

The idea for polynomial expansion is to calculate neighbourhood of each pixel with a polynomial where the main focus is on the quadratic polynomials giving local signal model, expressed in a local coordinate system,

$$f(x) \sim x^{\mathrm{T}}Ax + b^{\mathrm{T}} + c \tag{2.9}$$

Where A is a symmetric matrix, b a vector and c a scalar. Estimation of the coefficient are from a weighted least square fit to the signal values in the neighbourhood. Polynomial expansion is based on certainity and applicability where certainty is set to zero outside the image as it has no impact on coefficient estimation.The applicability determines the relative weight of the points in the neighbourhood based on their position in the neighbourhood. Most of the weight is at the center point and let the weight decrease radially. The scale of the structure is determined by the width of the applicability which will be captured by the expansion coefficients.

2.4.2 Displacement Estimation

Analyzing polynomial that undergoes an ideal translation as the result of polynomial expansion is that each neighbourhood is approximated by a polynomial. Let us consider the exact quadratic polynomial

$$f_1(x) = x^{\mathrm{T}}A_1x + b^{\mathrm{T}}{}_1x + c_1 \tag{2.10}$$

And we construct a new signal f_2 by a global displacement by d,

$$\begin{aligned} f(x) &= f_1(x-d) \\ &= (x-d)^{\mathrm{T}}A_1(x-d) + b_(1)^(T)(x-d) + c_(1) \\ &= x^{\mathrm{T}}A_1x + (b_1 - 2A_1d)^{\mathrm{T}}x + d^{\mathrm{T}}A_1d - b_1d + c_1 \\ &= x^{\mathrm{T}}A_2x + {b_2}^{\mathrm{T}}x + c_2 \end{aligned} \tag{2.11}$$

Now equating the coefficients in quadratic polynomial gives,

$$A_2 = A_1 \tag{2.12}$$
$$b_2 = b_1 - 2A_1 d \tag{2.13}$$
$$c_2 = d^{\mathrm{T}} A_1 d - {b_1}^{\mathrm{T}} d + c_1 \tag{2.14}$$
$$\tag{2.15}$$

From the observation, we can see that translation d can be solved atleast if A1 is non-singular.

$$2A_1 d = -(b_1 - b_1) \tag{2.16}$$
$$d = -(1/2){A_1}^{\text{-1}}(b_2 - {}_1) \tag{2.17}$$

2.4.3 Practical considerations

For the practical consideration, we replace the global polynomial in equation (2) with the local polynomials approximations. So we start by doing the polynomial expansion of both the images, giving the coefficient$A_1(x), b_1(x)$and $c_1(x)$for the first image and $A_2(x), b_2(x)$and $c_2(x)$ for the second image. Ideally $A_1 = A_2$ according to equation (4) but we have to settle it for practical approximation

$$A(x) = (A_1(x) + A_2(x))/2 \tag{2.18}$$

And we also introduce,

$$\Delta b(x) = -(1/2)(b_2(x) - b_1(x)) \tag{2.19}$$

To obtain the primary constraint,

$$A(x)d(x) = \Delta b(x) \tag{2.20}$$

d(x) represents the replacement of the global displacement in equation (3) with a spatially varying displacement field.

2.4.4 Estimation over neighbourhood

To eliminate the noise, we make the assumption that the displacement fields is slowly changing, so that we can integrate information over a neighbourhood of each pixels. Therefore, we try to find $d(x)$ satisfying equation (11) as well as possible over neighbourhood I of x,

$$\sum_{\Delta x \epsilon I} w\Delta x \parallel A(x + \Delta x)d(x) - \Delta b(x + \Delta x) \parallel {}^2 \tag{2.21}$$

Where $w(\Delta x)$ is the weight function for the points in the neighbourhood and the minimum is obtained for

$$d(x) = (\sum wA^{\mathrm{T}}A)^{\text{-1}} \sum wA^{\mathrm{T}}\Delta b \tag{2.22}$$

Now the minimum value is given as

$$e(x) = (\sum w\Delta b^{\mathrm{T}}\Delta b) - d(x)^{\mathrm{T}} \sum wA^{\mathrm{T}}\Delta b \tag{2.23}$$

Practically $\mathrm{A}^{\mathrm{T}}\mathrm{A}$, $\mathrm{A}^{\mathrm{T}}\Delta b$ and $\Delta b^{\mathrm{T}}\Delta b$ are computed pointwise and average these with w before the displacement is solved. It is sometime useful to add the weight as in $c(x+\mathrm{x})$ in equation (12) because it can be handled easily by scaling A and Δb accordingly.

Chapter 3

Literature Review

3.1 Frame interpolation via Adaptive Convolution Existing Implementation

Interpolation of video frames simply means inserting or adding new frames between two frames. Given the previous and next frames where the work is to insert the frame between the two. Video frames can be interpolated by averaging the previous and the next frame if it is without motion estimation. The quality and the performance of the frame interpolation will increase if the motion estimation is included in the process during interpolation. Motion estimation is used to estimate the motion vectors between frames and the pixels are then interpolated along the path of motion vectors. Many different frame interpolation algorithms has been developed and utilized in the field of computer vision and image processing but most of them concentrate on high frame rate video. Block matching technique can be simply used in motion estimation that is explained in “Video Frame interpolation by adaptive convolution” by Simon Niklaus, Long Mai and Feng liu in their project [7]. The variation is mainly between the block sizes, search space and search technique in the algorithm. Following figure explain the frame interpolation by convolution network.[**?**][3]

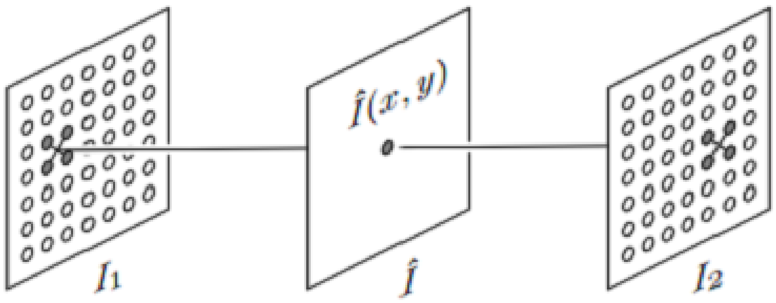


Figure 3.2: Interpolation by motion estimation and color interpolation .[7]

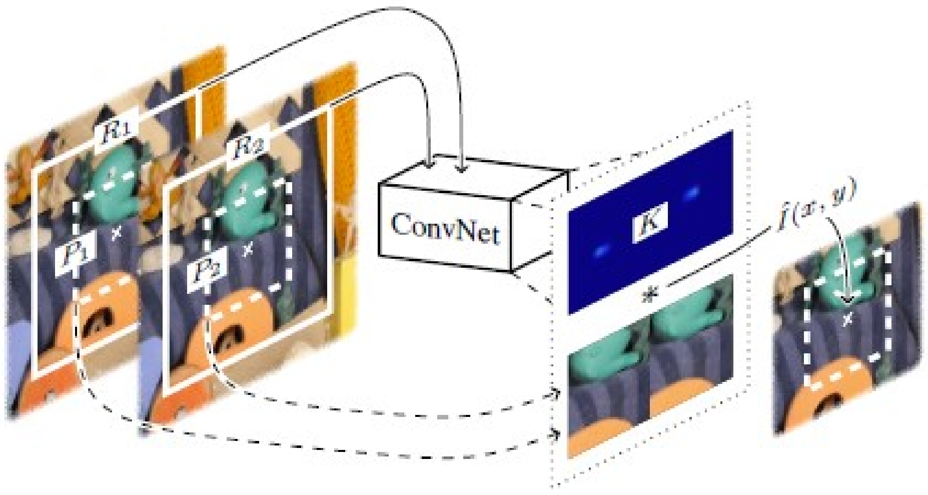


Figure 3.1: Pixel interpolation by convolution. For each output pixel (x,y), this method estimates a convolution kernel K and uses it to convolve with patches P1, and P2 centered at (x,y) in the input frames to produce its color I(x,y).[7]

Video frame interpolation aims to interpolate frame $\hat{I}$ temporally in the middle of the two input framesI_1 and I_2 within the same video. Traditional interpolation methods find the color of pixel $\hat{I}(x, y)$ in the interpolated frame in two steps: dense motion estimation (using optical flow) and pixel interpolation. As the matter of fact, we can find pixel for $\hat{I}(x, y)$ using corresponding pixel in $I_1(x_1, y_1)$ and $I_2(x_2, y_2)$ and then interpolate the color these corresponding pixels. Resampling images $I_1 and I_2$ to obtain the corresponding values to produce a high-quality interpolation result often involves pixels from both the frames. Rounding the coordinates to find the color in $I_1(x_1, y_1)$ and $I_2(x_2, y_2)$ is prone to aliasing while resampling with the fixed kernel sometimes cannot preserve sharp edges well. So in that case advance re-sampling methods exist and can be used for edge preserving re-sampling, which however requires high quality optical flow estimation. The method that is used in the program is to combine motion estimation and pixel synthesis into a single step and formulate pixel interpolation as a local convolution over patches in the input images $I_1 and I_2$. So, the combination of motion estimation and pixel synthesis into a single step provides more accurate solution than the two-step traditional procedure.

3.2 Description concerning limitations and size of the task/project

3.2.1 Limitation of Thermal Camera

- The quality of camera varies the price range (often US dollar 3,000 or more) due to the expense of the larger pixel array (state of the art 1024X720), while less expensive models (with pixel arrays of 40x40 up to 160x120 pixels) are also available. Camera with fewer pixels reduce the image quality making it more difficult to distinguish proximate targets within the same field of view.

- Many models do not provide the irradiance measurements used to construct the output image; the loss of this information without a correct calibration for emissivity, distance, and ambient temperature and relative humidity entails that the resultant images are inherently incorrect measurements of temperature

- Images can be difficult to interpret accurately when based upon certain objects, specifically objects with erratic temperatures, although this problem is reduced in active thermal imaging

- Accurate temperature measurements are hindered by differing emissivity's and reflections from other surfaces

- Most cameras have $\pm 2\%$ accuracy or worse in measurement of temperature and are not as accurate as contact methods

- Only able to directly detect surface temperatures

3.2.2 Limitation of Eulerian Video magnification

when the motions are small, this approach to motion magnification is robust and fast. If the motions are large, this processing can result undesired output. However, one can detect when this happens and suppress magnification in this case by first stabilizing the video. Limitation to how well spatio-temporal filtering can remove noise and amplified noise can cause image structures to move incoherently. Linear amplification depends on first order Taylor series expansion, so when the input motion is too large, the initial expansion is not accurate and the output contains ghosting artifacts instead of magnified motion. Secondly the noise in the video is amplified by factorα and the output video has noise of variance $2\alpha^2\sigma^2$ that is much larger amount than in the input video. In [9] noise amplification was partially reduced by reducing the amplification of high spatial frequency temporal variation. Laplacian pyramid is constructed for the temporal variations and using a lower amplification factor for high spatial frequencies. Figure 3.3 is palm when the noise is introduced in the frame.

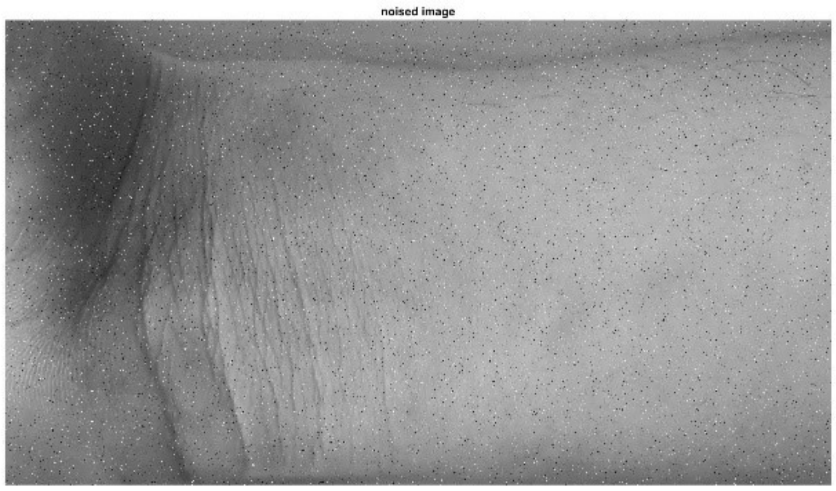


Figure 3.3: Noise introduced in figure of palm

3.3 Filters and Wavelets

A method of estimating a signal's frequency content is called frequency domain analysis. The signal must be set to frequency domain to preform frequency domain analysis. Fourier transform equation is used for this analysis as shown in equation (1).

$$f(\omega) = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}dt \tag{3.1}$$

Where $F(\omega)$ is Fourier transform of f(t), the signal to be analyzed and ω is the frequency in radian/second and t is time in seconds. This equation is used for the continuous signal and for the discrete single, Discrete Fourier transform is used as show in equation (2)

$$F[k] = \sum_{n=0}^{N-1} f[n]e^{-2\pi jkn/N} \tag{3.2}$$

Where $F[k]$ is the DFT of N samples of the function$f[n]$ and k and integer n represents the frequency index and the time index for the samples. The number of calculation can be significantly reduced using FFT that uses "Divide and conquer" algorithm to reduce the number of calculations. DFT is completed in N2 calculations and that of FFT is completed in NlogN calculations where N is the number of samples and logarithm is taken in base 2. A required filter is selected to isolate a range of frequencies within the signal. An ideal low pass filter selects only the frequencies below the cutoff frequency i.e. rejects all the frequencies above the cutoff frequency. The frequency response of this filter has rectangular shape but in time domain impulse response is sinc function as shown is figure 3.4 and 3.5.

