Efficient Registration of Limited Field of View Ocular Fundus Imagery

by

Christo Carel van der Westhuizen

Thesis preseted in partialfulfilment of the requirements for the degree Master of Science in Engineering in the Facultyof Engineering at Stellenbosch University

Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Supervisor: Dr. M.M. Blanckenberg
Department of Electrical & Electronic Engineering

December 2013

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: December 2013...............................

Abstract

Efficient Registration of Limited Field of View Ocular Fundus Imagery

C.C. van der Westhuizen

Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MScEng (Electronic)

December 2013

Diabetic- and hypertensive retinopathy are two common causes of blindness that can be prevented by managing the underlying conditions. Patients suffering from these conditions are encouraged to undergo regular examinations to monitor the retina for signs of deterioration.

For these routine examinations an ophthalmoscope is used. An ophthalmoscope is a relatively inexpensive device that allows an examiner to directly observe the ocular fundus (the interior back wall of the eye that contains the retina). These devices are analog and do not allow the capture of digital imagery. Fundus cameras, on the other hand, are larger devices that offer high quality digital images. They do, however, come at an increased cost and are not practical for use in the field.

In this thesis the design and implementation of a system that digitises imagery from an ophthalmoscope is discussed. The main focus is the development of software algorithms to increase the quality of the images to yield results of a quality closer to that of a fundus camera. The aim is not to match the capabilities of a fundus camera, but rather to offer a cost-effective alternative that delivers sufficient quality for use in conducting routine monitoring of the aforementioned conditions.

For the digitisation the camera of a mobile phone is proposed. The camera is attached to an ophthalmoscope to record a video of an examination. Software algorithms are then developed to parse the video frames and combine those that are of better quality. For the parsing a method of rapidly selecting valid frames based on colour thresholding and spatial filtering techniques are developed. Registration is the process of determining how the selected frames fit together. Spatial cross-correlation is used to register the frames. Only translational transformations are assumed between frames and the designed algorithms focuses on estimating this relative translation in a large set of frames. Methods of optimising these operations are also developed. For the combination of the frames, averaging is used to form a composite image.

The results obtained are in the form of enhanced grayscale images of the fundus. These images do not match those captured with fundus cameras in terms of quality, but do show a significant increase when compared to the individual frames that they consists of. Collectively a set of video frames can cover a larger region of the fundus than what they do individually. By combining these frames an effective increase in the field of view is obtained. Due to low light exposure, the individual frames also contain significant noise. In the results the noise is reduced through the averaging of several frames that overlap at the same location.

Uittreksel

Doeltreffende Registrasie van Lae Kwaliteit Okulêre Fundus Beelde

("Efficient Registration of Limited Field of View Ocular Fundus Imagery")

C.C. van der Westhuizen
Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Tesis: MScEng (Electronic)
Desember 2013

Diabetiese- en hipertensiewe retinopatie is twee algemene oorsake van blindheid wat deur middel van die behandeling van die onderliggende oorsake voorkom kan word. Pasiënte met hierdie toestande word aangemoedig om gereeld ondersoeke te ondergaan om die toestand van die retina te monitor.

'n Oftalmoskoop word gebruik vir hierdie roetine ondersoeke. 'n Oftalmoskoop is 'n relatiewe goedkoop, analoë toestel wat 'n praktisyn toelaat om die agterste interne wand van die oog the ondersoek waar die retina geleë is. Fundus kameras, aan die ander kant, is groter toestelle wat digitale beelde van 'n hoë gehalte kan neem. Dit kos egter aansienlik meer en is dus nie geskik vir gebruik in die veld nie.

In hierdie tesis word die ontwerp en implementering van 'n stelsel wat beelde digitaliseer vanaf 'n oftalmoskoop ondersoek. Die fokus is op die ontwikkeling van sagteware algoritmes om die gehalte van die beelde te verhoog. Die doel is nie om die vermoëns van 'n fundus kamera te ewenaar nie, maar eerder om 'n koste-effektiewe alternatief te lewer wat voldoende is vir gebruik in die veld tydens die roetine monitering van die bogenoemde toestande.

'n Selfoonkamera word vir die digitaliserings proses voorgestel. Die kamera word aan 'n oftalmoskoop geheg om 'n video van 'n ondersoek af te neem. Sagteware algoritmes word dan ontwikkel om die videos te ontleed en om videogrepe van goeie kwaliteit te selekteer en te kombineer. Vir die aanvanklike ontleding van die videos word kleurband drempel tegnieke voorgestel. Registrasie is die proses waarin die gekose rame bymekaar gepas word. Direkte kruiskorrelasie tegnieke word gebruik om die videogrepe te registreer.

Daar word aanvaar dat die videogrepe slegs translasie tussen hulle het en die voorgestelde registrasie metodes fokus op die beraming van die relatiewe translasie van 'n groot versameling videogrepe. Vir die kombinering van die grepe, word 'n gemiddeld gebruik om 'n saamgestelde beeld te vorm.

Die resultate wat verkry word, word in die vorm van verbeterde gryskleur beelde van die fundus ten toon gestel. Hierdie beelde is nie gelykstaande aan die kwaliteit van beelde wat deur 'n fundus kamera geneem is nie. Hulle toon wel 'n beduidende verbetering teenoor individuele videogrepe. Deur dat 'n groot versameling videogrepe wat gesamentlik 'n groter area van die fundus dek gekombineer word, word 'n effektiewe verhoging van data in die area van die saamgestelde beeld verkry. As gevolg van lae lig blootstelling van die individuele grepe bevat hul beduidende ruis. In die saamgestelde beelde is die ruis aansienlik minder as gevolg van 'n groter hoeveelheid data wat gekombineer is om sodoende die ruis uit te sluit.

Acknowledgements

I would like to express my sincere gratitude to the following people:

- My supervisor, Dr Mike Blanckenberg, for giving me this wonderful opportunity and for the support and the time dedicated to me.

- Wayne Swart, for the technical advice and collaboration.

- My friends and family for always giving advice and help when possible. And for being more than willing test subjects along the way.

Dedications

This thesis is dedicated to:
My parents. Thank you for the emotional and financial support over my years of studying. Thank you for raising me with the urge to always observe and learn with every opportunity.
And to my dear friend, Lucas, without your support and understanding this would never have been possible.

Contents

List of Figures

List of Tables

List of Abbreviations

FOV	Field of View
SPECT	Single-photon Emission Computed Tomography
RANSAC	Random Sample Consensus
MRI	Magnetic Resonance Imaging
SNR	Signal to Noise Ratio
DLL	Dynamic-link Library
ROI	Region Of Interest

Nomenclature

$h[.]$	Discrete 1D signal
$g[.]$	Discrete 1D signal
$f[.]$	Discrete 1D signal
$f(x, y)$	Continuous 2D function
$g(x, y)$	Continuous 2D function
$G(x, y)$	Continuous 2D Gaussian function
$I_{...}$	A discrete 2D image matching the description '...'
K	A discrete 2D convolution kernel
T	A discrete 2D template- or sensed image
$.width$	Width of a 2D image(I), template(T) or kernel(K)
$.height$	Height of a 2D image(I), template(T) or kernel(K)
N	Width and height of an image or kernel of size $N \times N$
P	Width and height of a convolution kernel's padding
r_{ccorr}	1D cross correlation result
R_{ccorr}	2D cross correlation result

$\mathrm{R}_{ccorr\ normed}$	Normalised 2D cross correlation result
$\mathrm{R}_{sq\ diff}$	2D square difference template matching result
H	Transformation matrix
θ	Rotation in degrees
ρ	Radius
$V_{...}$	2D vector indication the offset of an entity matching the description '...'
d_x	derivative kernel in the x direction
d_y	derivative kernel in the y direction
$sobel\ d_x$	Sobel derivative kernel in the x-direction
$sobel\ d_y$	Sobel derivative kernel in the y-direction
Y	Luma intensity of a YCrCb colour
Cr	Croma Red intensity of a YCrCb colour
Cb	Croma Blue intensity of a YCrCb colour
σ	Standard deviation
σ_x	Standard deviation in the x-direction
σ_y	Standard deviation in the y-direction
β	FOV angle in degrees

Chapter 1

Introduction

1.1 Description

Fundus cameras and ophthalmoscopes are used to examine the inside of a patient's eye. The back interior of the eye is known as the ocular fundus. Here the examiner can observe blood flow without any intrusion and with careful observations, certain diagnoses can be made. In a routine examination, the retina, the part of the eye capturing the incoming light, is inspected to ensure that it is in a healthy condition. Any damage to the retina can lead to partial or complete blindness.

Two common conditions that cause the degradation of the retina are hypertension and diabetes. If a patient suffers from these conditions, it is imperative that the progression of the disease is carefully monitored, as proper treatment of the underlying cause can reduce the damage to the retina.

As stated, two devices can be used to examine the ocular fundus. The first, the fundus camera, is a large stationary device that takes high quality digital images of the entire fundus. The second, the ophthalmoscope, is a handheld analog device that offers a limited view of the fundus at a time. The quality of the images obtained with a fundus camera are superior to that which can be observed with an ophthalmoscope. However, the fundus cameras come at a higher cost.

The aim is to design and implement a system that digitises imagery from an analog ophthalmoscope and to combine this imagery in an effort to increase its quality. This solution could offer a cost-effective alternative to fundus cameras that is ideal for use in the field.

The concept is that a number of limited FOV (Field of View) images can be combined to yield an image that cover a larger section of the fundus. Also, by combining several noisy images a better estimate of the true visual can be obtained.

1.2 Content Overview

Firstly an overview of the anatomy of the eye and the ophthalmic examination process is given. These aspects dictate the nature of the captured imagery and thus influences design choices at a later stage. Difficulties in the conduction of a typical examination are also elaborated on.

Thereafter, a broad review of image registration is given. Methodologies to solving registration and instances where it is used, is discussed. Literature regarding the registration of retinal (or fundus) images is also covered.

A proposed system and its benefits are then discussed. The cost of the system is compared to that of other devices. The benefits of having digital images with increased quality are also commented on.

For the design of the algorithms a look is taken at the captured videos and the quality they possess. Thereafter, the design choices are discussed in great detail. Methods of achieving registration in the specific application as well as specific optimisations are discussed.

Finally, the results are presented. The output of the system is enhanced images. These are compared with benchmark images captured with a fundus camera.

1.3 Objectives

Several objectives were set for the research. Firstly, emphasis is placed on the cost and size of the device. The aim is to develop a system that is lightweight while offering a reduced cost alternative to fundus cameras.

As the main focus is on the design of the algorithms, several objectives are defined in terms of quality, speed and the robustness of these algorithms.

To attain quality, the aim is to achieve images of increased FOV as well as an increase in detail observable. The aim is not to match the quality of fundus cameras, but to investigate what can be achieved at a reduced cost.

Emphasis is also put on the speed of the processing since efficient algorithms could allow implementation directly on a mobile device.

The robustness of the algorithms are to be considered throughout the design process. The algorithms should be able to operate with as little user input as possible and should not be sensitive to changes in capturing conditions.