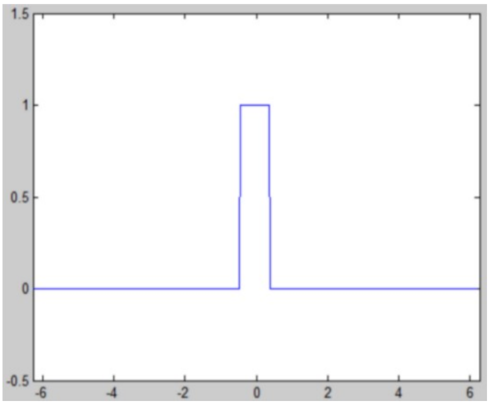


Figure 3.4: Frequency response of ideal low pass filter [8]

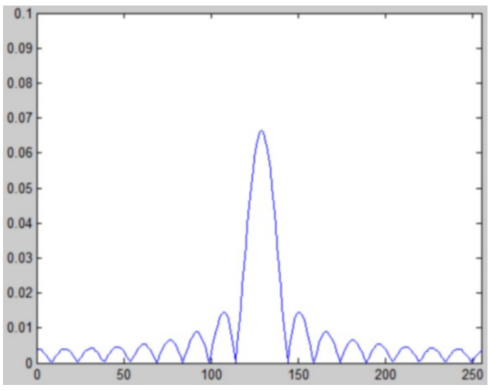


Figure 3.5: corresponding time domain impulse response[8]

Opposite to low pass filter is high pass filter that rejects the low frequency below cutoff frequency and pass the signal above that. There are two cutoff frequency in band pass filter that rejects the frequencies which are not between those two cutoff frequencies. As shown in the figure (3.4), an ideal low pass filter is used for temporal processing in this project. By setting the zero values to all the frequencies that are outside of the selected band, this filter is implemented.

A mathematical transform that was developed to provide localization in both time and frequency is called wavelet [8]. The Fourier transform is only localized in frequency to determine what frequencies are present in the signal. The continuous wavelet is the method by which a signal, f(t) is decomposed into wavelets, $\Psi_(s, \tau)$, as shown in the equation (3.3).

$$\Psi_(s, \tau) = \int_{-\infty}^{\infty} f(t)\psi_(s, \tau)^*(t)dt \tag{3.3}$$

where * in the equation is complex conjugate, variables s and are scaling and translation indices. They are also dimensions of the output of the continuous wavelet transform. Since continuous wavelet is inefficient in many ways because of redundancy in the scaled wavelets infinitely many wavelets are generated and have no close form of analytical solutions. Therefore, to solve this issue, discrete wavelets are defined, as given in equation (3.4).

$$\Psi_(j, k)(t) = (1/\sqrt{(}s_0 j)\psi((t - k\tau_0 s_0)/s_0 j) \tag{3.4}$$

Where j and k denotes scaling and translation. The variable s_0>0 is a fixed scaling step for discrete wavelets and τ_0 is a fixed translation factor depending upon the scaling step. In addition to that discrete wavelet is made orthonormal by using equation (3.5).

$$\begin{aligned}\int \psi_{j,k}(t)\psi^{*}_{m,n}(t)dt &= \{1 if j = mand k = n \\ &= 0 otherwise\}\end{aligned} \tag{3.5}$$

Wavelets is applied to filter certain frequencies. As every stretch in time domain compresses the frequency spectrum of wavelet by the same factor, to cover DC with a scaled wavelet, the scaling need to be infinite. Therefore, this issue needs to be resolved. For this wavelet function $\varphi(t)$, is defined in equation 3.6 with scaling steps up to j,

$$\varphi(t) = \sum_{j,k} \gamma_{j,k}(t) \tag{3.6}$$

Only the scaling and translation dimension are discrete but the transform is still not completely discrete. Therefore by defining the functions of scale factor j based on the previous steps in scaling function the multiresolution equation discretize the scaling and wavelet in time as shown in equation below:

$$\varphi(2^j(t)) = \sum_k h_{\mathrm{j+1}}[k]\varphi(2^{j+1}t - k)\psi(2^j(t)) = \sum_k g[k]\varphi(2^{j+1}t - k) \tag{3.7}$$

where scaling coefficient, h[] operates as a low pass filter when convolved with a signal and that of wavelet coefficient, g[k], operates as a highpass filter. The discrete wavelet transform is applied to both filters for the same signal giving two outputs: the low frequency content and the high frequency content of the signal. Each wavelet has different properties that amplifies different features in an image.

3.4 Image Pyramid

It is based on the pyramid concept of applying the discrete wavelet transform and down-sampling by 2 repeatedly, until the size of the image is too small to apply the filter. So, for each level of pyramid there is a low pass filtered image and the details that was removed from the image or the high pass filtered image.

There are two different pyramid implementations and are Laplacian and Gaussian pyramids. Each level of the Gaussian pyramid is the low pass filtered version of the previous level.

The application areas of image pyramid are mostly for JPEG image compression, restoration and image enhancement. To encode the image in as few bits as possible image pyramid is used by JPEG image compression using few bits to encode the higher frequencies than are used to encode lower frequency in specific bands. Noisy images uses image restoration by zeroing the values of high frequency components in the Laplacian pyramid. While image enhancement acts is opposite than the image restoration to bring out detail in levels of pyramid that are lost due to an image that is not is focus.

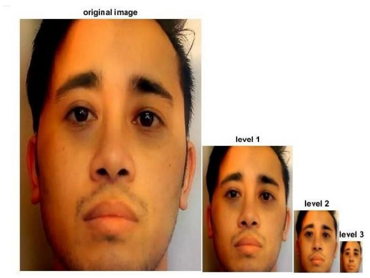


Figure 3.6: Gaussian Pyramid level of an image

Figure 3.7: level scaled low pass filter effect

Figure 3.8: level scaled low pass filter effect

3.5 Digital Video

Digital video is the collection of RGB frames where each pixel is the sum of three primary color components. Each color varies from 0-255 respectively. If one color intensity seems to increase then other relative color value will decrease. Therefore, for creating the blood flow in the face green and blue values are varied. The green frame of the RGB frame is used to view the region of blood flow in the color magnification. NTSC (National Television System Committee) video color format is used in work that implements YUV color space which has three component. They are luma and two color difference values. Luma value denotes the brightness for each pixel and is used for black and white components. The color difference value are U= Blue minus luma and V= Red minus luma. YUV color space is similar to RGB and the values can be converted as shown in equations [37].

$= 0.299 + 0.587 + 0.114 \quad = 0.492(-) \quad = 0.877(-)$

$= \ + 1.140 \quad = \ - 0.395 - 0.581 \quad = \ + 2.032$

3.6 Blood Flow

As the thesis is related to the flow of blood in face, hand and palm and the technique for this is Eulerian motion magnification. Basically, it the facial artery that provides blood flow in the face. It becomes angular artery and flows to the at the nose along the forhead. From forehead, angular vein takes the de-oxygenated blood to the facial vein along the sides of nose. Finally, the facial vein takes the blood back to angular vein which takes the de-oxygenated blood away from the head back to heart [38]. Blood flow in hand starts from branchial artery which feeds the radial and ulnar arteries where these are joined at the deep palmar arch in the palm of the hand. The deep palmar arch branches into the palmar digital branches that goes down to the fingers. The structure of vein in hand is the dorsal venous network that takes the oxygenated blood back to the heart [38]. The dorsal network is on the back of hand and is composed of superficial veins that are close to skin surface. As there is considerable amount of blood flow near the surface of the skin, this color change is detected more readily.

3.7 Motion Interpolation

One of the simple way to insert interpolated frames between the existing ones is to slow down the video clips. Positive effect of this will be that it is easy to implement and its negative side would be it doesn't look good. The algorithm is to insert a frame$F_\mathrm{i}$between each pair of framesF_0 at time t_0 and F_1 at time t_1 given as follows:

$$t_\mathrm{i} = (t - t_0) f_{\mathrm{i,x,y}} = (1 - t_\mathrm{i}) F_{0,\mathrm{x,y}} + t_\mathrm{i} F_{1,\mathrm{x,y}} \tag{3.8}$$

Here in this case we only consider the pixel value and find linear interpolation between each frame. However, if we consider the motion of an object in the scene, it is not only to interpolate pixel values but also to gradually deform the image according to the optical flow. Optical flow is the movement of the object in a scene and is a vector field that for each visible point in the scene in the first frame tells us where the corresponding point is in the second frame.

Let us consider$V_{\mathrm{x,y}}$ be the optical flow field. For each pixel (x, y) it defines a vector with two elements,$[x, y]$. Then for each pixel (x, y) in the resulting image, flow vector $V_{\mathrm{x,y}}$ can be looked up. As we can say that this pixel "came from" a point that lies back along the vector $V_{\mathrm{x,y}}$ without too much loss of precision and will go to a point along the forward direction of the same vector. Since $V_{\mathrm{x,y}}$ is the vector from pixel (x, y) in the first frame to the corresponding pixel in the second frame, back coordinates can be found $[x_\mathrm{b}, y_\mathrm{b}]$and "forward coordinates"$[x_\mathrm{f}, y_\mathrm{f}]$ which are then used for interpolation.

$$t_\mathrm{i} = (t - t_0/(t_1 - t_0)[x_\mathrm{b}, y_\mathrm{b}] = [x, y] - t_\mathrm{i} V_{\mathrm{x,y}} \tag{3.9}$$

The point (x,y) in the interpolated frame comes from a point in the first frame that lies along the line f(u)=[x,y]-$u_{\mathrm{x,y}}$ where u denotes ti that varies from 0 to 1 as we move from first frame to second frame. The point [x_b,y_b] is a coordinate pair that can be interpreted as the point we get when we move against the optical flow from [x, y] a distance proportional to the time that has passed since the first frame.

$$[x_\mathrm{f}, y_\mathrm{f}] = [x, y] + (1 - t_\mathrm{i}) V_{\mathrm{x,y}} \tag{3.10}$$

Similarly, $[x_\mathrm{f}, y_\mathrm{f}]$ represents the point we arrive if we move along the optical flow, a distance proportional to the time left to the second frame.

$$F_{\mathrm{i,x,y}} = (1 - t_\mathrm{i}) F_{0,\mathrm{xb,yb}} + t_\mathrm{i} F_{1,\mathrm{xf,yf}} \tag{3.11}$$

After the motion estimation, it is necessary to determine the optical flow. For this purpose, we first divide the first frame into the square block of size s, and try to match it against the second frame at every location within the distance d in the second frame. Thus, for each point [x,y], cut out a small square $[x, y] - [x + s, y + s]$ for each point in the region $[x - d, y - d] - [x + d, y + d]$ add the sum of squares of the difference between the small square and the second frame. The pseudo code for optical flow look like this:

```
block = crop (firstFrame, x, y, x+s, y+s);
bestSum = \texttt{MAX\_FLOAT};
bestPosition = [x,y];
for (int dx = -d; dx < d; ++dx) {
    for (int dy = -d; dy < d; ++dy){
    correspondingBlock = crop (secondFrame, x+dx, y+dy, x+s+dx, y+s+dy);
       difference = subtract (block, correspondingBlock);
   sum = 0
       for each pixel p in difference:
       sum = sum + sqr (p)
      if (sum < bestSum)
          bestSum = sum
           bestPosition = [x+dx,y+dy]
   }
}
opticalFlow (x,y) = bestPosition - [x,y]
```

3.8 Motion Model in OpenCV

OpenCv (open source computer vision) is a built in library that includes the library of programming functions and machince software learning library for computer vision applications and to accelerate the use of machine perceptron in the commercial product. The library contains more than 2500 optimized algorithms that can be used to detect and recognize the faces, identify the object, classify human actions in videos, track moving objects and in many more commercial applications. The interface includes C, C++, Python, Java and MATLAB and supports Windows, Linux, Mac Os. It is written in c++ and has a templated interface to work with STL containers. Different tutorial and its documentations are available in its official websites and other blogs to help the users with different configuration problem and other errors. Opencv motion model includes

1. Translation(`MOTION_TRANSLATION`): The first image is translated by(x,y) coordinates to obtain the second image. For this model, we need to estimate two parameters i.e. (x and y)
2. Euclidean(`MOTION_EUCLIDIAN`): The first image is rotated and shifted version of the second one. In this model, there are three parameters i.e. x, y and angle. When a square undergoes Euclidean transformation, the size does not change, right angle remains unchanged after transformation.
3. Affine(`MOTIONA_AFFINE`): This transformation is the combination of rotation, translation, scale, and shear. In this case if a square undergoes transformation parallel remains parallel but lines meeting at right angles no longer remain orthogonal.

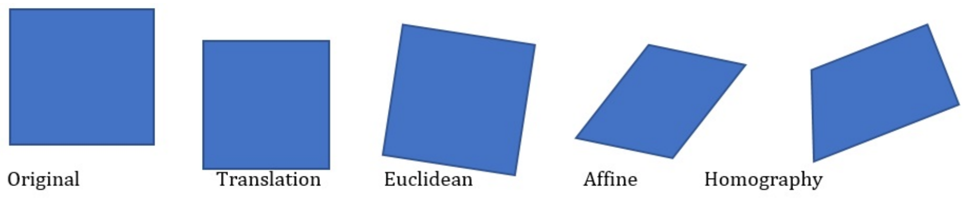


Figure 3.9: Image of square transformed by different motion models. [39]

4. Homography(`MOTION_HOMOGRAPHY`): This transformation is mainly focused on 3D effect whereas all above are 2D transforms. A square when transformed using this can change to quadrilateral.

In opencv an Affine transform is stored in 2x3 sized matrix where Translation and Euclidian transform are special cases of the affine transform. In the translation, rotation, scale and shear parameters are zero, but in Euclidian transform scale and shear parameters are zero. Therefore, it is necessary that translation and Euclidean transform are also stored in 2x3 matrix. The images can be brought into alignment using the function *wrapAffine*, once the matrix is estimated. Homography is stored in 3x3 matrix. The image can be brought into alignment using warpPersepective function once the homography is estimated.

Image Registration using Enhanced Correlation Coefficient(ECC) Maximization.ECC is an image alignment algorithm that was introduced in opencv3 on 2008 by George D. Ecangelidis and Emmanouil Z. Psarakis in the paper titled "Parametric Image alignment using Enhanced Correlation Coeffient"[40]. Advantage of this model are it is invariant to photometric distortions in contrast and brightness and the iterative scheme to solve the problem is linear though the objective function is nonlinear function of the parameters.

Red, blue and green channels in an image are not as strongly correlated, however human eye can detect the scene that are more strongly correlated in the gradient domain. Even though the intensities may be different in the three channels, the edge map generated by object and color boundaries are consistent. The image gradient feature from opencv can be implemented to further enhance the result of the output frames in the program in order to sharpen the edges of the overlapped images before creating the output video. Following steps are done to implement the image gradient in the program.[40]

1. Read 8 bit color image in which the three channels are concatenated vertically.
2. Find the height and width of color image.
3. Extract the three channels from the gray scale image
4. Merge three channels into one color image
5. Set space for aligned image
6. Define the motion model
7. Set the space for wrap matrix
8. Set the wrap matrix to identity

9. Wrap the blue and green channel to red channel and call the gradient function
10. Calculate the x and y gradients using Sobel operator
11. Combine the two gradients and return the value to main program
12. Use the Perspective wrap when the transformation is a *Homography*
13. Use warpAffine when the transformation is not *Homography*
14. Merge the three channels
15. Display the final output

3.9 Image Registration

Image registration is the process of transforming sets of data into one coordinate system or simply aligning two or more images from the same scene which involves designating the image as reference i.e. fixed image and applying geometric transformations to the other images so that they are aligned with the reference image[41]. Misaligning of the images can be due to various reason. Usually, the images are captured under variable conditions that can change the camera perspective. It can also be the result of lens and sensor distortions or difference between the capture device.

Image registration is often used as a preliminary step in other image processing applications that is necessary to compare or integrate the data obtained from different measurements. Spatial domain method works in the image domain that match the intensity pattern or features in images[41]. Some of the features matching algorithm are outgrowths of traditional techniques for performing manual image registration where an operator chooses corresponding control points(CP) in images. When the number of control points exceeds the minimum required to define the appropriate transformation model, iterative algorithm can be used to robustly estimate the parameters of a particular transformation types for the registration of the image. But frequency domain methods find the transformation parameters for registration of the images while working in the transfer domain. And applying the phase correlation method to a pair of images produces a third image which contain single peak. The location of the peak corresponds to the relative translation between the images. But phase correlation method is resilient to noise, occlusions and other defects typical of medical or satellite images. In addition to that the phase correlation uses Fast Fourier Transform to compute the cross correlation between the two images resulting in large performance gain. Due to the properties of Fourier transform [23], the rotation and scaling parameter can be determined in a manner invariant to translation.

There is a level of uncertainty associated with registering images that have any spatio-temporal differences. A confident registration with a measure of uncertainty is critical for many change detection applications such as medical diagnostics.[41][24] In remote sensing applications where a digital image pixel may represent several kilometers of spatial distance (such as NASA's LANDSAT imagery), an uncertain image registration can mean that a solution could be several kilometers from ground truth. Several notable papers have attempted to quantify uncertainty

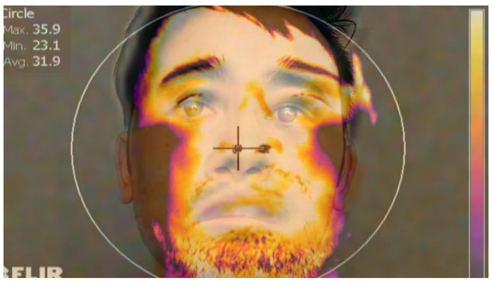


Figure 3.10: Left figure is final result without considering control point for image registration(QT-framework)

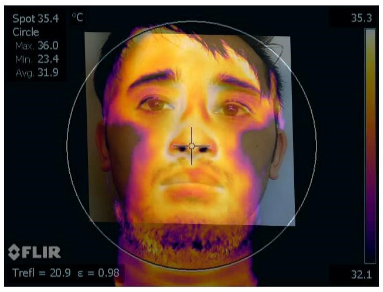


Figure 3.11: right one is with control point registration for an image(MATLAB).

in image registration in order to compare results. However, many approaches to quantifying uncertainty or estimating deformations are computationally intensive or are only applicable to limited sets of spatial transformations [41].

Image registration has applications in remote sensing (cartography updating), and computer vision. Due to the vast applications to which image registration can be applied, it is impossible to develop a general method that is optimized for all uses.

Medical image registration (for data of the same patient taken at different points in time such as change detection or tumor monitoring) often additionally involves elastic (also known as nonrigid) registration to cope with deformation of the subject (due to breathing, anatomical changes, and so forth). Nonrigid registration of medical images can also be used to register a patient's data to an anatomical atlas, such as the Talairach atlas for neuroimaging.

- The Image Processing Toolbox™ and Computer Vision System Toolbox™ offer three image registration solutions:[42]
- Intensity-Based Automatic Image Registration maps certain pixels in each image to the same location based on relative intensity patterns. This approach is best suited for workflows that involve a large collection of images or when you require an automated workflow. This functionality resides in the Image Processing Toolbox.[42]
- Control Point Registration allows you to manually select common features in each image to map to the same pixel location. This method of registration is best suited for images that have distinct features. It resides in the Image Processing Toolbox.[42]
- An automated feature-based workflow automatically aligns images by selecting matching features between two images. This workflow includes feature detection, extraction, and matching, followed by transform estimation. Features can be corners or blobs and the distortion can include rotation and scale changes.[41]

Figure below show the image when the control point are selected for an image with correct stretching and on with error if the control point are not selected:

3.10 Gradient Domain image processing

Image gradient refers to a directional change in intensity or color in an image that can be used to extract information form images. Mathematically, the gradient of two variable function at each image point is a 2D vector with the components given by the derivatives in vertical and horizontal direction. At each image point, the gradient vector points in the direction of largest possible intensity increase, and the length of gradient vector corresponds to the rate of change in that direction. Since the intensity function of a digital image is only at discrete points, derivatives of this function cannot be defined unless we suppose that there is underlying continuous intensity function which has been sampled at the image points. With some additional assumptions, the derivatives of the continuous intensity function can be computed as a function on the sampled intensity function i.e. digital image. Approximation of these derivatives functions can be defined at varying degrees of accuracy. The most common way to approximate the image gradient is to convolve an image with a kernel, such as Sobel operator or Prewitt operator. The gradient of an image is one of the most basic building block in image processing where it is often utilized in maps and other visual representation of data in order to convey additional information.[43] It is a type of digital image processing that operates on difference between neighbouring pixels, rather than a pixel values directly. Mathematically, an image gradient represents the derivative of an image so the goal of gradient domain processing is to construct a new image by integrating the gradient which requires solving Poisson's equation. Processing images in the gradient domain involves two step processes where the first is to choose image gradient i.e. often extracted from one or more images and then modified. And the second step involves solving Poisson's equation to find new image that can produce gradient from the first step. It can further be extended to moving images by considering the video clip to be a cube of pixels and solving 3d Poisson equation. [43] Image gradients can be used to extract information from images. Gradient images are created from the original image (generally by convolving with a filter, one of the simplest being the Sobel filter) for this purpose. Each pixel of a gradient image measures the change in intensity of that same point in the original image, in a given direction. To get the full range of direction, gradient images in the x and y directions are computed. One of the most common uses is in edge detection. After gradient images have been computed, pixels with large gradient values become possible edge pixels. The pixels with the largest gradient values in the direction of the gradient become edge pixels, and edges may be traced in the direction perpendicular to the gradient direction. One example of an edge detection algorithm that uses gradients is the Canny edge detector.[44] Image gradients can also be used for robust feature and texture matching. Different lighting or camera properties can cause two images of the same scene to have drastically different pixel values. This can cause matching algorithms to fail to match very similar or identical features. One way to solve this is to compute texture or feature signatures based on gradient images computed from the original images. These gradients are less susceptible to lighting and camera changes, so matching errors are reduced. The gradient of an image is given by the formula:

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{df}{dx} \\ \frac{df}{dy} \end{bmatrix}$$

where, $\frac{df}{dx}$ is the gradient in the x direction and $\frac{df}{dy}$ is the gradient in y direction.The gradient direction can be calculated by the formula:

$\theta = tan^{-1}(gy/gx)$ one dimensional filter to image A by convolution is applied to calculate df/dy,

$\frac{df}{dy} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} * A$ * denotes one dimensional convolution operation where 2x1 filter shift the image by half pixel. Therefore, 3x1 filter can be used $\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$

Chapter 4

A state-of-the-art investigation

A post-processing technique is introduced to improve the Eulerian video magnification method, which is a state-of-the-art motion magnification method to manipulate small movements in videos based on spatio-temporal filtering. The proposed method uses the Eulerian video magnification as a video spatio-temporal motion analyser to get the pixel-level motion mapping. Then the input video pixels are wrapped based-on this mapping to amplify the motion. This processing does not involve pixel value modifying, which makes it supports larger amplification and is significantly less influenced by the frame noise.

"Two-Frame Motion Estimation Based on Polynomial Expansion" is an approach that has been applied in different algorithm to change the video frame rates. It works on the polynomial expansion by approximating each neighbourhood pixel of both frames by quadratic polynomials. Evaluation on Yosemite sequence [45] shows the good result. And for frames overlapping many techniques have been implemented. However, for the project Butterflow algorithm for frame interpolation is implemented and Linear blending techniques for overlapping the video sequences.

Chapter 5

Approach

Since the problem is to work with the videos from both the visible and thermal camera it is quite challenging task as both the cameras produce videos but each of them is different. It is because visible camera works on reflection of the light or optical flow algorithm while that of thermal camera works on temperature of the object.

Each of the algorithm further implements different filters to process the image correctly. Similarly, it is different in case of thermal camera as it is sensitive to heat. Slight change in temperature will fluctuate the video color of the object. It is also important to consider the interpolation algorithm if we should work with both the images simultaneously. Therefore, for this also we have considered some specific interpolation algorithm.

According to the specified task for thesis, we first choose the method to work with the regular video where the method EVM is applied on videos. First re-implementation is done for example videos provided in the site[9] where the code is in MATLAB. After examining the result, we re-apply it to our examples that are taken for face, palm, hand and palm.

Next, we choose method to interpolate the video frames as the main goal of the thesis work is to observe the combined result of visible and thermal video. Therefore, we cannot get the desired result if you directly try to overlap the frames from both the videos as thermal camera works at lower frame rate. Because of that the frame rates cannot be matched and we cannot obtain the result. So, we choose “Butterflow algorithm” by Duong Pham[18] for solving this problem to obtain the desired frame rate that can be exactly matched with the frame rate of the regular video.

The Butterflow is using OpenCV library that uses ”dense optical flow in OpenCV”. Opencv is opensource library for most of the image processing task. Most of the task of this thesis paper are based on the algorithm in OpenCV documentation [46]. The algorithm used by OpenCV for video frame interpolation was devised by Gunner Farneback ”Two-Frame Motion Estimation Based on Polynomial Expansion” [5]. Butterflow algorithm is also one of the opensource software for interpolating the video frames.

5.1 Methodology

The objective of the present work is divided into two different section where one task is to work with EVM to reveal temporal variation in videos that are difficult or impossible to observe with naked eyes and display them in indicative manner. Spatial decomposition followed by temporal filtering is applied to the input video sequence and then amplified to reveal the hidden information in the resulting signal. The result was generated using non-optimized MATLAB code and the computation time per video was on the order of a few minutes. As it is easy to re-implement the code of mathematical package, So MATLAB was selected for the first section of the project. Although C++ is much faster than MATLAB, all the mathematical function for the code has to be re-written in c++. Therefore, for simplicity of the project work MATLAB was considered for the first part.

For the second part of the project, Butterflow algorithm is considered to interpolate the video sequence for the IR videos. When the video frames are interpolated, we create an application in Qt-framework for the user-friendly interface. In the application, we have made three segments of the output window where the first segment is for EVM video, middle segment for IR video and last segment for overlapping and displaying the final result from both the frames. In addition to that we have used the library from OpenCv [46], which provides all different types of image processing algorithm in advance. C++ is considered for overlapping the video sequence and making the easy user interface.

5.1.1 Library for parallelization

On the software side MATLAB, C++/OPENCV(that includes all the library for opencl and image processing) using QT-framework are used for the project. For the hardware we have implemented Video camera(1200D canon) for visible video and FLIR(ThermaCAM SC640*) for thermal video, whose spectral range is 7.5-13μm. The emissivity (ε=0.98), temperature reflexivity is 20^{0}C, object distance is 0.3m and atmospheric temperature was 22.3^{0}C. To conclude, EVM algorithm, Butterflow algorithm as the supportive method for the frame interpolation were selected for the work. Works during the project:

- MATLAB for re-implementation of EVM ,
- Msys64 for Butterflow algorithm implementation for frame interpolation
- QT/C++/OpenCV for the result of the project.

Using the MATLAB software provided by SIGGRAPH paper, The Eulerian motion magnification was implemented. There are two folders for data and result where input video is taken in data folder and generated result is stored in result folder. The processing started with a function call for each video recorded as follows:

```
Function=amplify_spatial_Gdown_temporal_ideal(vidFile,outDir,alpha,level,fl,fh,samplingRa
```

Where the function statement calls the color magnification code. Input arguments:

vidFile - input video file to process

outDir - the location to save the magnified output video files

alpha - the magnification factor which directly multiplies the filtered frames where the values are set from 90 to 500 and 500 for most of human skin video processing.