Chapter 2

Ocular Fundus Examination

2.1 Introduction

An ocular fundus examination offers a view of the inside of a patient's eye; here lies the only part of the body where *in vivo* blood flow can be observed directly without any intrusion. During the fundus examination a physician inspects the interior of the patient's eye using an ophthalmoscope and checks for certain diseases or conditions.

Before the examination process can be discussed, a brief overview of the anatomy of the eye is given in the following section. Throughout this chapter [32] is used as reference for the discussion on the anatomy and conditions affecting the eye.

2.2 Anatomy of the Eye

The basic anatomy of the human eye is illustrated in figure 2.1. Here some of the main structures are highlighted.

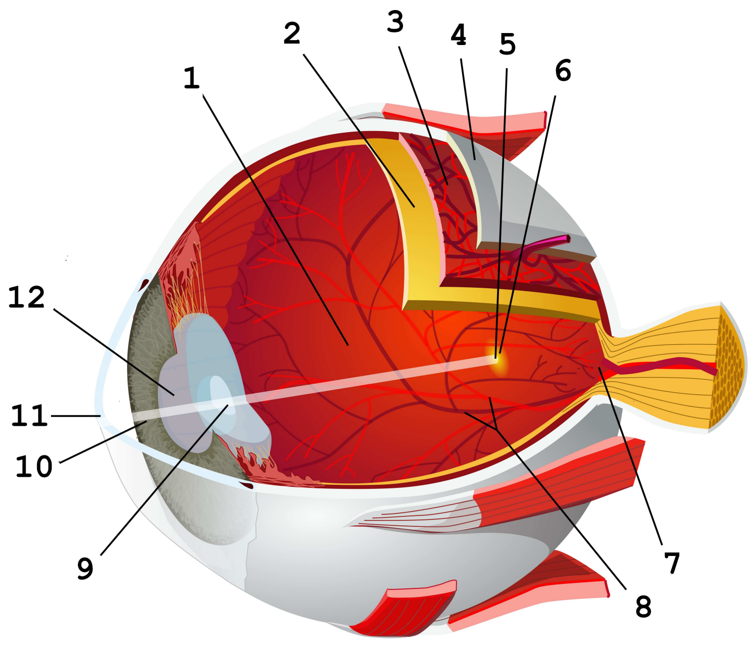


Figure 2.1: Anatomy of the Eye: 1) Posterior Compartment 2) Retina 3) Choroid 4) Sclera 5) Fovea 6) Macula 17) Optic Disk 8) Retinal Arteries and Veins 9) Lens 10) Iris 11) Cornea 12) Pupil [45]

The outer most layer of the eye consists of the transparent cornea, opaque white sclera and the limbus (not shown) that connects the two. These layers protect the eye with the cornea covering and protecting the iris and lens beneath it.

The task of regulating the amount of light entering the eye is executed by the iris. Light enters the eye through an opening in the centre of the iris called the pupil. Sphincter and dilator muscles allow the iris to dilate and constrict to change the size of the pupil and thus changing the amount of light that enters the eye. Light passing through the pupil is focused on the retina at the back of the eye by the lens with the lens shape adjusted by the ciliary body. As the shape of the lens is adjusted, so is the focal distance of the eye.

The area at the back interior of the eye is known as the optical fundus. Fundus is a medical term that refers to the area of an organ visible opposite to its opening. In the human eye this area lies opposite the lens and pupil and can be observed through these structures. Figure 2.2 shows a clear image of a human ocular fundus. The ocular fundus consists of several structures that are of interest, namely the retina, optic disk, fovea and macula. These structures are typically visible during an ophthalmic examination and are of importance for this study. These structures also serve as landmarks to determine whether a clear view of the fundus is obtained.

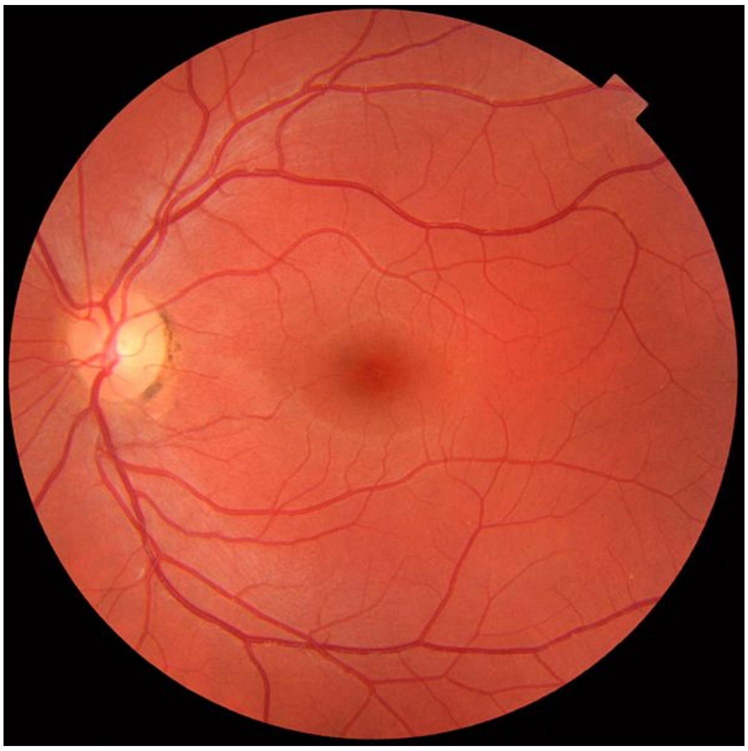


Figure 2.2: Image of a healthy retina in an ocular fundus image [19].

Photoreceptors on the retina allow for the capture of images focused on it by the lens. These images are then transmitted to the brain via the optic nerves. The retina is a thin, delicate and transparent tissue with the only observable part being the blood vessels that supply it. When looking at the fundus the blood vessels of the retina can be seen against the background of the choroid and scelera. Good blood supply is imperative to the functioning of the retina and without proper supply, areas of the retina dies and leads to partial or complete blindness. During an ophthalmic examination the focus is the health of the vascular system since it is the component mostly effected by disease. The retina itself is not inspected since it is transparent. There are instances where the retina would be visible due to damage, however, this is not the norm.

Another structure clearly visible on the fundus is the optic disk. The optic disk is where nerves and blood vessels enter the eye. This area is also typically inspected during an examination. Disease in other areas of the retina rarely affect the whole eye. They start localised and increase in size as the damage progresses. However, at the optic disk the situation is different. Since blood vessels and nerves convene here; conditions affecting the optic disk can compromise the functioning of the whole eye and can cause complete blindness. The optic disk is thus closely inspected during an ophthalmic examination. Shape, colour and integrity of the optic disk is observed to check for irregularities.

2.2.1 Diseases

Diabetic retinopathy, as the name suggests, is the degradation of the patient's retina due to diabetes. It is a common cause of blindness accounting for 8% of blindness among South Africans [10]. To prevent vision loss, careful monitoring of the retina is suggested, with a fundus examination recommended once a year for diabetic patients.

Treatment of diabetic retinopathy is conducted by managing the diabetes itself. If the diabetes worsens; the indicators in the retina increases. These indicators also serve as a method of gauging the damage to other organs caused by diabetes.

If left unchecked, diabetic retinopathy develops into proliferative diabetic retinopathy, with half of the cases of proliferative retinopathy resulting in complete blindness of the patient. However, if the diabetes is carefully regulated the damages caused in early stages may resolve without vision loss.

The indicators in the fundus of retinopathy are caused by a weakening of the blood vessels that supply the retina and the choroidial wall. These weakened vessels start to leak causing visible haemorrhages and in advanced cases, leads to loss of blood supply to sections of the retina. In turn, the loss of blood supply leads to starvation of the retina.

The indicators for diabetic retinopathy include:

- aneurysms
- exudates
- haemorrhages
- areas of neovascularisation

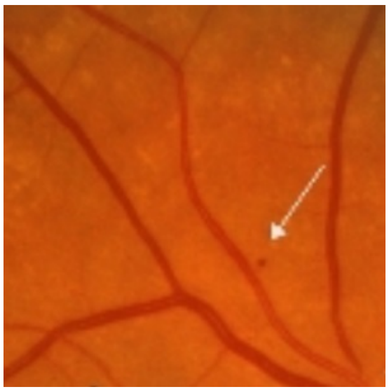
[32]

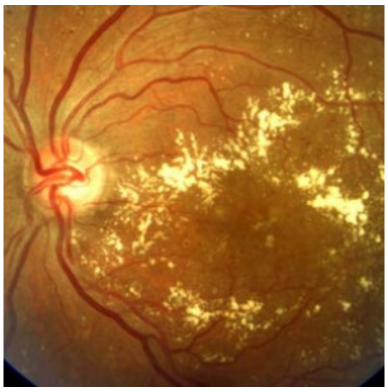
The indicators for prolific diabetic retinopathy include:

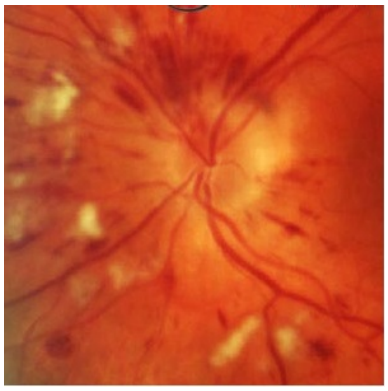
- neovascularisation
- edema
- scarring
- vitreal involvement

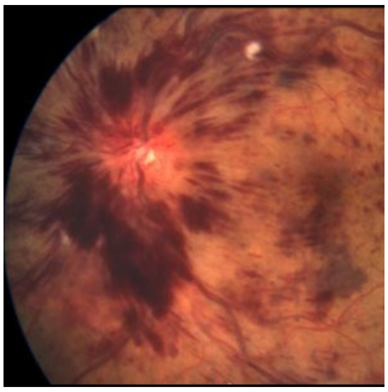
[32]

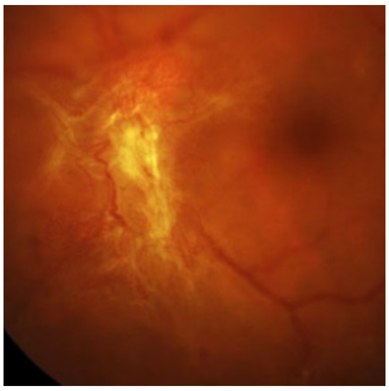
Figure 2.3 shows some of these mentioned indicators.


(a) Micro Aneurysm [13]


(b) Hard Exudates [23]


(c) Cotton Wool Spots [23]


(d) Flame Shaped Haemorrhages [23]


(e) Neovascularisation [23]

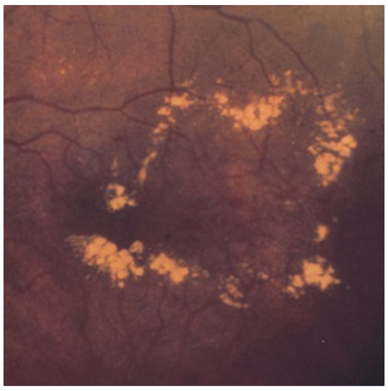

(f) Macular edema [46]

Figure 2.3: Damages to the retina.

2.3 Examination

The examination of the fundus is conducted using a ophthalmoscope or a fundus camera. An ophthalmoscope typically refers to a handheld device that the physician uses to directly observe the retinal fundus. Fundus cameras refer to larger stationary devices that stabilise the patients head and captures an image using a targeting lens system and an image sensor.

Handheld ophthalmoscopes are smaller, lighter in weight, and less expensive than fundus cameras, but offer images of poorer quality than those of fundus cameras.

The following section discusses the construction of a handheld ophthalmoscope. Thereafter the process of conducting and examination is discussed. Lastly, factors that hinder the examination are described.

2.3.1 Ophthalmoscope

Figure 2.4 shows a cross section of the Welch Allen Pan Optic Ophthalmoscope (A direct handheld ophthalmoscope). Other ophthalmoscopes may differ in the configuration of the components, however, the principles remain similar.

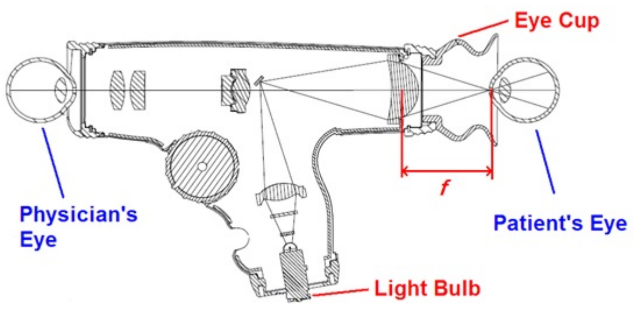


Figure 2.4: Cross-section of the Welch Allyn Pan-Optic handheld opthalmoscope (Courtesy of Wayne Swart).

The examiner aims the ophthalmoscope into the patient's eye shown on the right end and observes the reflected light on the left.

Three components perform the main functioning of an ophthalmoscope. These components are a light source; a beam splitter and a lens system. The light source is used to illuminate the patient's fundus. An obstacle arises with the placement of the light source since the beam of light should travel in along the same axis as the reflected light that the examiner observes. To achieve the illumination along the axis of observation without obstructing the view to the examiner, a beam splitter is used. The beam splitter is a mirror that causes the light from the source to be redirected into the patient's eye while allowing the reflected light to pass through and be observed by the examiner. The leans system is used to compensate for the lenses in both the examiner and the patient's eye and focus the light reflected from the patient onto the examiner's retina.

2.3.2 Procedure

Below, a list of the steps of conducting a fundus examination using the Welch Allyn Pan Optic Ophthalmoscope is given. The guide is based on the information available directly from Welch Allyn [2].

- Ensure that the examination room is darkened. This is not a prerequisite, but helps keep the patient's pupils dilated and eliminates undesired external light that might enter the scope.
- Adjust the focal wheel of the scope while viewing a distant object (5m away).
- Select the appropriate apeture using the aperture wheel. The ideal apeture for view through an undilated pupil is the smaller apeture.
- Turn on the light source by turning the rheostat until its fully right. This sets the source to the brightest possible setting.
- Instruct the patient to focus on a distant object while aiming the scope towards the pupil. The examiner uses his left eye to examine the patient's right eye, and his right eye to examine the patient's left.
- While aiming the scope, move closer to the patient's eye and aim towards the red reflex. The red reflex is red light reflected from the patient's retina.
- Locate the optic disk. Adjust the rheostat to reduce the light intensity for the patient's comfort.
- Trace the blood vessels outward from the optic disk to locate the macula and examine the blood vessels themselves.

2.3.3 Obstructions

When conducting the examination, a clear view of the fundus can be obscured by a number of obstacles and abnormalities. The following paragraphs discuss these interferences with regards to the eye's normal reaction to light as well as certain diseases or conditions that could obstruct the view of the fundus.

Reaction to light The duty of the pupil is to regulate the amount of light that enters the eye. When too much light is present, the pupil constricts and when too little light is present the pupil dilates allowing more light to pass through to the retina.

Conducting a fundus examination thus has an inherent problem. To view the fundus, it has to be illuminated, but the illumination causes the pupil to constrict and limits the viewable area.

Prolonged exposure to bright light can also cause irritation of the eye and with this irritation comes blinking and tear formation.

Besides the obvious obstruction of the view, the eye blinking causes movement and refocusing of the eye. These movements, called saccades, can cause a different section of fundus to be visible before and after the blink, causing the examiner to lose track of what section is being observed.

Blinking also spreads tearing across the eye, keeping the eye from drying out. This increased moisture can cause an increase in glare visible on the eye. Glare is typically caused by internal reflection in a lens system.In this instance, the reflection is between the lens of the patient's eye and the ophthalmoscope's lens.

A method of avoiding pupil constriction is the use of mydricatic drops. This causes an involuntary dilation of the pupil and allows a wider view of the fundus. While these drops aid the process, routine examinations are often conducted without them to increase patient comfort and reduce consultation time. A study is provided in [4] discusses the diagnostic value of imagery obtained from undilated pupils. They conclude that the imagery still contains valuable diagnostic information when considering the unobtrusiveness of the procedure.

Another method often used to circumvent pupil constriction in more expensive equipment, is the use of infra-red light to illuminate the eye. The human eye is only sensitive to a certain section of the light spectrum. Infra-red light falls just outside the visible range and is thus not observed by the patient. The patient's eye does thus not react to the illumination and the pupil does not constrict. A wider view of the fundus is achieved with minimal discomfort to the patient, however, since the light is not detected by the human eye, the examiner is also not able to directly observe the results. When using infra-red illumination, the use of a digital image sensor sensitive to the range is required. These additional lighting sources and image sensors cause a considerable increase in the cost of the device. This cost increase are discussed further in section 4.2.3.

Obstruction due to Disease Certain eye conditions can also obscure the view of the fundus. The following list is a brief mention of some of these conditions.

- Cataracts are a opacification of the lens, thus not allowing focused light to pass through it. It causes partial blindness in the patient and would obstruct the view of the fundus.

- Lens detachment can be caused by a number of factors and causes the fundus to not be observable.

- Retinal detachment is the detachment of the retina from the choroid. This leads to the blurring of the fundus at the region of detachment.

Detachment leads to the tearing of the retina and constitutes a medical emergency that requires immediate treatment to prevent blindness.

- Damage to the cornea in the form of scratches also obscures the view of the fundus.

While all of these conditions require medical attention and can causes partial or complete blindness, they are not associated with the routine fundus examinations. Routine examinations are typically employed to monitor the progression of the conditions mentioned in section 2.2.1.

Chapter 3

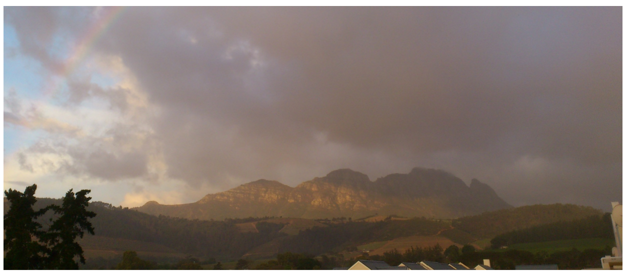
Image Registration

3.1 Introduction

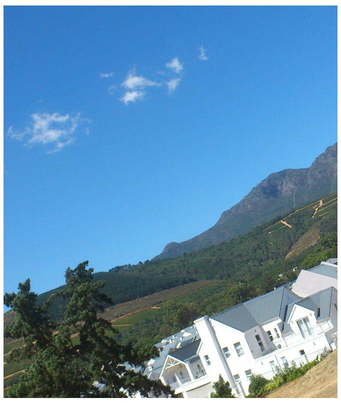
In this chapter an overview of what image registration entails is given, where it is used and some common approaches to solving it explained. Some of the approaches are discussed in more detail in areas where they were applied or tested in later chapters. If unfamiliar with the topic, the reader is encouraged to acquaint themselves with the basic principles of image processing as discussed in Appendix A before commencing with this chapter.

Image registration is in its simplest form the process of aligning images in such a way that their corresponding features overlap. As explained by Zitova and Flusser [53], it is the process of overlaying a set of images of the same scene taken in different conditions, at different times and from different viewpoints.

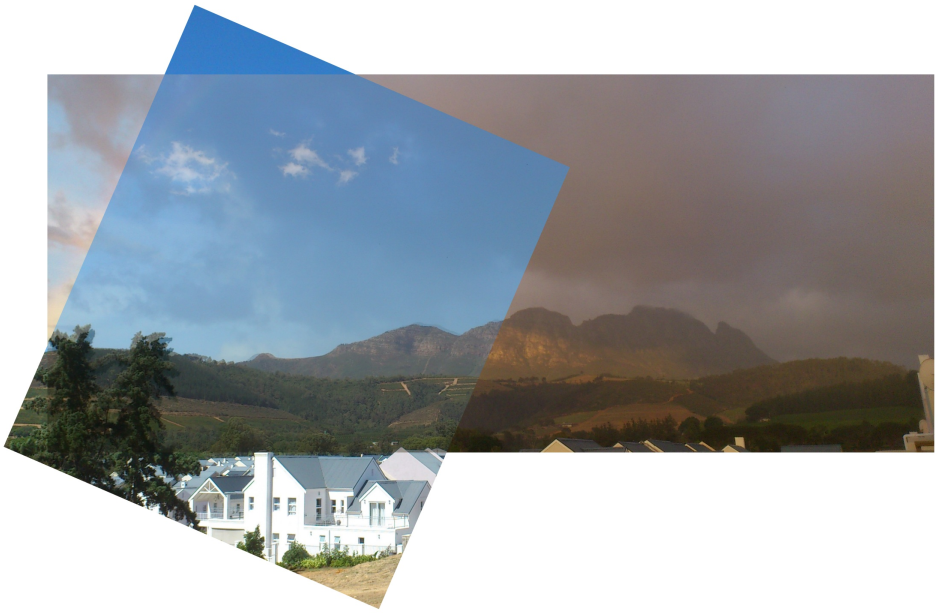
This fusion of information allows for a combination of multi-modal information, the detection and comparison of differences in a scene as well as the detection of certain objects [18].



(a) Mountain scene



(b) Mountain scene under different conditions



(c) Combined images

Figure 3.1: The registration and combination of two images.

The concept is demonstrated with a trivial example. The two images shown in figure 3.1a and 3.1b cover the same mountain scene from slightly different views and were taken in different weather conditions. To align these images, one would note that some sections of the mountain are visible in both the images and that the images can be moved around to have these mountain sections overlap. The mountain thus serves as a constant to align the images by and is called an invariant feature [11]. By rotating, scaling and moving the images, the mountain lines up as shown in Figure 3.1c.

The image in figure 3.1a is referred to as the reference image and the image in figure 3.1b the sensed image. These are common terms used throughout this text. The reference image refers to an image acquired at a previous occasion and the sensed image is a new image that is to be aligned to the reference image.

This example illustrates how a human operator would interpret and solve the problem of registering two images. Ideally, one would want to design a system that efficiently and autonomously registers a larger set of images.

In early development of image registration, the process was human driven to a large extent, with an expert selecting features that correspond between two images [53]. In contrast with this, modern methods aim to operate autonomously with minimal user assistance.