Level : Gaussian pyramid level for the spatial filtering where the value is varied from 0 with no spatial filter to 8 in the video processing. Most of the videos are processed using the levels 1-5.

fl, fh : lower and upper cuttoff frequency for the ideal temporal filter window respectively, so to include the pulse of human.

SamplingRate - the frame rate of the processed video which is set 30 Fps.

ChromAttenuation - The attenuation of the color in the magnification which is always set to 1 for the processing to maximize the color magnification.

5.1.2 Spatial Filtering

By using a Gaussian pyramid decomposition, spatial filtering is used and is implemented for all frames defined by *startIndex* and *endIndex* variables. The band of Gaussian pyramid is selected based on the level variable.

```
Gdown_stack= build_GDown_stack(vidFile, startIndex, endIndex, level);
```

First the RGB frames are converted to NTSC format using this function. Then the filter and down-sample operation is executed on each frame recursively from level 1 to the selected value. Selection of the spatial filter are from the list of wavelets provided in the namedFilter function. Filters for processing were Binomial 5 filter, Haar and the Daubechies 4 where Binomial 5 filter was the default filter and was used as below:

```
Kernel= sqrt(2) *BinomialFilter(5);
Function[kernel]= BinomialFilter(sz)
Kernel=[0.5,0.5]
For n=1:sz-2
Kernel=conv([0.5*0.5], kernel);
end
```

The Haar wavelets was implemented using code below where it filters the discontinuities in an image.The Daubechies 4 wavelet is low pass filter portion that smooth the discontinuity in an image.

First these kernels were convolved in x-direction and then in y-direction with 2D frame from the video using *conv2* from MATLAB with valid output size. The convolution output size is computed not to include any of the edge condition of the convolution as shown below:

```
Size= [width of frame- max(width of filter-1, 0), height of frame- max(height of filter-1
```

The edge condition in testing were reflection, extension, repetition and zero-padding. It is used to define the assumptions to occur at the edge of the frame during filtering when an operlap occurs between the filter and the edge of the frame. When the filter and edge condition are applied on the frame and the image is down sampled by a factor of 2 in each direction, the frame is the filtered and down sampled again. And after that they are passed to temporal filtering processing.

5.1.3 Temporal Filtering

The frames are temporally filtered in spatial filtering process. After spatial filtering, function is called for ideal-bandpass filtering as shown below.

```
Filtered_stack= ideal_bandpassing(Gdown_stack,1,fl,fh,samplingRate);
```

Where *Gdown_stack* is the spatial filtering output and the second argument 1 is input dimension for ideal bandpass filtering. Fl and fh are lower and upp cuttoff frequency on the filter. The Fourier transform of the input is taken on each pixel over time as the filtering is done in frequency domain. Frequency that lie outside fl and fh are set to zero and inverse Fourier transform is taken. Temporal spatial filter is the real portion of the output of inverse Fourier transform. Below code explains its implementations.

```
Function filtered= ideal_bandpassing(input, dim, wl,wh, samplingRate)
    Input_shifted= shiftdim(input, dim-1)
    Dimensions= size(input_shifted);
    n= Dimensions(1);
    dn= size(Dimensions, 2);
    Freq= 1:n;
    Freq=(Freq -1)/n*samplingRate;
    mask= Freq> wl & Freq<wh;
    Dimension(1) =1;
    mask =mask(:);
    mask =repmat(mask, Dimension);
    F= fft(input_shifted,[],1);
    F(~mask) =0;
    filtered =real(ifft(F,[],1));
    filtered = shiftdim(filtered, dn-(dim-1));
end
```

The desired portion of video is the output from ideal band pass filter which is then magnified.

5.1.4 Magnification

Chromatic attenuation is applied on the color portion of filtered output which is still in NTSC format while color magnification is used by multipliers on the filtered data. Therefore, the first component contains the black and white color information while remaining two defines the UV color space. Thus, scaled factor is added to second 2 components of YUV space to attenuate the color and chromatic attenuation is set to 1 to prevent attenuation. Now the temporal filter output is multiplied by the scale factor and alpha (the magnification factor).

After that the magnified output can be added to the original video frame by frame. The output increases the details visibility in the magnified video to find the differences between Gaussian pyramid levels and the pulse is more visible in the magnified filter output video.

Chapter 6

Experimental work for validation

6.1 Testing examples with few changes

Temporal bandpass filter is selected to pull out the motion or signals that needs to be amplified and the choice of the filter is dependent on the application. For color amplification of blood flow, a narrow passband produces a more noise free result. Thus, ideal bandpass filters are used for color amplification as they have passbands with sharp cut off frequencies. Low order IIR filter can be useful for both color amplification and motion magnification and are also convenient for a real-time implementation. In general we use two first order lowpass IIR filter with cut off frequencies ωl and ωc to construct an IIR bandpass filter. After that desired magnification value α and spatial cutoff frequency λc is selected. Using higher value of α violates the bound to exaggerate specific motions or color changes at the cost of increasing noise. This approach achieves the chrominance component of each frame by doing all the processing in the YIQ(Y represents luma information and I and Q chrominance) space. The chrominance component I and Q can be attenuated before conversion to the original color space. In order to boost the power of the specific signal, spatial characteristics of the signal can be used to estimate the spatial filter size. Let the noise power level be σ^2. Now we want to find a spatial low pass filter with radius r such that the signal power is greater than the noise in the filtered frequency region. The wavelength of the cut off frequency of such a filter is proportional to its radius r. So, the signal is S(r). The estimation of noise power σ^2 can be done by examining pixel values in a stable region of the scene by using technique in [Liu et al. 2006]. We can solve the following equation for r as the filtered noise σ^2 is inversely proportional to $r2$:

$$S(r) = \sigma^2 = k\sigma^2/r2 \tag{6.1}$$

Where k is a constant that depends on the shape of the low pass filter and this equation gives an estimate for the size of the spatial filter needed to reveal the signal at a certain noise power level.

Video		c	l(Hz)	h(Hz)	fs(Hz)
Face	10	16	0.4	3	30
Palm	10	80	0.4	3	30
Hand	10	80	0.4	3	30
Plant	60	90	306	6.2	30
Baby	10	16	0.4	3	30
Camera	120	20	45	100	300
Wrist	10	80	0.4	3	30
Face2	20	80	0.83	1	30
Face1	100	1000	0.83	1	30

Table of α, λ_c,ω_l,ω_h values used to produce the various outputs.[9]

The area of interest for revealing broad but subtle motion, temporal filters are used. As for in example for *face 2* video, second order IIR filter with slow roll-off regions is used to magnify the motion of the head rather than amplifying the change in skin color. But, our area of interest is change in color of the skin so we implemented same values of the face that is applied in the code. Figure for the frequency response of some of the temporal filters used in the project.

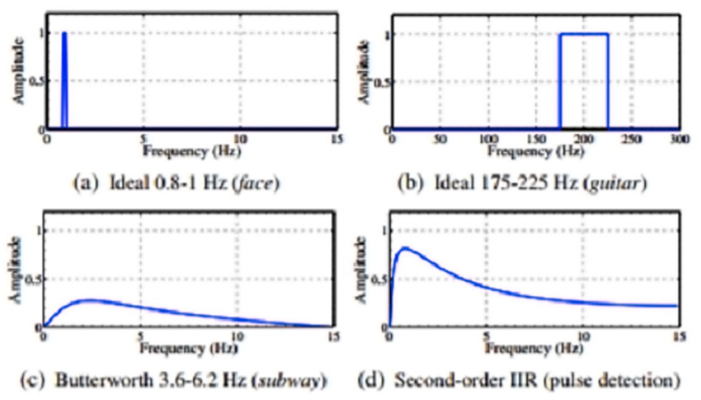


Figure 6.1: Temporal filters used. Ideal filters (a) and (b) are implemented us ing DCT(discrete cosine transform). Butterworth filter(c) is used to convert a user-specified frequency band to second order IIR structure and is used in our real-time application. The second order IIR filter(d) also allows user input. The second order filters have broader passband than ideal filter.[9]

Although we have obtained the peculiar motion of object in our video but our main objective was to amplify the color change as the blood flows through the face, hand and palm. And for the plant we want to check the subtle change in motion of the plant. For the face, we have applied Laplacian pyramid and set finest two levels to 0. At first, we down sample and apply spatial low pass filter to each frame to reduce both quantization and noise and to boost the subtle pulse signal that we are interested in. Then we pass each sequence of frame through ideal band pass filter with passband of 0.83 Hz to 1Hz (50 bpm to 60 bpm. And then a large value of (magnification value) α=100 and (spatial cut off frequency) λ_c1000 was applied to the resulting

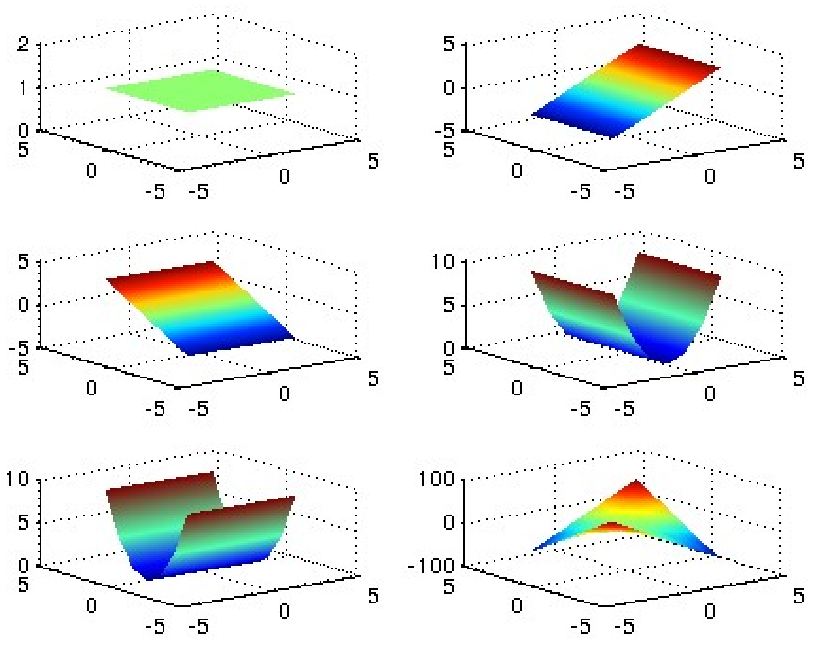


Figure 6.2: 2D basis function.

signal to emphasize the color change as much as possible. The final video is formed by adding this signal back to original.

6.2 Practical Implementation for Dense optical flow expansion based on Polynomial basis

BF algorithm is implemented on thesis work which implements the technique called dense optical flow in the program. Based on polynomial representation of an image we will observe the dense motion estimation and by approximating the local neighbourhood of an image using quadratic equation the polynomial basis of an image is obtained. By equating the coefficient, the basis displacement between two frames can be obtained. Using the same polynomial representation, estimation of dense optical flow is obtained at each point in the image. [47]

Let us consider polynomial transformation under translation using the polynomial expansion coefficient derived from current and previous frames[**?**]From that we obtain the estimate of displacement vector. Polynomial expansion idea is to approximate a neighbourhood pixel of a point in 2D function with a polynomial. Now let us consider the quadratic polynomials basis 1, x^2, y^2, x, y, -dy, where pixel values in neighbourhood of an image is represented by

$$f(x) = x^{\mathrm{T}} A x + b^{\mathrm{T}} x + c \tag{6.2}$$

Where A is a symmetric matrix, b is a vector and c is a scalar. Now the coeffiecient can be estimated by weight least square estimate of pixel values about neighbourhood. Brightness constant assumption is made with all optical flow algorithm i.e. brightness of an image in adjacent frame is constant. Considering the translation motion d at the point *f(x,y)* in an image.

$$f_1(x) = x^{\mathrm{T}} A_1 x (b_1{}^{\mathrm{T}} x + c_1 f_2(x) = f_1(x - d) = A_1 (x - d)^T (x - d) + b_1{}^{\mathrm{T}} (x - d) + c_1 f_2(x) = f_1(x - d) = x_{\mathrm{T}} A_1 x + (b_1 \tag{6.3}$$

Now equating the coefficient in two polynomials because of brightness constant assumption.

$$A_1 = A_2 b_2 = (b_1 - 2A_1 d) c_2 = d^{\mathrm{T}} A d - (b_1)^{\mathrm{T}} d + c_1 \tag{6.4}$$

Where A is non-singular

$$d = (-1/2) A^{-1}) (b_2 - b_1) \tag{6.5}$$

Therefore, the displacement vector can be obtained by equating the coefficient of the polynomials at each point in the image assuming there is overlap between the region of interest that is neighbourhood image in adjacent frames. Now lets consider we have an estimate of displacement $\bar{d}$ and we want to estimate region of interest about the neighbourhood at point P(x, y) and at point P(x+ d.x, y+ d.y). Following the above steps the polynomial basis are extracted and the computations are performed.

$$\bar{b}\ _2 = b\_1 - 2A\_1 \bar{d} \tag{6.6}$$

Since we have already found $\bar{d}$, A_1, b_1, total displacement can be estimated,

$$d = -0.5 * A^{-1} (b_2 + \bar{b}_2 - b_1) \tag{6.7}$$

Thus, with every successive iteration a better estimate of displacement vector can be obtained and the iteration terminate when the displacement vector is below threshold in successive iterations of specific number of iterations is completed. Initial displacement vector is (0,0) so that the image patch or region of interest in the current and previous frames are at the same. Following function is used for the computation optical flow displacement vector:

Thus, with every successive iteration a better estimate of displacement vector can be obtained and the iteration terminate when the displacement vector is below threshold in successive iterations of specific number of iterations is completed. Initial displacement vector is (0,0) so that the image patch or region of interest in the current and previous frames are at the same. Following function is used for the computation optical flow displacement vector:

*updatePoly(const float *ptr1,const float *ptr2,Point2f d,bool flag, float *M,Point p,Size s)*

Where *updatePoly* function computes the optical flow displacement vector, *ptr1* and *ptr2* are pointer to array containing the polynomial basis component, *d* for the estimation of displacement vector at the current point, flag for indicating if the point is border pixel or not, *M* a pointer to output array returning the computed coefficients, *p* a co-ordinate at the current point where the coefficient is evaluated and *s* for averaging the size of window. If the flag is on the average A_1 and A_2 is calculated for the $ptr1$ and $ptr2$ and then the process proceeds with the next step of computation which includes the calculation of $-(b1 - b2)$ using the pointer $ptr1$ and $ptr2$ from above. After that the sum for iterative estimation is done i.e.$b2 = b2 + \bar{b2}$. Finally, the final displacement is computed where $d = A^-1(b2 - b1) * 0.5$ In addition to that there is another function:

$AverageFlow(constMat\&\_\_R0, constMat\&\_\_R1, Mat\&\_\_flow, Mat\&matM)$

Where $R0$ is the polynomial basis coefficient for previous frame, $R1$ is polynomial basis coefficients of current frame, flow for estimate of current displacement field and $matM$ contains the coefficients of polynomial basis for computing displacement field.