3.2 Image Registration Applications

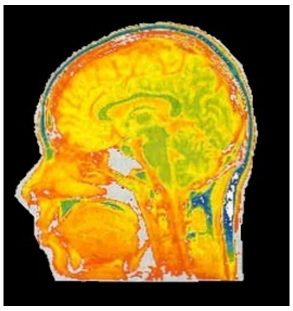
Image registration has broad set of uses. In this section we discuss some of the methodologies to image registration and elaborate with some example applications.

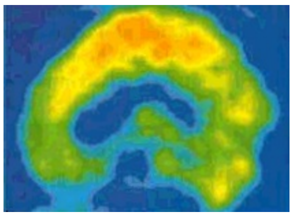
A popular application is image mosaicking. Mosaicking is the combination of a number of photos of a scene taken from different viewpoints to compile a wide angle image of the scene. The compiled image is at a higher pixel resolution than what could be achieved with a single photo of the scene.

An extreme example of image mosaicking is the commercial Gigapan system [38]. Here an electronic system incrementally pans a camera, with a zoom lens attached, over a desired scene to gather a large set of high detail images. These images are then combined to form a single high resolution image. The scenes are typically static city- or landscapes since the process is slow and temporal change would create undesired artefacts.

There is also a wide selection of panoramic mobile applications available that execute the same task on a simplified level. The user is prompted to sweep the device's camera over a section of landscape, a series of images are captured by the device and combined to form a single panorama. The accuracy of these mobile applications are typically poor because of constraints in processing time and power.

The combination of images from sensors capturing data in different manners, called multi modal image registration, is another common use of image registration. This it typically used for medical applications where registration is employed to combine data from different scanning devices [53]. Figure 3.2 demonstrates the combination of MRI and SPECT scans. Note that the structures described by the respective scans are unalike, but that registration is still achieved. Finding corresponding features in apparent dissimilarities is the main focus of multi-modal registration.


(a) MRI


(b) SPECT

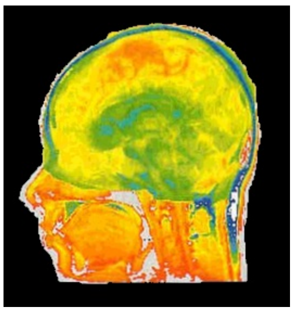
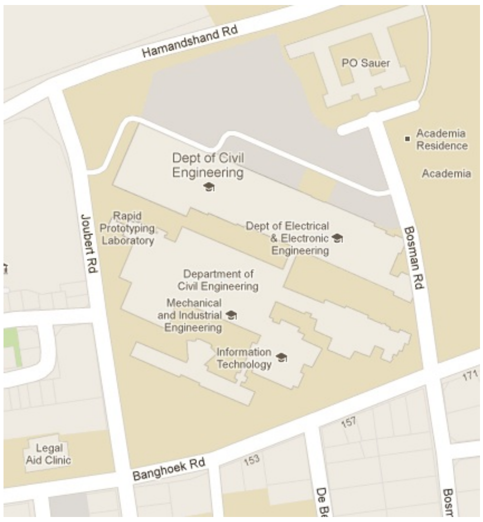

(c) Combined MRI- and SPECT scan

Figure 3.2: Registration of Multi Modal Medical scans [25].

The registration of aerial or satellite imagery is another common application of image registration [52] [49] [47]. Registering aerial images of forested areas offers a unique opportunity for environmental monitoring; satellite imagery taken over a certain timespan can be registered to give an indication of the rate of deforestation over a region. Also, the well know Google Maps utilises image registration to combine a vast amount of aerial and satellite imagery.

Images are also registered to know models, where an example in this context would be the aligning of an aerial photograph to a map of known roads and buildings. Figure 3.3 shows an image aerial image from Google Maps with its corresponding map shown beside it.


Hamandshand Rd
PO Sauer
Academia Residence
Academia
Dept of Civil Engineering
Rapid Prototyping Laboratory
Dept of Electrical & Electronic Engineering
Joubert Rd
Bosman Rd
Department of Civil Engineering
Mechanical and Industrial Engineering
Information Technology
Legal Aid Clinic
Banghoek Rd
171
157
153
(a) Street map

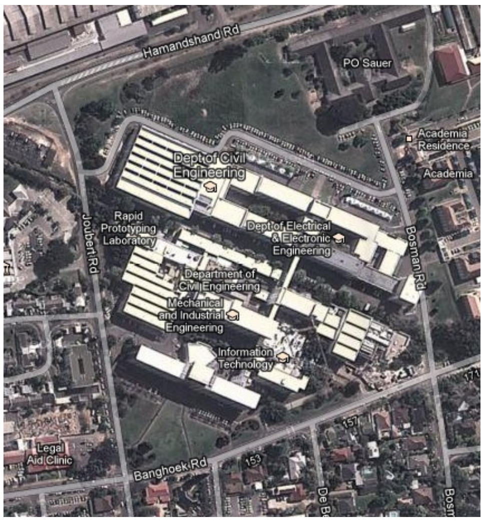
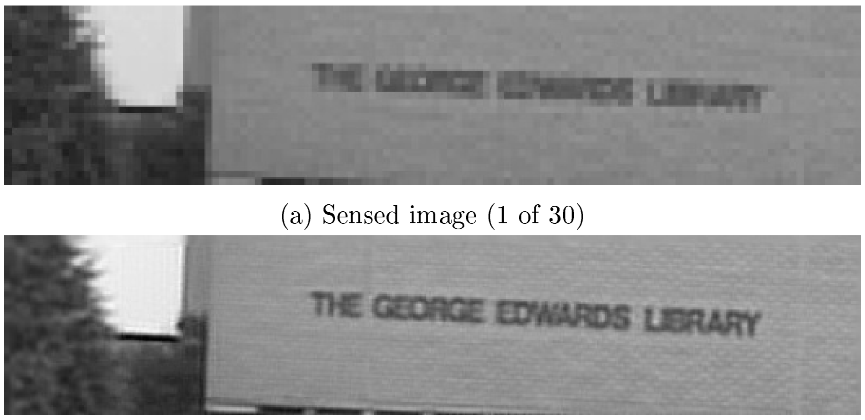

Hamandshand Rd
PO Sauer
Academia Residence
Academia
Dept of Civil Engineering
Rapid Prototyping Laboratory
Dept of Electrical & Electronic Engineering
Joubert Rd
Bosman Rd
Department of Civil Engineering
Mechanical and Industrial Engineering
Information Technology
Legal Aid Clinic
Banghoek Rd
(b) Satellite image

Figure 3.3: Registration of a street map to satellite imagery [17].

The last application of image registration discussed, is that of image super-resolution. With super-resolution several low resolution images from the same viewpoint are combined to form a single high resolution image estimation [42]. It is important to distinguish this from simple image mosaicking. With super-resolution the high resolution estimation goes beyond the capabilities of the image sensor. With simple mosaicking the highest spatial resolution captured is determined by the capabilities of the image sensor.

With super-resolution images are registered with sub-pixel accuracy. Due to slight movement in the image sensor, offsets are obtained that can be expressed as fractions of a pixel. This information at slight offsets allow the construction of an estimate of a scene that has a spatial resolution higher than that of the sensed images. Shown below is one of a series of sensed images 3.4a and the super-resolution estimation of the scene 3.4b.



(a) Sensed image (1 of 30)

(b) Estimation result

Figure 3.4: Super-resolution algorithms applied to range of sensed images [42].

The applications mentioned here are only a few of the uses of image registration. It is important to note that the applications are very diverse and that all of these examples requires (to different degrees) a tailored solution to achieve the desired results in terms of accuracy, speed and autonomy. In the next section some common image registration terminologies are mentioned and the typical steps involved with image registration discussed.

3.3 Process

As discussed in section 3.2, image registration has a wide range of applications, with most requiring a tailored solution. Regardless, there are still a common set of steps in the registration process that apply to most situations. These steps as described by Zitovae and Flusser [53] are illustrated in Figure 3.5.

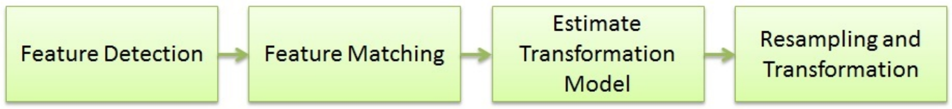


Figure 3.5: Steps of image registration.

The first step is Feature Detection. Here invariant features are selected through an appropriate algorithm or by a human operator. These invariant features are expected to be present and unaltered in all the sensed images. They are selected according to certain criteria that would have them identifiable in most of the instances in which they occur. After the features have been detected, they are matched between images to determine which are present in more than one. Sets of matching features over the range of images are then noted. With features matched, the transformation between images is estimated by calculating transformation between sets of matching features. Once the transformation has been determined, all the registered images are transformed (translated, rotated, stretched, scaled, etc.) to enable their corresponding features to overlap.

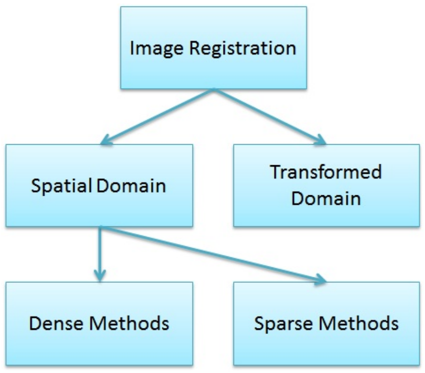


Figure 3.6: Approaches to image registration.

These steps form the base for most registration processes and outlines the similarity between different implementations. However, the similarity ends here. The steps outlined in figure 3.5 can be approached in a number of different methodologies in different domains.

A broad classification of these main approaches as described by van der Walt [42] are outlined in figure 3.6. Image registration can either be addressed in a the spatial domain, working with the coordinates as is, or in a transformed domain. A wide set of transformations can be utilised for image registration, e.g. conducting cross-correlation in the frequency domain rather than in the spatial domain.

Furthermore, the spatial domain approach can be divided into sparse and dense methods. Sparse methods operate by extracting certain features from an image and using them to estimate the transformation, while dense methods use all pixel intensity values in the estimation process.

In literature dense methods are also referred to as area- or intensity based methods while sparse methods are also referred to as feature based methods. These terms vary and the classification of a method is often deduced from the implementation.

In the next section these different approaches with regards to the typical steps in image registration are discussed.

3.4 Feature Detection and Matching

Depending on the methodology the feature detection and matching stages can be interdependent and are thus discussed together in this section.

The concept of invariant features were introduced in the example in Section 3.1 where the mountain serves as an invariant feature to detect and match. This is a simple task for a human operator to perform, however, a mountain is a complex object to recognize and for an automated algorithm it is preferred to specify a feature on a much simpler level.

The following two sections (3.4.1 and 3.4.2) describe the identification and matching of these invariant features with regards to dense and sparse methods.

3.4.1 Dense Feature Methods

Dense methods put emphasis on feature matching rather than detection. Explicit detection and extraction of features are avoided as the whole intensity map of an image is used during the matching. The specific location of features are not extracted, but, certain operations are still performed on the input images to highlight attributes. An example of such an operation would be to increase the contrast of an image, to increase the visibility of the features or to apply a smoothing filter to eliminate noise. Section 3.6.1 discusses this preprocessing further as typically applied to retinal images.