In this function also computation of average flow field is done using the OpenCv library $cv :: GaussianBlur(.....)$ and updating the borders. After that the determinant for inverse is calculated to computer the displacement flow field. The method EstimateFlow in the program computes the coefficient $A, b2, b1$ for displacement field computation which calls method UpdatePoly for each pixel in an image. However, the displacement field obtained may be discontinuous and contain noise and other artifacts. For the displacement vector can be averaged over a neighbourhood to get a better estimate of the displacement field. The following method computes the average of $A1, b2, b1$ and computes the displacement flow field.

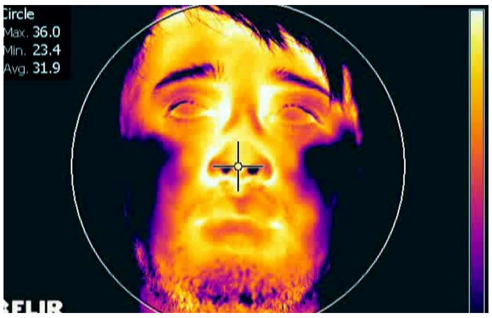


Figure 6.3: First frame

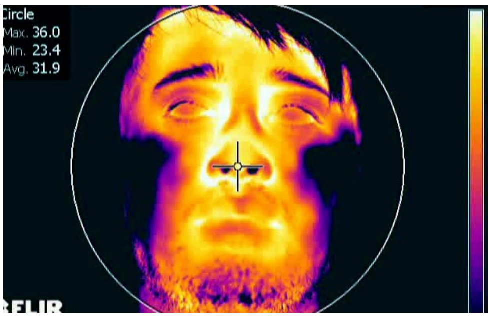


Figure 6.4: second frame

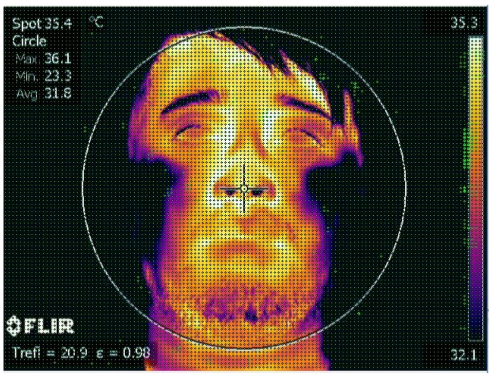


Figure 6.5: Result of Dense optical flow

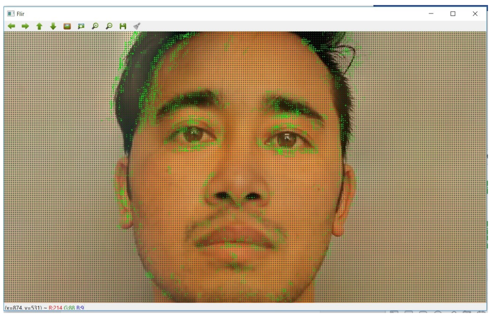


Figure 6.6: Dense optical flow in thermal image between two image frames and also in regular image between two frames . Green dot represents the dense optical flow. And the green color with darker portion denotes more optical flow field.

Chapter 7

Structure and Implementation in single modules

7.1 Application

First, the design of the application was created so that It can be user friendly. There are three segments in the output where user can see the Eulerian video output, Flir video output and the result of the project. The result includes the overlapping the two videos sources from the normal video camera and video from Flir camera. During the overlapping of the video, windows resizing function was used so that both the videos can be overlapped over one another.

Application is a Qt widgets application. Animation is not provided as it requires each step of algorithm and all the effect cannot be shown in the video. This is due to the reason that its not only the single frames we are working with. Following figure shows the structure of an application for this thesis project.

7.2 Extension of video analysis

Video image processing has become an important task this days with the growing technology in computer vision application. OpenCV is the library that can be used in any framework contains many different supportive files for the task of image processing. It supports GUI library(Highgui) i.e. provides the good user interface to interact with real time system in many different applications. However, the library has no support for buttons and other designing task,

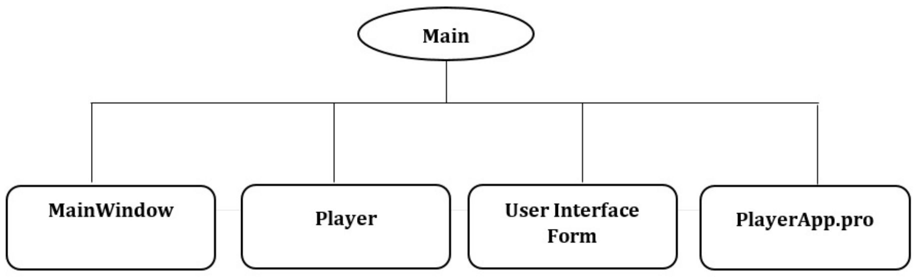


Figure 7.1: Main structure application

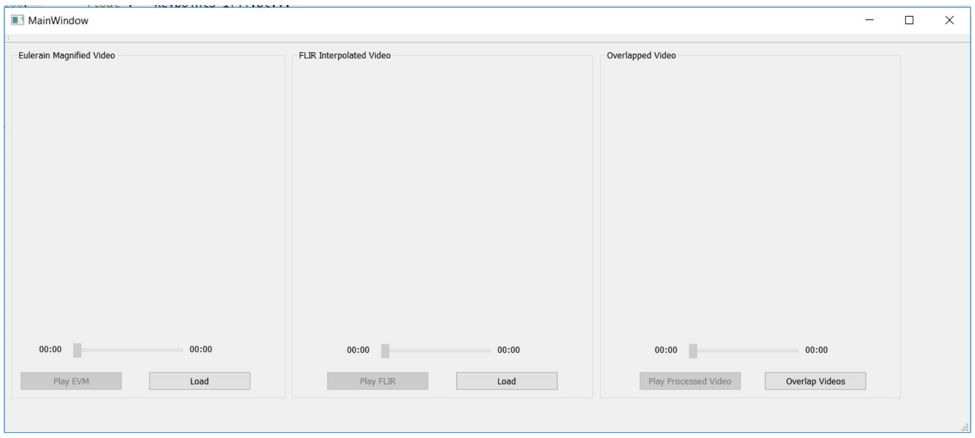


Figure 7.2: Design of the project

so it is preferable to use a QT GUI application, but displaying the video in QT GUI is not spontaneous as it is with opencv library.

For the extension of video analysis work we have created an application that could process both the videos from thermal camera and from the visible camera and display as a one overlapped video sequence. At first OpenCV was configured in the system and also in Qt widget application. For this purpose, we have used cmake gui application, so that the library gets configured in the system. At first the design was created for easy user interface. Prototype of the design looks like as in the figure 7.2.

As shown in the figure the main window is segmented so that all the videos can be loaded to see the outcome of the project. The first segment will first load the video for Eulerian magnified video, the second one will load the thermal infrared video and both the videos are overlapped in the third section. Each section is in their specified group box so that it is not mixed up with one another. The project is designed in QT framework as it provides all the necessary widgets for developing the easy user interface.

7.3 Correlation between both color and temperature over time

Human body has certain temperature that is very useful in diagnosing different kinds of diseases. As the infrared camera work on temperature it is much likely to measure the human body at different temperature and it is normal that the body temperature is effected by the change in temperature of the environment. It can also be used for non-invasive diagnosis of peripheral vascular disorders if the temperature gradients are observed at effective regions as an abnormal blood flow will be seen in the affected regions. Thus, thermal imaging technique is an effective technique for detecting small temperature change in the human body due to vascular disorders. Video for thermal camera were taken at different temperatures as shown in table below:

Object	Min. temperature-Max. temperature(^{0}C)
Face	32.1-35.3, 33.3-35.3,31.6-36.6,30.4-35.4
Hand	31.7-33.8, 32.9-35.0, 29.9-35.0
Palm/wrist	32.5-34.6,31.8-33.9, 30.5-35.4
Plant	22.9-25.0, 22.0-24.1

Table showing the object at minimum and maximum temperature

As correlation is the interdependence of variable quantities so from the figure 6.2 and 6.3, it can be observed that there is a strong correlation between both color and temperature. At the lower temperature, the object color is darker relative to higher temperature as shown in the figure. At higher temperature, the color changes from blue to red to yellow and finally white. If there is white color in thermal image means that the object is at the highest temperature. For the experiment, we have taken different videos at different temperature which can be seen in table above. Below are few examples that have been implemented in the project:

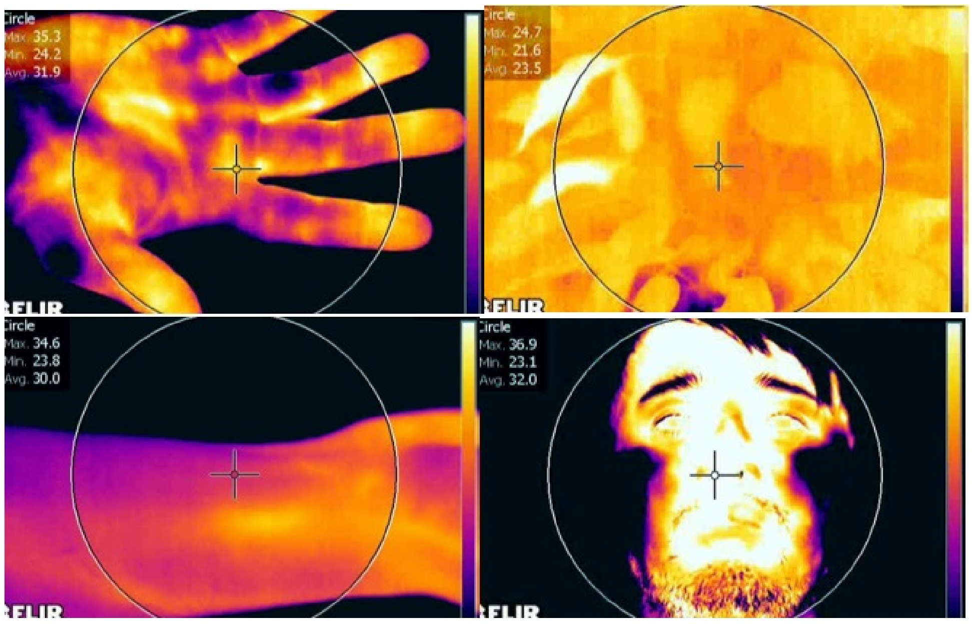


Figure 7.3: hand, plant, palm frames from thermal camera at different temperature where side-bar show the temperature range from coldest(black) to hottest(white)

Color in thermal imaging changes from dark black to white where white color in the picture denotes the maximum heat. As you can see in the figure 7.3 of hand, there is difference in color in all over the hand. Variation in color denotes variation in temperature due to blood flow in the hand (purple and dark color denotes it is at lower temperature and that of yellow and white denotes higher temperature.

7.4 Classes, namespace and structures

Instead of creating many modules, we have created an interface that is easy for the user to understand. The program source code is available in `https://source.uit.no`. we have created many different methods under player and MainWindow class to work simultaneously with GUI interface.

7.4.1 MATLAB for re-implementation of video magnification

The main idea of Eulerian Video magnification is to independently process the time series of color values at each pixel and is done by applying standard 1D temporal signal processing to each time series and amplify a band of temporal frequency. But the result has been limited because such a processing cannot handle general spatial phenomena such as large motions which involves complicated space time behaviour across pixel. Eulerian processing can approximate their amplification as the motion involved are small and first order Taylor series can be implemented to show linear, per pixel amplification of color variation.Formulation of Eulerian processing on special case of 1D translational motion of a diffuse object under constant lighting which can also be applied for arbitrary phenomena such as 3D motion and shiny objects.

For better output that handles better motion and less prone to noise Fourier series was introduced where complex-valued sine functions can be translated by shifting their phase. To break the limit of Fourier series to handle the same translation across the entire frame spatially localized sine functions are implemented by wavelet like representation called complex steerable pyramid. This representation decomposes image into a sum of complex wavelets corresponding to different scales, orientation and positions where each wavelet is like the amplitude and phase of complex sine functions. The complex sine functions provide locality in frequency while windowing provides locality in space. Each basis function is complex, consisting of a real, even-symmetric part (cosine) and an imaginary, odd-symmetric part (sine). This gives rise to a notion of local amplitude and local phase as opposed to the global amplitude and phase of Fourier basis functions. We use only a half-circle of orientations because basis functions at antipodal orientations (θ, $\theta + \pi$) yield redundant, conjugate coefficients.