Matching for intensity based methods are achieved by using correlation-like methods. Some of these methods can be executed in the spatial, as well as in the frequency domain. In the following two paragraphs we discuss the spatial domain approach and then it's frequency domain equivalent.

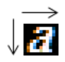
Spatial domain In the spatial domain the 2D cross correlation can be calculated by sliding a template image (or sensed image) (T) over a reference image (I); multiplying at each increment and summing the result. This operation is defined by eq 3.1.

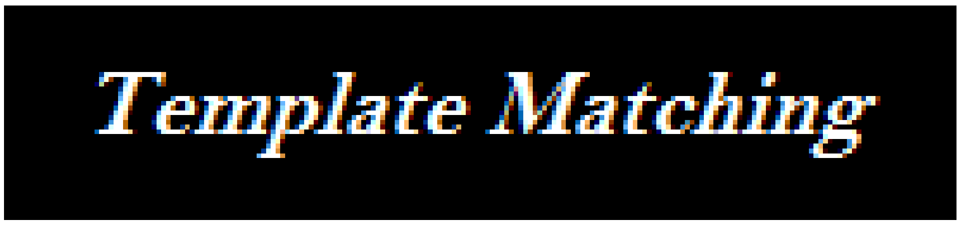
For images with a finite size, the result dimensions will be: $(I.width - T.width + 1; I.height - T.height + 1)$. The size reduction is as expected with kernel convolution in image-processing operations as elaborated on in Appendix A.5.

$$R_{ccorr}(x, y) = \sum_{x'y'} [T(x', y').I(x + x', y + y')]^2 \tag{3.1}$$

Spatial correlation is also referred to as template matching in image processing as it is commonly used to locate the occurrence of a small template in a larger image. It is typically reserved for small template images since its computation is considerably more expensive than its frequency domain equivalent when performed on larger images.

Figure 3.7 demonstrates the use of the operation to find occurrences of the letter 'a' in a paragraph. The cross correlation is calculated between the reference image (I) and the template (T) and the result thresholded to highlight prominent peaks. Each one of the peaks in Figure 3.7c represent the location of an 'a' character in the reference image.


(a) T



(b) I

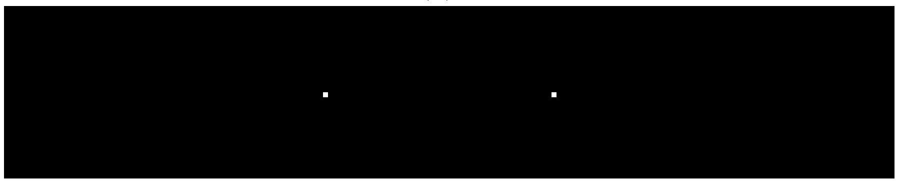
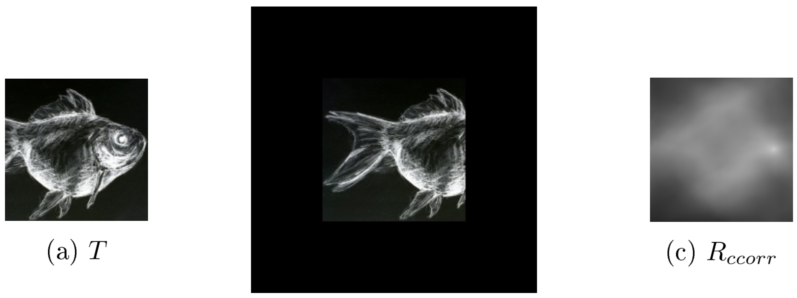

(c) Thresholded R_{ccorr}

Figure 3.7: Matching of a template to an image.

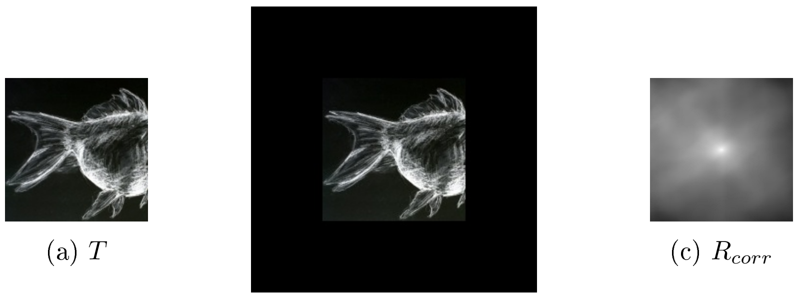
To apply template matching to find the offset between two images of the same size, the reference image has to be padded. There is a range of padding methods available: we demonstrate using zero-padding for simplicity. Figure 3.8 shows two cropped versions of the same image. Figure 3.8b is the padded reference image and figure 3.8a the sensed image. We note a single strong peak in the result shown in figure 3.8c. The peak is the location where the features align and the distance measured from the centre of the correlation to the peak is equal to the offset between the images.



(a) T (b) I (c) R_{ccorr}

Figure 3.8: Estimation of the offset between two images.

If the offset between the images were zero, the peak would be located in the centre of the cross-correlation map, i.e the location where the images correlate the most. Figure 3.9 shows the autocorrelation of an image to demonstrate this. Note the location of the peak in R_{corr}.



(a) T (b) I (c) R_{corr}

Figure 3.9: Image auto-correlation example.

A commonly used variation to the calculation of the cross correlation of two images is to calculate the squared error or distance between two images as given by equation 3.2 [3]. Here the obtained result is searched for a minimum, as this represents the offset between the two images where the error is minimized.

$$R_{sq\_diff}(x, y) = \sum_{x'y'} [T(x', y') - I(x + x', y + y')]^2 \tag{3.2}$$

Note that this method of sliding (translating) one image over another is only applicable if translation is the only transformation present between the two images. The performance of this algorithm declines rapidly in the presence of any other transformations like rotation or scale [53]. This shortcoming is discussed further in section 3.5.

Frequency Domain Cross correlation in the frequency domain is given by equation 3.3. Here $f(x, y)$ and $g(x, y)$ are the reference- and sensed images respectively and $\mathcal{F}$ denotes the discrete 2D Fourier transform. It yields similar results to the operation in the spatial domain. For larger images, calculating cross correlation in the frequency domain is more efficient than doing so in the spatial domain [53].

$$R_{ccorr\ normalised}(x, y) = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}(f(x, y)).\mathcal{F}(g(x, y))^*}{|\mathcal{F}(f(x, y)).\mathcal{F}(g(x, y))^*|}\right\} \tag{3.3}$$

Figure 3.10 shows the comparison between the frequency domain and spatial domain calculations. Note that results are similar with the peak found at the same location in both images.

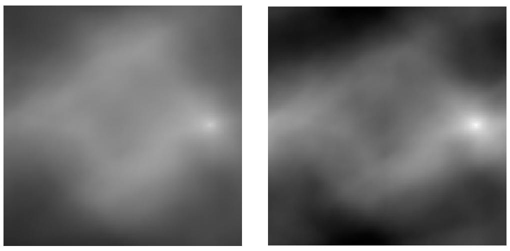

(a) Spatial domain cross-correlation (b) Frequency domain cross-correlation

Figure 3.10: Cross correlation comparison.

3.4.2 Sparse Feature Methods

A sparse feature detection algorithm aims to isolate a set of salient and invariant features. An invariant feature is a feature that remains unaltered when a certain set of transformations are applied to it [41]. The locating of invariant features that are immune against common transformations are invaluable to image registration as they can be used as anchor points when estimating the transformation between two images. These features are not necessarily important on a conceptual level and usually only consist of a pattern of pixels that are expected to remain constant over a range of sensing conditions and transformations.

Table3.1 lists some of the qualities that a good detected feature should adhere to, as described by Tuytelaars et al [41].

Quality	Description
Repeatability	An object or scene feature should be detectable from different views as well as in different viewing conditions.
Distinctiveness	A set of detected features should be distinguishable from each other in such a way that corresponding features from different images can be matched with ease.
Quantity	The quantity of detected features should be enough for the intended application. Depending on the registration problem, the number of required features can vary.
Accuracy	The location of the feature should be determined accurately and should not drift in respective sensed images.
Efficiency	Feature detection should be executed in a timely fashion. The execution time is dependant on the intended application and desired accuracy.

Table 3.1: Qualities of features selected by detection algorithm

There are a great number of detection algorithms available. In this section we simply note an example of one these algorithms, but do not discuss its working. The detection algorithm is strongly determined by the application. An application dictates the importance placed on the different qualities listed in table 3.1. For example, one could sacrifice accuracy for efficiency if the application only requires accuracy within a certain specified pixel tolerance.

An invariant-feature-detection algorithm should also take into account which transformations a feature should be more robust against. In an application where significant scale changes are expected, but not as much rotation, one would select an algorithm and parameters more robust to scale change than rotation. [41]

A robust feature that is commonly used for tracking, is corners. Corner features are not necessarily corners as in the traditional sense of the term. A corner is simply a large change in adjacent pixel intensities. Whether this relates to an actual corner, is not of concern. A corner is thus a region where a large gradient in the pixel intensities is detected. These gradient changes could constitute an actual 3D corner contained within the scene, edges in the scene, the boundaries of an object or simply a changing colour pattern on an object [41].

Below is an example of the popular Harris corner detection algorithm [20] applied to an image and a rotated version of the image. Detected corners are marked with blue circles. Note that the same features are selected, even though figure 3.11 (b) is subjected to a 90° rotation. The feature-detection algorithm is thus not sensitive to rotational transformation and would be suitable in an application where significant rotation is expected. The features in the image are also selected based on their distinctiveness as defined in table 3.1.

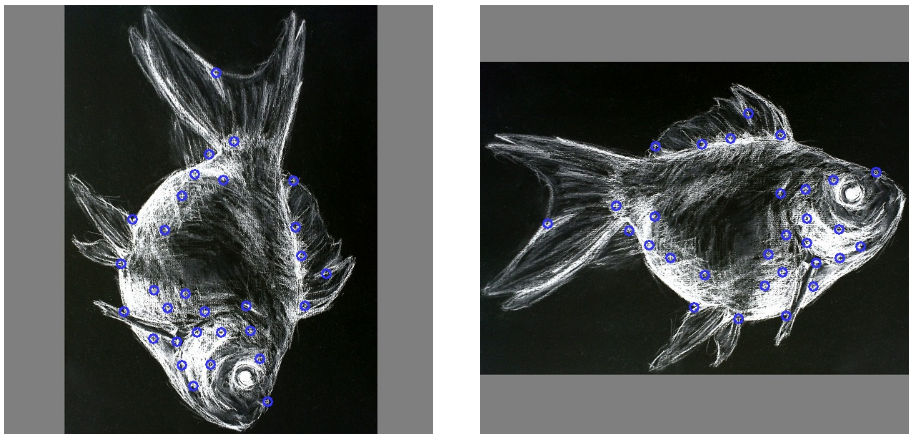

(a) (b)

Figure 3.11: Harris corner detector applied to rotated versions of an image.

Once salient features have been detected, they have to be matched between the reference and sensed images. Matches can be determined by applying cross correlation-like methods between individual features, as described in the previous section [53]. The cross correlation between large sets of features can be calculated rather efficiently since the image features are represented only by small patches of pixels.

This section briefly touched on feature detection and what a feature-detection algorithm should achieve. For an in-depth discussion on the subject, consult [41].