7.4.2 Class of the PlayerApp

The PlayerApp consists of all the classes. First of all, it contains *PlayerApp.pro* file where all the necessary *.dll library files are loaded. That is the library for OpenCV and the library for path. This file contains all the supportive files for video image processing. For OpenCV also we need to do extra set up that includes the extra modules from OpenCV. Therefore, all necessary header files from OpenCV are included before using it in the main program. There are two main classes that holds all the information and making of an app. First is main window class and the second is Player class. And main program holds the interface to connect both the class. In MainWindow class there is a namespace for main window and *Q_OBJECT*. It contains private slots to connect with the main window user interface that has push button for loading and playing the video files and the scroller to see the time frame for the video. This class implements the function to set and get all the necessary values for video processing. In the private member of this class we initialize pointer to player class and to user interface class. Player class contains vectors and matrices for video frames and the methods to manipulate the media player. This class also is responsible for getting and setting the video frames.

Player class inherits from the QThread class which allow it to run on its own thread so that the main window remains responsive while the video is still playing. Without this the video will cause the screen to freeze until it has finished playing. The signal function *processedImage(...)* will be used to get the output video frames to the main window.

LoadVideo() method uses videoCapture class to load the video and set the frame rate which is from the opencv library. The public method *play()* calls the *run()* method to start the thread which overrides QThread run method.Running the while loop in *run()* method to play the video after reading the frames which is then converted into QImage. Then same QImage is emitted to the mainaWindow object using the *processedImage(....)* signal. When the loop terminates, waiting for number of milliseconds is done, which is calculated using the frame rate of the video. In the destructor of player, videocapture object is released and waiting for run method to exit is done.

Now initializing myPlayer and connecting the signal emitted from player class to update the *playerUI()* slot so that every frame emitted would pass the slot. Code can be found `https://source.uit.no` for the player application. *UpdatePlayerUI* slot receives a QImage and resize it to fit the aspect ratio for displaying Where it displays the image by setting the

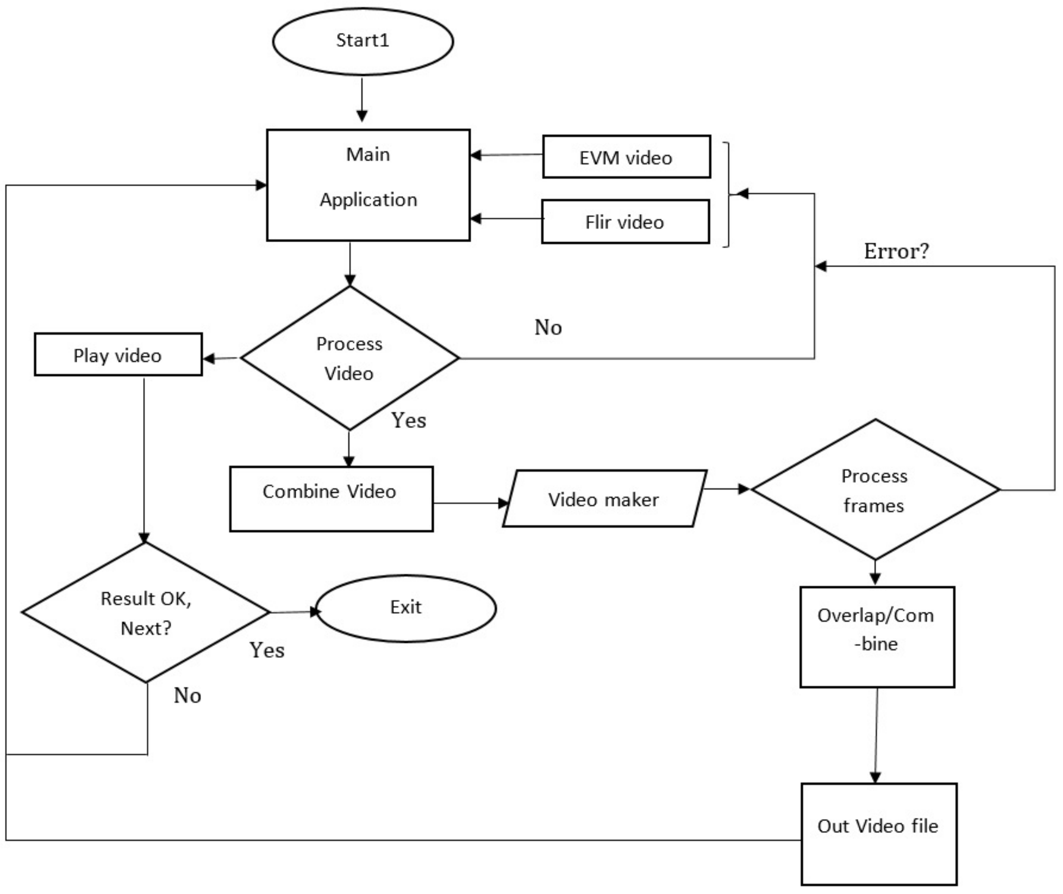


Figure 7.4: Flowchart of Main application

label pixmap. Finally , an instance for mainwindow class is created and set the delete on close attribute so that the object created are destroyed in the main function. In addition to that the horizontal slider class is added to see the time frame of the video sequence.

7.5 Problem while overlapping

Visible images are more sharper and clearer than infrared images as visible sensor arrays can be made with smaller detector elements and with far greater number of elements and also that it is not used to measure the temperature i.e. the images can be generated with only reflected radiations. Visible detector arrays have millions of elements but that of infrared detector arrays have far fewer so it is clear that visible images are more detailed than infrared images. In addition to that the visible images can be displayed in the same colors, shades and intensities as that seen by human eye so their structure and character are more interpreted more easily than infrared images. Visible images are almost produced by reflected visible light and that of infrared images used to measure temperature must record emitted infrared radiation. Reflected visible radiation can produce sharp contrast with sharp edges and intensity difference.

Sharp infrared reflection contrast is possible by using the surface of low emissivity (high infrared reflectance) next to the surface of high emissivity (low infrared reflectance) [48]. However, it is not usual to have surface with sharp temperature difference next to each other. Heat transfer between close objects can make failure in temperature difference by producing temperature gradients making it difficult to produce images of emitted radiation with sharp edges. This is the reason why infrared images that are used to measure temperatures are usually less sharp compared to visible images. To combine both the images side by side images should be taken simultaneously, but the spatial correlation suffer from parallax. It works fine for long distance where parallax is negligible but for the short to moderate distance, parallax is the issue.

7.5.1 Parallax

Displacement or difference in the apparent position of an object viewed along two different sight is called parallax. It is the phenomena that arises due to change in viewpoint because of the motion of the observer to that of observed where relative motion is essential. It is measured by the angle or semi angle of inclination between those two lines. Those objects that are near have more parallax than more distant objects when observed from different viewpoints, therefore it can be useful to determine the distances. Human including all other animals with two eyes uses parallax to gain depth perception estimate distance to objects by overlapping the visual fields from two eyes. Animals also use motion parallax, in which the animals move to gain different viewpoints. There is a device called parallax rangefinder that is used to find the range. One can determine distance by observing parallax, measuring angles and using geometry.[48]

7.5.2 Blending Visible and Infrared Images

Infrared video is taken from Flir SC640 12° while visible video is taken from canon 1200D at different instances so the result is somehow not automatically registered as there was variation in distance from the object and the camera and also the resolution of the visible camera was at 1280X720 pixel and that of flir was at 640X480 pixels. As the matter of fact that both the camera distance between the object and image should be at the same to match the visible image with the infrared image, so that the result obtained can overlay each other on the camera display. Infrared images and visible images are matched pixel by pixel where operator can easily identify the location of infrared point of interest on the target by noting where the features are in the blended images. The location of an infrared feature of interest can be precisely identified even if the infrared contrast is low and there is very little structure in the infrared image with the blending images. The result that is obtained is a bit shifted because of the distance between the object and image in both the camera. For the exact overlapping video between the visible and thermal following diagram should be taken in to consideration:

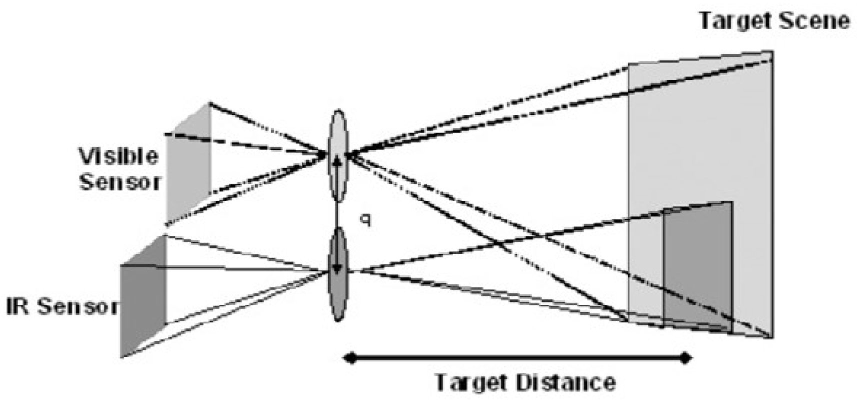


Figure 7.5: Optical path and sensor configuration [48]

There are two sensors and two distinct optical path that is considered for visible and thermal infrared camera. Both the sensors are parallel to each other with the target distance same for both. Both the sensor will see the target scene from slightly different view causing parallax because the optical path for the sensors are different. Visible optics remains in focus at all usable distances but that of infrared has low f-number and because of that a shallow depth of field which provides a means of determining distance of the target. So, infrared lens need to focus on adjustment for targets at different distances.

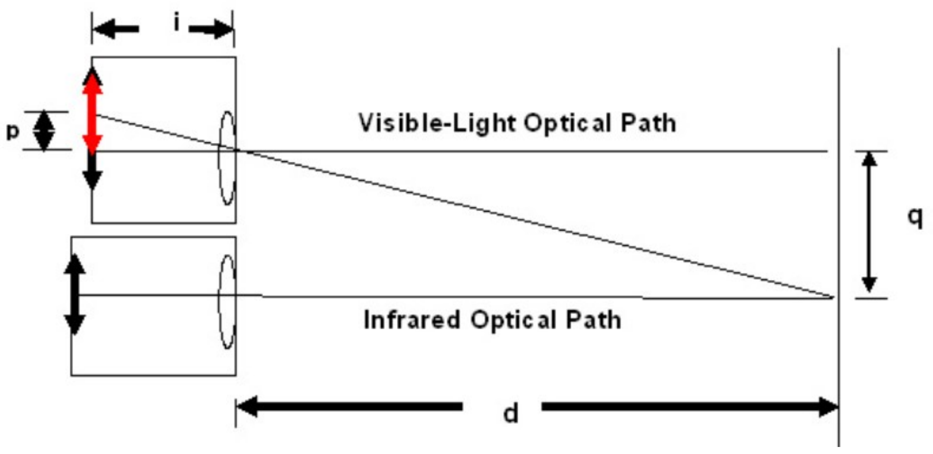


Figure 7.6: Parallax geometry [48]

7.5.3 Correction of Parallax

Based on the infrared camera focus distance parallax correction is done.

The figure shows the parallax geometry where the standard lens equations is given as

- d= distance to object
- i= distance to image
- f= effective focal length of lens
- q= separation distance between visible and infrared axis
- p= image offset at the visible focal plane

The distance between the two-sensor q and the lens focal length f are fixed. $q\ d$ is infrared optical path and $p\ I\ d$ is visible light optical path. When the video is captured from both infrared and visible camera sensors it is first saved and the sequence of image frames from the videos are taken for both the camera. And after that linear blending technique is implemented in order to see the overlapped images to create the sequence of image frames and again the video is created. Furthermore, the image registration technique is implemented for the result image in order to obtain the sharper image sequences in MATLAB.

7.6 Further Improvement

Local phase difference can be used to manipulate local motions where we take an image sequence, takes each frame into the complex steerable pyramid basis and then independently amplify the phase difference between all the corresponding basis elements. This is identical to linear amplification processing except that the representation from intensities to local spatial phase. Amplifying phase differences rather than pixel intensity differences has two main advantages: (a) it can support larger amplification factors, and (b) noise amplitude does not get amplified. The magnification of video can be slow using the complex steerable pyramid because the representation is much larger than input.

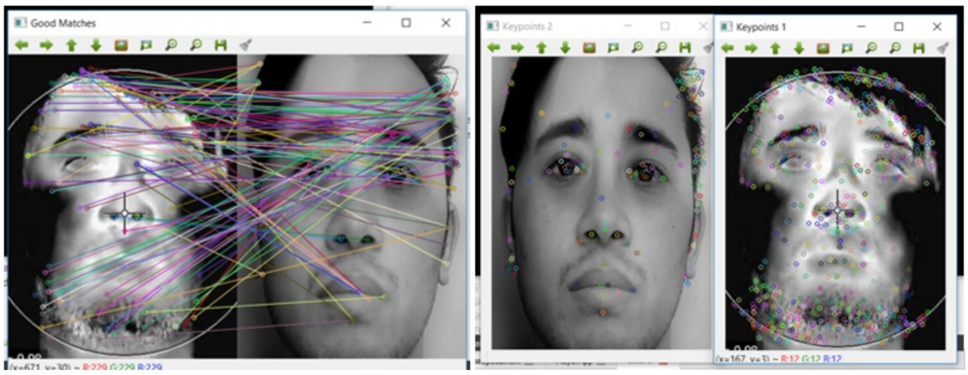


Figure 7.7: Result of Feature extraction.