3.5 Transformation Estimation

After isolating or highlighting features, the transformation between them can be estimated. The estimation can be conducted in the spatial domain or a transformed domain. The frequency-domain phase correlation example in section 3.4.1 is an example of a transformed domain solution. As mentioned before, with the spatial domain, there are two approaches: dense and sparse methods [42]. Sparse methods (also referred to as feature-based methods) operate on extracted features while dense methods (also referred to as area or intensity-based methods) take all pixel values into consideration.

This section starts by discussing what a transformation function entails and how it is denoted. The discussion then continues onto the dense-and sparse transformation-estimation approaches respectively and ends with an optimisation technique typically used for dense-feature registration.

3.5.1 Transformation Function

Given two images of the same scene taken from different viewpoints, the pixels from the sensed image can be mapped to the reference image given a transfer function (h). This is illustrated in equation 3.4. The transfer function can be non-linear and very complex, depending on the distortion present between the images.

$$f_0(x, y) = f_1(h(x, y)) \tag{3.4}$$

Figure 3.12 shows a possible non-linear transformation applied to an image, however, we focus only on the linear transformations and the estimation of a linear transform when given two sets of corresponding coordinates.

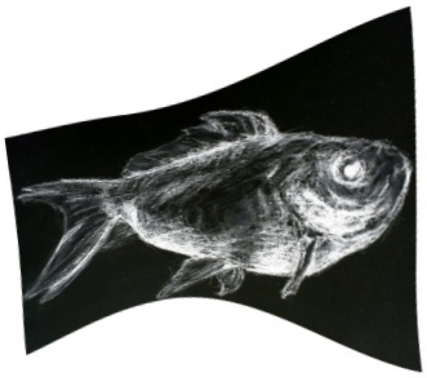


Figure 3.12: Non-linear spatial transformation applied to an image.

Figure 3.13 demonstrates a linear transform function applied to an image. The corresponding features in the two images are noted. Given the location of these corresponding features, the transformation between the set can be calculated.

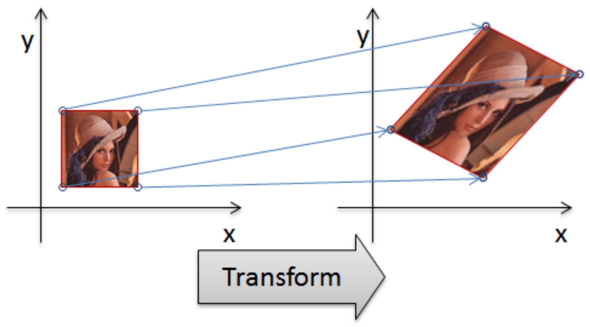


Figure 3.13: Template matching example.

A subset of linear transformations, called rigid transformations, are particularly of interest. Rigid transformations only represent translation and rotation. A key feature of rigid transformations are that they maintain straight lines and parallelisms between straight lines [53]. Figure 3.14 demonstrates the application of linear (affine) transformations to a square.

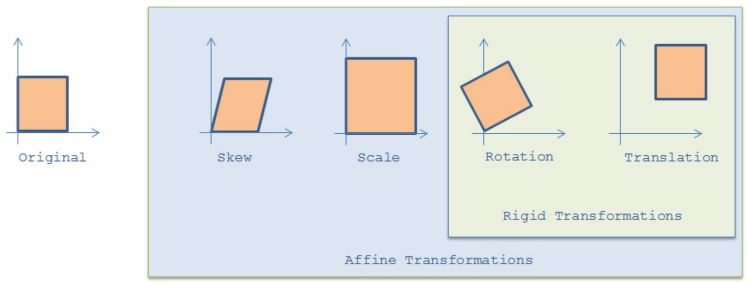


Figure 3.14: Linear transformations applied to a square.

Rigid transformations of point in a 2D space can be represented by a 3x3 matrix multiplication (as shown in equation 3.5) where H is the is the transformation matrix.

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = H \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{3.5}$$

To multiply a 2D coordinate with a 3 x 3 matrix, we write the coordinate in the homogenous form where $[x\ y]^T$ becomes $[x\ y\ 1]^T$. To convert a coordinate back from Homogenous to Euclidean we divide by z_0 [42]. When dividing by z_0, translation on the $Z - axis$ in a 3D space is converted to a scale change in a 2D space.

From equation 3.5 we deduce that the inverse of H can be used to get the transformation in the opposite direction as shown in equation 3.6. If H describes the transformation of the sensed image relative to the reference image, then H^{-1} describes the position of the reference image, relative to the sensed image

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = H^{-1} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.6}$$

The form of H is given by equation 3.7 where the rotation is denoted by θ and the translation by (t_x, t_y).

$$H = \begin{bmatrix} cos\theta & -sin\theta & t_x \\ sin\theta & cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

The matrix in equation 3.7 can be expanded to include other linear transformations like skew, scale and perspective, but is omitted here for simplicity.

Given a set of expected rigid transforms, we continue to estimate the transformation between two sets of coordinates. By inspecting equation 3.7, we note that the transformation matrix has three variables and would thus require a minimum of three matching points between reference- and the sensed images.

In the next section (3.5.2) the estimation of the transformation matrix using sparse methods is discussed. Estimating the transformation using dense methods is discussed in section 3.5.3.

3.5.2 Sparse Feature Transformation Estimation

There are a number of methods to determine the transformations between sparse feature sets. A method often used is that of error minimization, where the aim is to minimize the error between $(x_0)_e$ and $(Hx_1)_e$. Here x_e denotes the Euclidean form of the coordinate [42].

The shortcoming of a minimization of error approach is that it does not compensate for outliers that do not fit the estimated model. Outliers are a typical occurrence that happen when there are mismatched features present in the feature set.

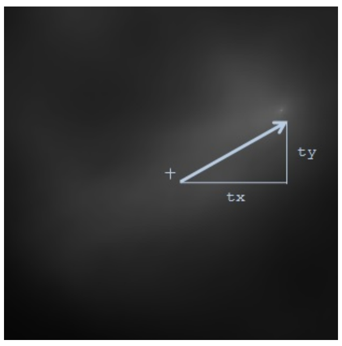
A popular approach to circumvent this problem is to use RANSAC (Random sample consensus). The problem arises since it is not known which features are inliers and which are outliers. With RANSAC a random subset of features are selected and assumed to be inliers. The number of feature pairs selected is equal to the minimum amount required to estimate all the parameters of the transformation model. For the transformation model in equation 3.7 the minimum is three. The transformation is estimated using these inliers with a method such as the error minimisation. The estimated transformation is tested against all features and inliers to the model are determined using a certain threshold. The accuracy of the model is then determined by calculating the error of all the inliers. This process is repeated for a predefined number of iterations. At the end of the iterations the model with the smallest error to all its inliers is selected as the best fit [12].

There are many variations of the RANSAC algorithm, as well as a vast number of other methods for the transformation estimation of a sparse feature set. They are not discussed here and the RANSAC method is simply offered as an example of a typical approach.

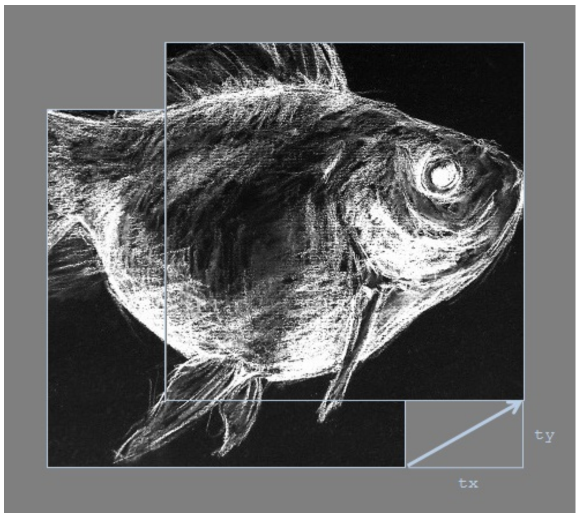
3.5.3 Dense Feature Transformation Estimation

Registering dense features, offers its own unique obstacles. In this section we discuss the increased execution time typically incurred by dense feature registration and discuss solving dense feature registration in a transformed domain for rigid transformations.

As discussed in section 3.4.1, cross-correlation- or error-minimisation methods can be executed by sliding the sensed image over the reference image and determining the offset where the best fit is achieved. Figure 3.15a shows the cross correlation result between two images (The reference image was zero padded to obtain the result, but the padding is omitted in the figure). The vector from the centre to the peak of the cross-correlation is equal to the translation between the reference and sensed image as illustrated in Figure 3.15b. This offset vector also describes the t_x and t_y components of the transformation matrix in equation 3.7.



(a)



(b)

Figure 3.15: Estimating translation between sensed- and reference image using cross correlation.

A limitation of the sliding template approach is that it only caters for translation between the reference- and sensed images. To extend this algorithm to include rotation, one would have to slide as well as rotate the sensed image to every anticipated position and calculate the cross correlation. This approach is not practically feasible in most situations since the execution time would increase exponentially for each additional transformation that has to be accounted for. It can only be used in a setting where a very limited number and range of transformations are expected, for instance, the situation where full translation, but only a range of rotation of 5° is expected. Here the image would be translated to all possible positions, but only rotated over a small range with a fixed increment.

An approach to achieve dense registration where full rotation and translation is expected was proposed by Castro and Morandi [9]. The approach solves the problem in a transformed domain. If an image is converted to a domain independent of translation and only sensitive to rotation, the rotation can be estimated in this domain. The estimated rotation can be used to correct rotation in the spatial domain and the translation solved using any other methods.

To apply their method, a key observation is made: The magnitude of the Fourier transform of an image is only sensitive to the rotation and not to the translation of the original images. A translation in the spatial domain only causes a phase shift in the frequency domain while the magnitude remains unaltered. However, a rotation in the spatial domain also causes a rotation in the magnitude spectrum.