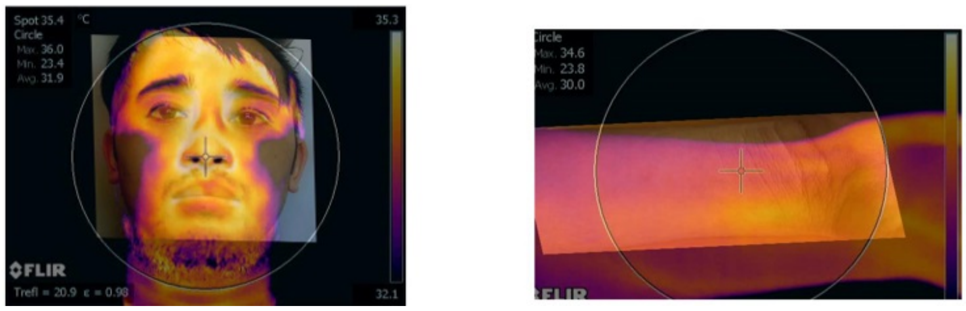


Figure 7.8: Result from MATLAB after the control point using image registration technique (using image registration toolbox).

7.6.1 Feature Extractions

Feature extraction has an important issue while detecting the basic features such as eyes, nose and mouth in the face. It also gives important information for distinguishing faces of different persons and stable with respect to the geometrical and photometrical variations. In thermal images, facial texture is localized around facial features and locating features such as eyes, nose and mouth position using interest point extraction. An interest point can be defined as any point in an image for which the signal or intensity changes two dimensionally and the corners as a point where there is two dominant and different edge direction in local neighbourhood of the point.

Output from feature implementing algorithm from OpenCV documentation is shown in figure 7.6. Due to the phenomena called parallex the feature extraction also could not find the exact point in an image. So for the image to find the exact match, both the image should be taken parallely

7.6.2 Result From MATLAB

This is the final output image from MATLAB if the control points are selected using image registration technique. So as an alternative to application in QT-framework using, MATLAB can also be used to overlap the image using image registration technique.

Chapter 8

Result

The result is the overlapping of the output from EVM for regular videos and Butterflow algorithm for IR-thermal videos. Result after the overlapping of image in the main application is shown in figure below:

Hand, face and wrist were chosen to evaluate the performance of EVM to find the blood flow by observing the redness in the color of skin. But a plant was only chosen to see the peculiar motion within few seconds of time. The subject pulse was measured to configure temporal filtering after the test and like that default values were used for spatial filtering. The video was taken at the distance of 0.3m for the flir camera but in case of regular camera it was not considered. For filming face, it was normal as sitting in chair and that of wrist and hand were placed on a table to prevent the motion. There was a bit motion in face as there were no support for it. The video was recorded many times before a conclusive result were found to determine what setup would produce most clear result to determine region of blood flow. There are also motion artifacts such as breathing, eye motion and muscle twitches in the faces that shows up as region of high color variation. Due to this it makes EVM to determine which region of high color variation are from blood and which are from motion. Figure show the change in skin color after applying EVM on the regular video. Redness in each image frame is varying either because of the motion or due to the movement. The result is the outcome of this

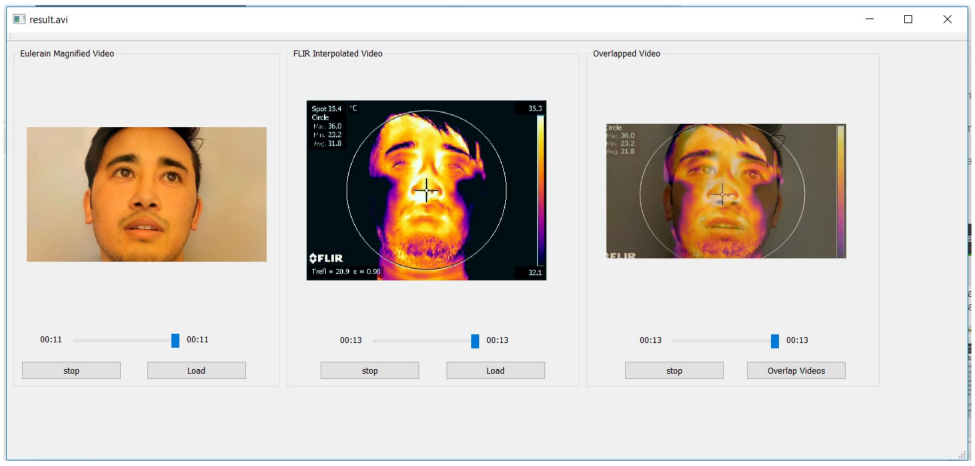


Figure 8.1: Result of face overlapping

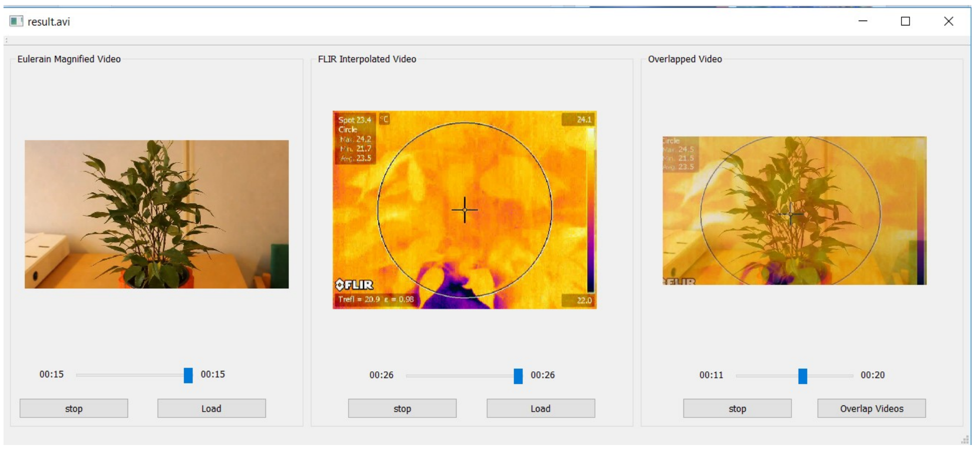


Figure 8.2: Result of plant overlapping.

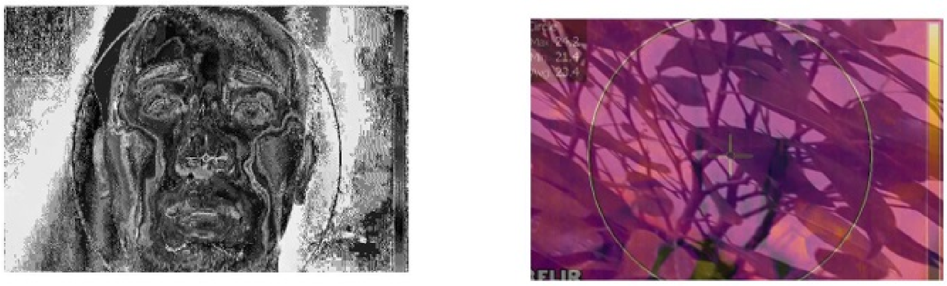


Figure 8.3: grayscale view of image when mixed and overlapped plant.

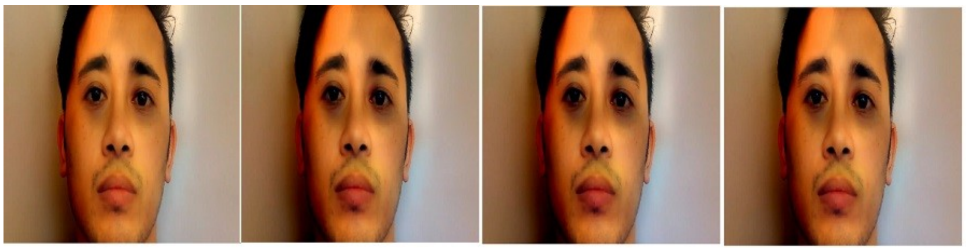


Figure 8.4: Result after applying EVM

$(amplify\_spatial\_Gdown\_temporal\_ideal(inFile, resultsDir, 60, 5.5, ..1.833333, 2.66667, 30, 1)$ function on video input. Due to motion artifacts in the eyes there appears a region of blood flow. And other artifacts are motion of lips that show the color variation. It is clearer from the result of thermal camera when there is move, more heat is observed in that region. Figure shows the color change due to movement of lips and eyes.

In the figure, there is consecutive frames where one can observe the change in color of nose hole due to inhaling and exhaling of air. It is the result from Butterflow algorithm after the frames are being interpolated. It is due to change in temperature of the inhaling in exhaling

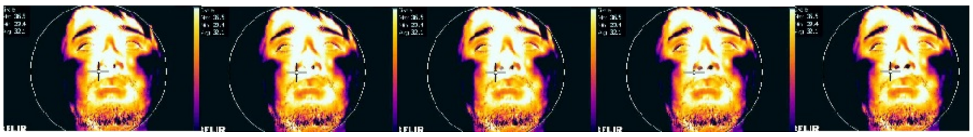


Figure 8.5: Change in color due to motion

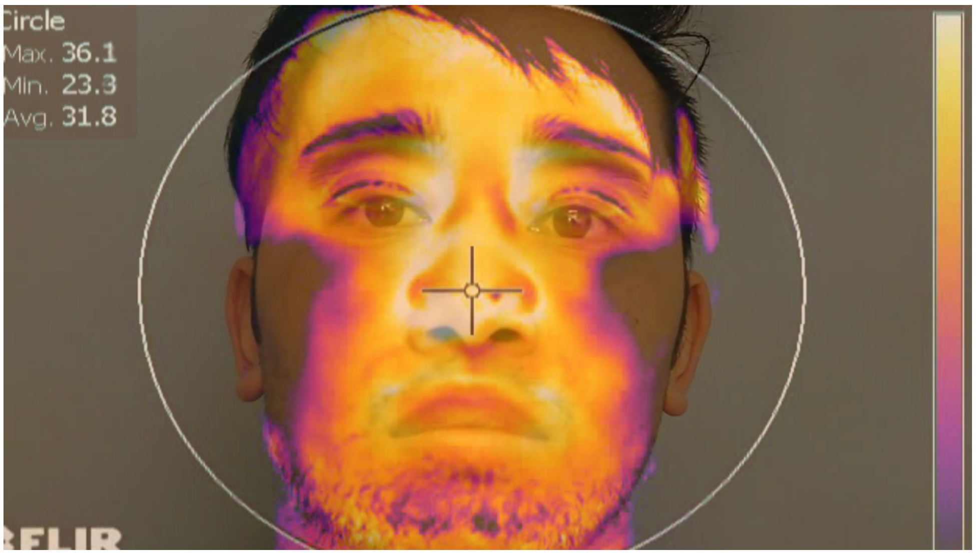


Figure 8.6: Result from overlapping

that the thermal camera observes different color for same video frame.

Chapter 9

Discussions and conclusion

Following objectives were fulfilled during the present work of thesis:

- The method EVM [9] was implemented that reveals the hidden information standard video sequence as justified in 6.1 and its theory explained in 2.3.
- Thermal video frames were interpolated using BF algorithm as explained in section 2.4 and the result that are obtained are in appendix section.
- Theoritical explanation of correlation between temperature and color over time.
- Two video sources were overlapped using linear blending technique.
- Code applicaiton is optimized and explained in chpater 7.
- Other single modules were tested which could probably be used such as feature extraction, imaging stitching etc.

In this way all the objective were achieved. However, before starting the project few things that were were not considered:

1. Distance between object and image
2. image resolution were not considered.

Due to that, the result obtained is not exactly overlapped but somehow it looked overlapped. One of the main reason for not overlapping of video sequence was due to the difference in size of videos where the regular video was taken as higher resolution and that of thermal video at fixed resolution(smaller resolution). The first method to implemented was EVM in MATLAB which is the first part of thesis work . The work was successfully completed. While moving to the second section to interpolate the frames. Thermal video was not provided on time, so have to wait nearly long time. References for second part was not included, so was really a challenging task to do research and find the method to test for the result. BF algorithm was implemented for second part and the result for it is pasted in appendix section for its validation. The frames obtained from this method exactly match with the frame rate of regular videos.

The main challenging task after that was to overlap those two different video sequence where OpenCV library is used to read, capture and write the video sequence. The application exactly play video from both the sources and overlapp them successfully if above mention things are considered while taking the video sequence. Image registration technique is just implemented from the MATLAB which provide control point selection function. Because of which the exact point in an image could be tracked and stretched to overlap both the image frames at matching position. Same technique could be implemented in main applicaiton but due to error in gpu driver, it is limited to blending technique only. For developing main applicaiton for the extension of video analysis QT-widgets application is selected with configuration with OpenCV [46]. Since the thesis project is experiment by programming so first section was done using MATLAB to work with all the videos frames in sequence whereas for the second section BF algorithm is taken into consideration. It is because it works with video frames to blend two frames and to give the perception of more fluid animation commonly used in high frame rate videos [18].

Time constraints is one of the important factor of this project and also the library set which is really a time-consuming task for the project. From the thesis work, we concluded that before doing work with image processing, few things should be taken into consideration such as distance, visibility, size and frames of the video should be properly taken. If we want to work with two videos simultaneously, both the video camera should be parallel so that the distance remains same.

Thus, for this thesis work i came to know more about MATLAB, OpenCV and more about image processing technique and different algorithm. Link to the main application is as follows:`https://source.uit.no/dinesh.s/MagnifierApplication`

Output from Butterflow processing of video

```
Dinesh@MSI MINGW64 ~/butterflow
butterflow − v − r30MOV_0028.mp4
'import sitecustomize' failed; use -v for traceback
[butterflow:INFO]: Version 0.2.3
[butterflow:INFO]: Cache directory: c:/users/dinesh/appdata/local/temp/butterflow-0.2.3
[butterflow:INFO]: At least one compatible OpenCL device was detected
[butterflow:INFO]: Using device: Intel(R) HD Graphics 4600 (autoselected)
[butterflow:INFO]: Hardware acceleration is enabled
[butterflow:INFO]: Rendering:
[butterflow:INFO]: Sequence: Duration=0:00:13.152000 (13.15s), Frames=104, Rate=7.9833741188
[butterflow:INFO]: Subregion (0): Time=0:00:00-0:00:13.152000 Frames=0-103 Speed=1.0,Du-
ration=?,Fps=?
[butterflow:INFO]: Rendering to: mov_0028.21688.mp4
[butterflow:INFO]: Final destination: C:/msys64/home/Dinesh/butterflow/out.mp4
[butterflow:INFO]: [Subprocess] Opening a pipe to the video writer
[butterflow:INFO]: Rendering progress: 0.00%
[butterflow:INFO]: Start working on Subregion (0): Time=0:00:00-0:00:13.152000 Frames=0-
103 Speed=1.0,Duration=?,Fps=?
[butterflow:INFO]: Frames in region: 0-103
[butterflow:INFO]: Region length: 104
[butterflow:INFO]: Region duration: 13.152000s
[butterflow:INFO]: Number of frame pairs: 103
[butterflow:INFO]: Interpolation rate: 3
[butterflow:INFO]: Time stepping: 0.250,0.500,0.750
[butterflow:INFO]: Frames to write: 394
[butterflow:INFO]: Will interpolate: 412
[butterflow:INFO]: Extra frames (to discard): 18
[butterflow:INFO]: Drop every: 22
[butterflow:INFO]: Dupe every: 0
[butterflow:INFO]: Ready to run: 104 times
[butterflow:INFO]: Showing a sample of the first and last 15 runs:
[butterflow:INFO]: To write: S0 I2,1 0.76%
[butterflow:INFO]: To write: S1 I3,0 1.78%
[butterflow:INFO]: To write: S2 I3,0 2.79%
[butterflow:INFO]: To write: S3 I3,0 3.81%
[butterflow:INFO]: To write: S4 I3,0 4.82%
[butterflow:INFO]: To write: S5 I3,0 5.84%
[butterflow:INFO]: To write: S6 I2,1 6.60%
[butterflow:INFO]: To write: S7 I3,0 7.61%
[butterflow:INFO]: To write: S8 I3,0 8.63%
[butterflow:INFO]: To write: S9 I3,0 9.64%
```

```
[butterflow:INFO]: To write: S10 I3,0 10.66%
[butterflow:INFO]: To write: S11 I3,0 11.68%
[butterflow:INFO]: To write: S12 I2,1 12.44%
[butterflow:INFO]: To write: S13 I3,0 13.45%
[butterflow:INFO]: To write: S14 I3,0 14.47%
[butterflow:INFO]: To write: S15 I3,0 15.48%
[butterflow:INFO]: <Snipping 74 runs from the console, but will update progress periodically
every 7 frames rendered>
[butterflow:INFO]: <Rendering progress: 20.30%>
[butterflow:INFO]: <Rendering progress: 27.16%>
[butterflow:INFO]: <Rendering progress: 34.01%>
[butterflow:INFO]: <Rendering progress: 40.86%>
[butterflow:INFO]: <Rendering progress: 47.46%>
[butterflow:INFO]: <Rendering progress: 54.31%>
[butterflow:INFO]: <Rendering progress: 60.91%>
[butterflow:INFO]: <Rendering progress: 67.77%>
[butterflow:INFO]: <Rendering progress: 74.62%>
[butterflow:INFO]: <Rendering progress: 81.47%>
[butterflow:INFO]: To write: S90 I2,1 88.07%
[butterflow:INFO]: To write: S91 I3,0 89.09%
[butterflow:INFO]: To write: S92 I3,0 90.10%
[butterflow:INFO]: To write: S93 I3,0 91.12%
[butterflow:INFO]: To write: S94 I3,0 92.13%
[butterflow:INFO]: To write: S95 I3,0 93.15%
[butterflow:INFO]: To write: S96 I2,1 93.91%
[butterflow:INFO]: To write: S97 I3,0 94.92%
[butterflow:INFO]: To write: S98 I3,0 95.94%
[butterflow:INFO]: To write: S99 I3,0 96.95%
[butterflow:INFO]: To write: S100 I3,0 97.97%
[butterflow:INFO]: To write: S101 I3,0 98.98%
[butterflow:INFO]: To write: S102 I2,1 99.75%
[butterflow:INFO]: Run 103 (this is the final run):
[butterflow:INFO]: To write: S103
[butterflow:INFO]: Done rendering Subregion (0)
[butterflow:INFO]: [Subprocess] Closing pipe to the video writer
[butterflow:INFO]: Rendering is finished
[butterflow:INFO]: Moving: mov_0028.21688.mp4 -> C:/msys64/home/Dinesh/butterflow/out.mp4
[butterflow:INFO]: Write ratio: 394/394, (100.00%)
[butterflow:INFO]: Final output frames: 104 source, +290 interpolated, +0 duped, -0 dropped
[butterflow:INFO]: Output file size: 1775.43 kB (-797.46 kB)
[butterflow:INFO]: Rendering took 1.39 mins, done.
```

Bibliography

[1] Kenneth Andersson and Hans Knutsson. *Continuous normalized convolution..* Continuous normalized convolution.. In: ICME (1). IEEE, 2002, pp. 725 728. isbn: 0-7803-7304-9.

[2] Kenneth And Hans: Continuous normalized Convolution
`http://www.dblp.uni-trier.de//db/conf/icmcs/icme2002-1.html`

[3] Kenneth Andersson and Hans Knutsson, carl-Fredrik *Continuous normalized convolution In signal Processing* In: Signal Processing 87.3 (Mar. 22, 2007), pp. 353 365.
`http://www.dblp.uni-trier.de//db/conf/icmcs/icme2002-1.html`

[4] Gunnar Farnebäck *Motion-based Segmentation of Image Sequences* In: Signal Processing 87.3 (Mar. 22, 2007), pp. 353 365. LiTH-ISY-EX-1596. MA thesis. SE-581 83 Linköping, Sweden: Linköping University, 1996.

[5] Gunnar Farnebäck*Polynomial Expansion for Orientation and Motion Estimation* Dissertation No 790, ISBN 91-7373-475-6. PhD thesis. SE-581 83 Linköping, Sweden: Linköping University, Sweden, 2002.
`http://lmi.bwh.harvard.edu/papers/pdfs/gunnar/farnebackSCIA03.pdf`

[6] Gunnar Farneback. *Two-Frame Motion Estimation Based on Polynomial Expan-sion* In: SCIA. LNCS 2749. Gothenburg, Sweden, 2003, pp. 363 370.
`http://lmi.bwh.harvard.edu/papers/pdfs/gunnar/farnebackSCIA03.pdf`

[7] *Video Frame Interpolation via Adaptive Convolution* Simon Niklaus, Long Mai, Feng Liu Portland State University *Video Frame Interpolation via Adaptive Convolution*
`https://arxiv.org/pdf/1703.07514.pdf`

[8] *DIGITAL SIGNAL PROCESSING OF HUMAN SKIN VIDEOS TO ESTIMATE REGIONS OF SIGNIFICANT BLOOD PERFUSION* ELYSE BENNER (Department of Electrical Engineering) CASE WESTERN RESERVE UNIVERSITY *DIGITAL SIGNAL PROCESSING OF HUMAN SKIN VIDEOS TO ESTIMATE REGIONS OF SIGNIFICANT BLOOD PERFUSION*
`https://etd.ohiolink.edu/!etd.send_file?accession=case1441126290&disposition=inline`

[9] *Eulerian Video Magnification for Revealing Subtle Changes in the World* Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, William T. Freeman (MIT CSAIL) Quanta Research Cambridge, Inc.
`http://cacm.acm.org/magazines/2017/1/211095-eulerian-video-magnification-and-analysis/`
`https://www.extremetech.com/extreme/149623-mit-releases-open-source-software-that-reve`

[10] Liu C, Torralba A, Freeman W T, et al. Motion Magnification. ACM Transactions on Graphics. 2005, 24(Jul): 519-526. *Motion Magnification*

[11] Gautama T, Van Hulle M M. 2002, 13(5): 1127-1136 *A Phase-Based Approach to the Estimation of the Optical Flow Field Using Spatial Filtering. IEEE Transactions on Neural Networks.*

[12] Rubinstein M, Liu C, Sand P, et al. Colorado Springs, USA: IEEE Computer Society, 2011. 313-320 *Motion De-noising with Application to Time-lapse Photography. In: IEEE Computer Vision and Pattern Recognition.*
`http://ieeexplore.ieee.org/document/7003748/`

[13] Jacobs, David. Class Notes for CMSC 426 (2005) *"Image gradients."*
`https://en.wikipedia.org/wiki/Image_gradient`

[14] Gonzalez, Rafael; Richard Woods. Digital Image Processing (3rd ed.). Upper Saddle River, New Jersey: Pearson Education, . *Digital Image Processing (3rd ed.)*
`http://www.fluke.com/fluke/uses/comunidad/fluke-news-plus/articlecategories/rd/how%20i`

[15] Mrinal Kanti Bhowmik, ,Sandip Shil,Priya Saha

Feature Points Extraction of Thermal Face Using Harris Interest Point Detection
`http://www.sciencedirect.com/science/article/pii/S221201731300577X`

[16] Lisa Gottesfeld Brown, , ACM Computing Surveys archive, volume 24, issue 4, December 1992), pages 325 - 376 *A survey of image registration techniques (abstract)*

[17] A. Ardeshir Goshtasby: 2-D and 3-D Image Registration for Medical, Remote Sensing, and Industrial Applications, Wiley Press, 2005. *biological imaging and brain mapping*
`http://www.comp.nus.edu.sg/∼cs4243/lecture/register.pdf`

[18] By Duong Pham *Butterflow Algorithm Butterflow Algorithm*
`https://github.com/dthpham/butterflow`

[19] Toga, Arthur W. (1998-11-17). Academic Press. ISBN 9780080525549 *Brain Warping.*

[20] University of Utah.Retrieved 2016-03-21.
`utah.pure.elsevier.com.`

[21] ResearchGate. doi:10.1023/B:VISI.0000043755.93987.aa. Retrieved 2016-03-12 *"Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms"*

[22] Joshi, S. C.; Miller, M. I. (2000-01-01). . IEEE transactions on image processing: a publication of the IEEE Signal Processing Society. *"Landmark matching via large deformation diffeomorphisms"*
`https://www.mristudio.org/wiki/`

[23] B. Srinivasa Reddy, B. N. Chatterji: . IEEE Transactions on Image Processing, vol. 5, no. 8. *An FFT-Based Technique for Translation, Rotation and Scale-Invariant Image Registration*

[24] G. Wohlberg, S. Zokai: A paper on using the log polar transform for registration. *ROBUST IMAGE REGISTRATION USING LOG-POLAR TRANSFORM*

[25] Simonson, K., Drescher, S., Tanner, F., IEEE Pattern Analysis and Machine Intelligence, Vol. 29, No. 1, January 2007 *A Statistics Based Approach to Binary Image Registration with Uncertainty Analysis.*

[26] Domokos, C., Kato, Z., Francos, J., Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2008 *Parametric estimation of affine deformations of binary images.*

[27] Dinesh Shrestha *Magnifier App*
`https://source.uit.no/dinesh.s/MagnifierApplication.git`

[28] N.Narayana Murty, A.Venkateswara Rao, M.Tech,*International Journal* &*Magazine of Engineering, Technology, Management and Research*
`http://www.ijmetmr.com/olseptember2015/NNarayanaMurty-AVenkateswaraRao-105.pdf`

[29] Pedro Boloto Chambino *Android-based implementation of Eulerian Video Magnification for vital signs monitoring*
`http://p.chambino.com/dissertation/pulse.pdf`

[30] *Laser Doppler Monitoring*
`http://www.perimed-instruments.com/laser-doppler-monitoring`

[31] *Doppler radar* Doppler Radar
`https://en.wikipedia.org/wiki/Doppler_radar`

[32] WIKI*Thermography*
`https://en.wikipedia.org/wiki/Thermography`

[33] Platt*Platt Electric supply*
`https://www.platt.com/search.aspx?q=phi13`

[34] Vitrox *ViTrox Corporation Berhad*
`https://en.wikipedia.org/wiki/ViTrox`

[35] Bayer Filter*Bayer Filter*
`https://en.wikipedia.org/wiki/Bayer_filter`

[36] *Digital Image Processing Digital Image processing*
`https://en.wikipedia.org/wiki/Digital_image_processing`

[37] *Encyclopedia*
http://www.pcmag.com/encyclopedia/index/a

[38] *Health Medical Team 2015*
http://www.healthline.com/health/about-us

[39] *Image alignment in opencv (ECC)*
http://www.learnopencv.com/image-alignment-ecc-in-opencv-c-python/

[40] Georgios D. Evangelidis ; Emmanouil Z. Psarakis*Parametric Image alignment Using Enhanced Correlation Coefficient Maximization*
http://ieeexplore.ieee.org/document/4515873/

[41] Image registration*Image registration*
https://en.wikipedia.org/wiki/Image_registration

[42] Image registration tool box *Image registration in MATLAB*
https://se.mathworks.com/discovery/image-registration.html

[43] *Gradient domain image processing* Gradient domain Image processing
https://en.wikipedia.org/wiki/Gradient-domain_image_processing

[44] Canny Edge Detection*Canny EDge detection*
http://docs.opencv.org/trunk/da/d22/tutorial_py_canny.html

[45] *A Study of the Yosemite Sequence Used as a Test Sequence for Estimation of Optical Flow* Kalviainen H., Parkkinen J., Kaarna A.
https://link.springer.com/chapter/10.1007/11499145_67

[46] *OpenCv Documentation* OpenCv library
https://docs.opencv.org

[47] *Dense Optical flow Expansion based on polynomial basis Approximation* Code Project
https://www.codeproject.com/Articles/807839/Dense-Optical-Flow-Expansion-Based-On-Poly

[48] *Blending thermal and visible light* Infrared solutions Inc, Fluke company
http://www.fluke.com/fluke/uses/comunidad/fluke-news-plus/articlecategories/rd/how%20i