One also notes that a rotation in a Cartesian coordinate system becomes a translation in a Log-Polar system where the mapping from Cartesian to log-polar is given by equation 3.8 and illustrated by figure 3.16.

$$\rho = log\left(\sqrt{x^2+y^2}\right), \quad \theta = tan^{-1}\left(\frac{y}{x}\right) \tag{3.8}$$

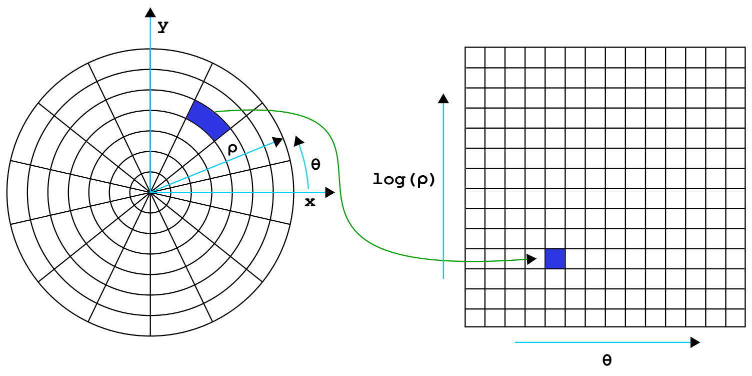
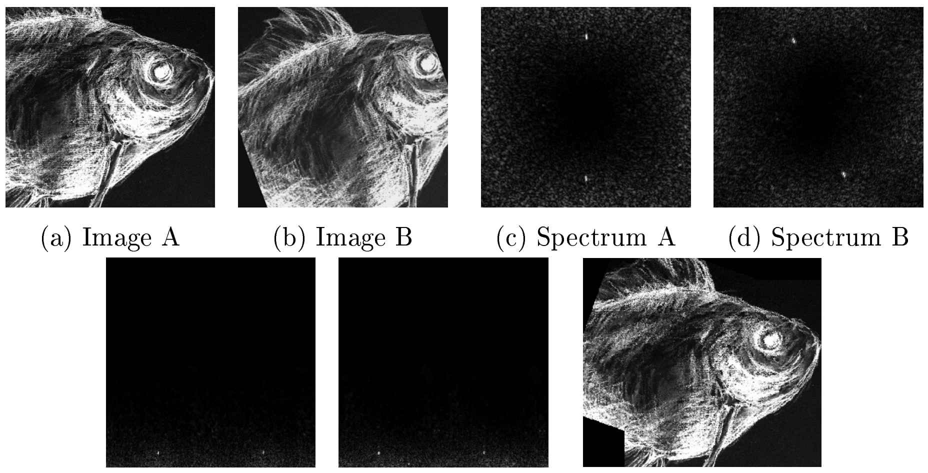


Figure 3.16: Mapping a Cartesian image to the log-polar domain [36].

If the spectral magnitude of the image is converted to a log-polar coordinate system, the Fourier-Mellin domain representation is obtained. It is then found that a rotation of an image in the spatial domain results in a translation in the Fourier-Mellin domain. And also that a translation in the spatial domain has limited effect in the Fourier-Mellin domain. As long as the translation does not cause significant clipping of the images.

Figure 3.17 shows two overlapping sections of the same image. The section in figure 3.17b is rotated by 20°. Below the images and their respective spectral magnitudes are shown in figure 3.17c and 3.17d. Note the 20° rotation between the two spectrums. The log-polar plot of the spectrums are shown in figure 3.17 and it is noted that rotation has been converted to a Cartesian translation.


(a) Image A (b) Image B (c) Spectrum A (d) Spectrum B

(e) Fourier-Mellin A (f) Fourier-Mellin B (g) Combined images

Figure 3.17: Registration in the Fourier-Mellin domain.

The translation in the Fourier-Mellin space can be estimated using the conventional cross-correlation- or error-minimisation methods described earlier. Once the rotation is estimated, the rotation of the sensed image can be corrected in the spatial domain. And after the rotation is corrected, the translation is estimated in the spatial or frequency domain.

3.5.4 Optimisation Technique

The template matching approach (typically used for dense feature registration) has the main drawback of the exponential increase of execution time as the number of expected transformations increase. An attempt to optimise the process is to conduct image registration at multiple resolutions. Starting at a low resolution- and working towards to a high resolution estimation. This process is often achieved with the use of image pyramids. An image pyramid is the deconstruction of an image into its components at different resolutions. Figure 3.18 demonstrates this deconstruction of an image.

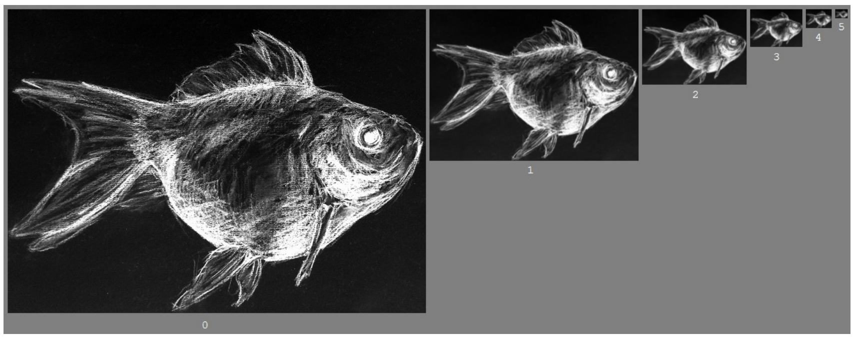
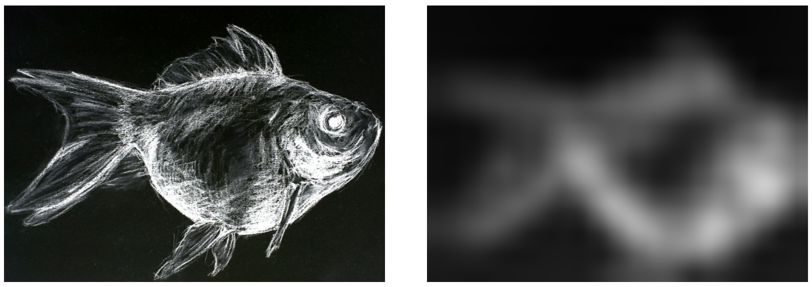


Figure 3.18: Image pyramid representation.

The top level (*level*5) of the pyramid contains the largest features present, and the bottom level (*level*0) contains the smallest features of the image. The high-resolution features can be considered as the high-frequency features and the low-resolution features as low-frequency features from a signal processing point of view. Figure 3.19 shows *level*0 and *level*5 of the pyramid scaled to the same size. Note how *level*0 contains all the detail of the image and that *level*5 only describes the general shape of the of the content.



(a) Level 0 (b) Level 5

Figure 3.19: A comparison of an image to its low resolution descriptor.

A number of different methods can be used for the deconstruction of an image into its pyramid. However, the general process remains the same. An image is subjected to a succession of filters and down samples. The pyramid in figure 3.20 is a Gaussian pyramid. Here a Gaussian blur is applied and then the image is down-sampled by a factor of two. Each level yields lower resolution information contained within the image. The deconstruction can be repeated until the lowest desired resolution is achieved. Refer to appendix A.5 for a discussion on filter operations and the Gaussian filter in particular.

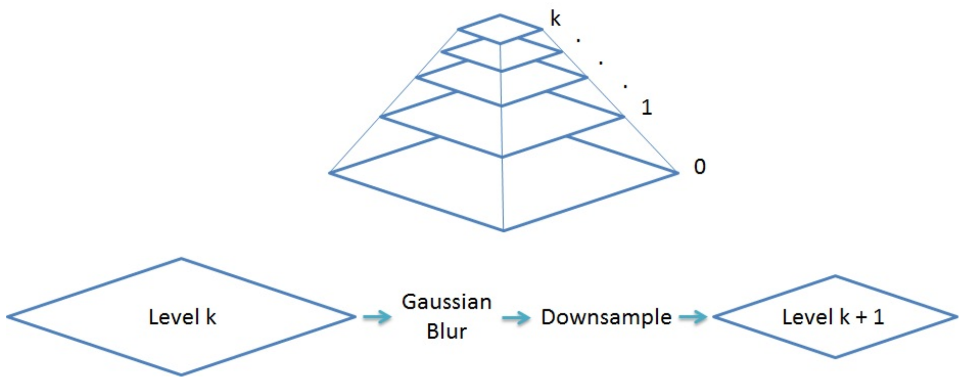


Figure 3.20: Deconstruction of an image into a Gaussian pyramid.

To register two images, the sensed- and reference image are first deconstructed into their respective pyramids. A coarse to fine registration is applied to images by starting at the highest level ($levelk$) of the pyramids. The highest levels are registered and the transformation between them estimated. Since the pixel resolution on these levels are very low, the estimation is calculated rapidly for a number of assumed transformations. The low resolution estimation is upscaled and serves as a starting point for registration at the next level. This process is repeated until registration at the highest resolution is achieved. At each level the template matching is only conducted in the region of the estimate provided by the previous.

This approach greatly increases the execution time of an exhaustive search for the transformation between two images. One disadvantage, however, is that an error at a low resolution can be relayed to the high-resolution estimate. This occurs when data at a low frequency correlates at a different location than that of the higher frequencies contained in the image. The high-resolution estimate will then be based on the low resolution error.

Image pyramids are not only reserved for dense methods but can be utilised with sparse methods as well. With a sparse feature set, the more prominent features are registered first and then the algorithm moves onto the detection and registration of features at a higher resolution.

3.6 Retinal Image Registration

Retinal image registration follow the general steps of image registration as described in the previous section. Features that are to be tracked are selected and matched between images and the transformation between them are estimated.

The invariant features to track in a retinal image, are typically the blood vessels and the optic disk since they are clearly visible and not expected to change in shape. Other areas of the retina consist of large uniform regions with no visible landmarks to track.

Retinal image registration has two areas generally addressed in literature. The first is the segmentation of the blood vessels from the rest of the observed areas and the second is the unique variations in the registration methods that are required because of the differing appearance of retinal images captured under different conditions.

3.6.1 Blood Vessel Segmentation

Blood vessel segmentation of retinal images is a problem commonly addressed, not only for its use in image registration, but also in its added value in diagnostic purposes. Abnormalities in the shape of the blood vessels can indicate the conditions discussed in section 2.2.1 and automatic segmentation of the vessels in retinal images is often suggested to aid the physician or perform an automatic diagnosis [50], [30] and [1]. Here, the extracted blood vessels, could serve as input to a diagnostic application where classifiers automatically determine if a patient should be referred to a physician for additional screening.

Numerous methods, which range in complexity and efficiency, have been suggested to achieve this goal, we briefly discuss some of them in this section. The output of the segmentation algorithm is typically a binary mask indicating highlighted vessels.

A common suggestion for segmenting blood vessels, is to select the green channel of a retinal image as vessels are more prominent in this channel [48] [26] [15]. Shown in figure 3.21, is a retinal image separated into its red, blue and green colour channels. Note the higher contrast between the blood vessels and the background in the green channel.

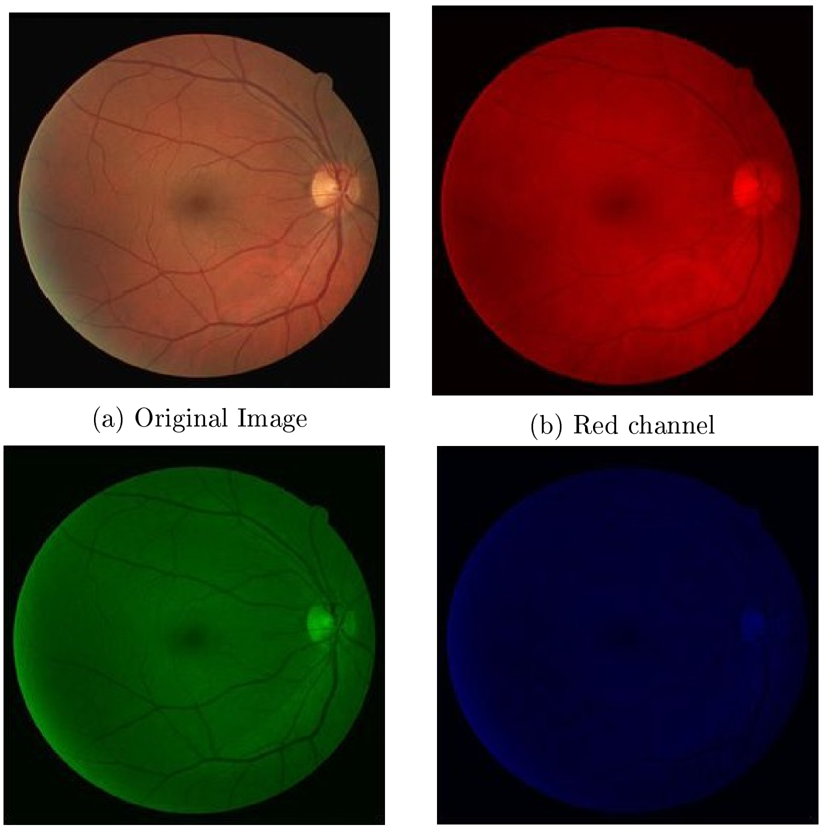

(a) Original Image (b) Red channel (c) Green channel (d) Blue channel

Figure 3.21: A retinal image divided into its red, green and blue colour channels [37].

The intensity map of the green channels is now used to segment blood vessels from the background. The methods for achieving this are diverse and we describe general approaches while focusing on the preprocessing aspect.

The exact implementation and combination of operations vary between situations. Preprocessing is concerned with increasing the contrast of the vessels and the removal of noise and interference. Typically preprocessing is applied before more complex operations and classifiers are applied to extract the vessels[48] [37]. For a more in depth overview of blood vessel-segmentation techniques, consult Mabrouk et al [29].

Filtering Filtering is ubiquitous in image processing. It is employed to isolate a signal from its background, reduce noise or remove any known interference. In segmenting blood vessels, it is mainly used in two ways: The first is for the removal of known interference in retinal images and the second is the highlighting of vessels using matched filters.

Most retinal images suffer from what is known as vignetting. Vignetting is strong central illumination of an image that decreases radially outwards from the central camera axis towards the sides of the captured image [21]. Figure 3.22 shows a model of a vignette pattern.

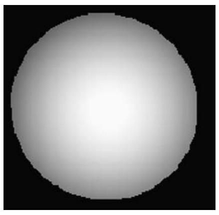


Figure 3.22: A model of vignetting [27].

Vignetting has the consequence of blood vessels around the centre of a retinal image to have a higher contrast against the background than those around the edges. Non-uniform illumination and contrast poses a problem to a dense feature registration algorithms since it can cause corresponding features to appear dissimilar in different images [6].

A simple solution is to normalise retinal images through the application of a high-pass filter. [48] uses a median filter to estimate the background illumination and subtracts this from the original image. Since the median filter is a smoothing filter, the result of the operation is the highlighting of any high-frequency content and the discarding of low-frequency components.

Figure 3.23 shows the result of the operation. Here the retinal image contains non-uniform illumination and a varying-background intensity that is removed. The result is then inverted to have only the vessels remain on a dark background.

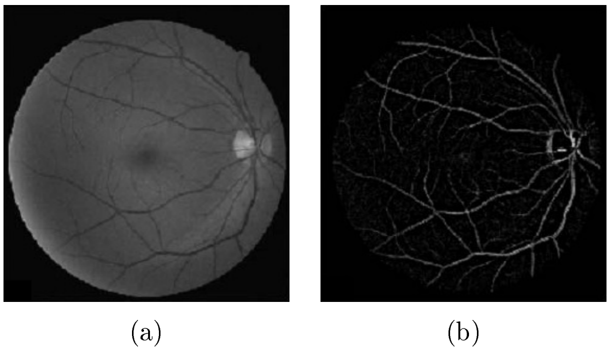


Figure 3.23: Retinal image with varying background illumination removed [48].

The second use of filters in retinal image segmentation is the use of matched filters. A matched filter is a filter that resembles a certain feature in appearance. By convolving with the filter, these matched features are highlighted.

For retinal images, matched filters can be constructed that resemble blood vessels and can aid in their extraction. An observation is made that the profile of a blood vessel resembles a Gaussian curve [7]. A matched filter is constructed that estimates the shape the vessels with a Gaussian curve. This curved is rotated to different orientations to match vessels at different orientations.

The Gabor filter is typically employed to achieve this goal as demonstrated in [37]. The Gabor filter consists of a Gaussian modulated sinuous function as described by equation 3.9. The intensity map of the filter is shown in figure 3.24. Here the filter has a bright ridge surrounded by dark valleys. The blood vessels can be seen as piecewise linear sections that match with the filter at different orientations.

$$g\left(x,y\right)=e^{\left[-\pi\left(\frac{x_p^2}{\sigma_x}+\frac{y_p^2}{\sigma_y}\right)\right]}\cos\left(2\pi x_p\right)$$

$$x_p = x\cos\theta + y\sin\theta$$

$$y_p = -xsin\theta + y\cos\theta \tag{3.9}$$

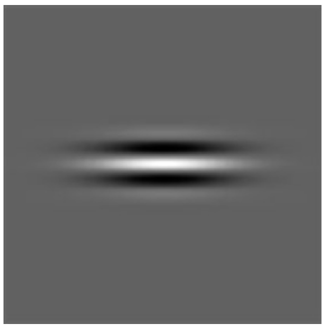
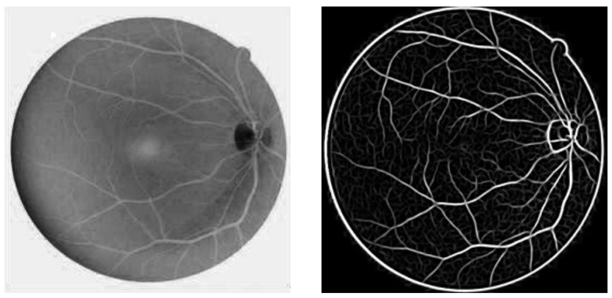


Figure 3.24: 2D Gabor kernel.

[37] Demonstrates the response of an inverted retinal image to the application of a Gabor filter. Here, an estimate to the average blood vessel width is made and used to construct the Gabor filter. The result is shown in figure 3.25.

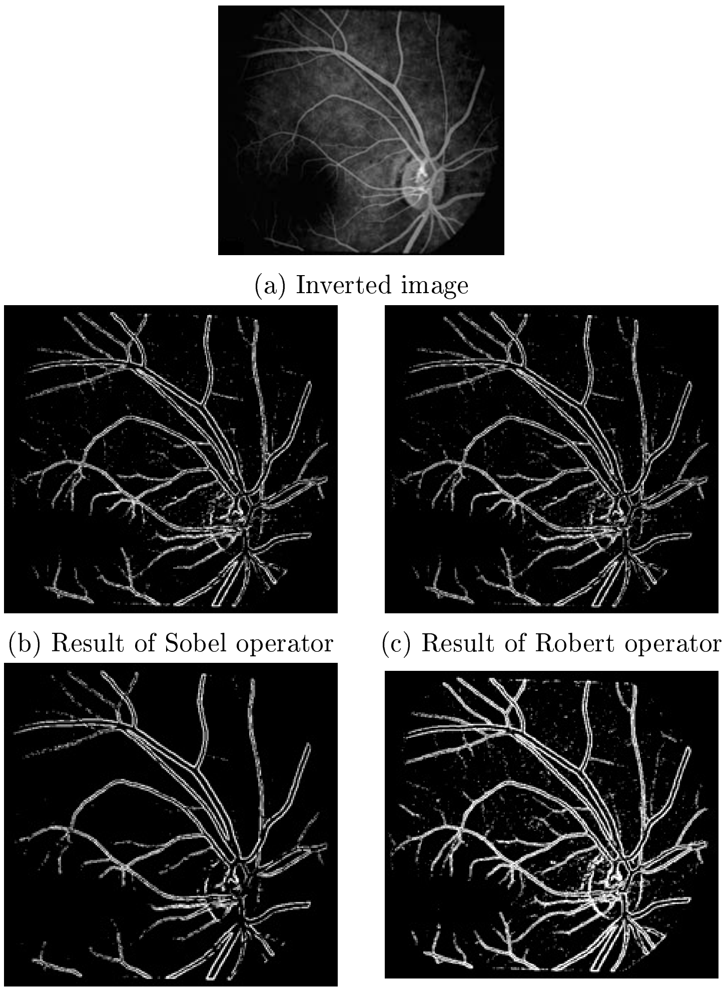


(a) Inverted retinal image (b) Blood vessels highlighted using Gabor filters

Figure 3.25: Highlighting of blood vessels using a Gabor filter [37].

These filter operations are used for preprocessing an highlighting the features. After the vessels are highlighted, either a threshold is applied to create a mask, salient features are extracted for registration, or a graph representing the vessel structure is estimated.

Difference Operators Difference operators are convolution kernels that calculate the gradient components of an image at each pixel. These convolution kernels highlight pixels with high gradients against their neighbouring pixels. A high gradient in a signal translates into a high contrast in the image. By examining the gradients contained within a retinal image, an estimate of the edges can be made. The contrast between a blood vessel and its surrounding background is high and a high gradient is expected at their boundaries. By thresholding the gradient intensities the boundary locations of the vessel are determined. These operations typically yield a binary mask, indicating where edges are located. Many edge detection operators exist and we simply note the result of applying a selection of them to a retinal image in figure 3.26. Consult [29] for a discussion on the different operators with regards to blood vessels segmentation.



(a) Inverted image

(b) Result of Sobel operator

(c) Result of Robert operator

(d) Result of Prewitt operator

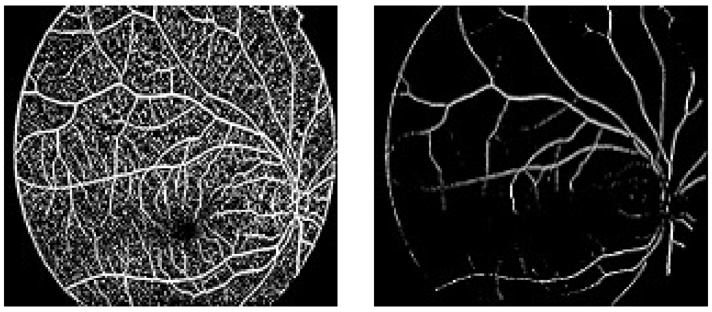
(e) Result of Canny operator

Figure 3.26: Highlighting of retinal blood vessels using various edge detection methods[29].

Morphologic Operators Morphological Operations, discussed in Appendix A.5, is often employed in blood vessel segmentation and is used in two distinct ways. The first is in gradient detection. The operation serves as a morphological method of edge detection. Here the result of an erode operation is subtracted from the result of an opening operation. The result of the subtraction is thresholded to yield strong edges in the image.

The other application in retinal images is the removal of interference in thresholded images. The binary mask, obtained by thresholding a preprocessed retinal image, typically contains noise. The noise is different in structure from the blood vessels in the mask and can be removed using the morphological open operator. The open operator consists of a successive erode and dilate operation and will remove small unconnected regions in a binary mask. The erosion shrinks all structures and in the process eliminates the smaller structures.

The following dilation restores the remaining structures to their original size. Figure 3.27 demonstrates the use of the operator. Note that the large blood vessels are maintained and other smaller structures are removed.



(a) Image containing noise (b) Noise removed

Figure 3.27: Highlighting of retinal blood vessels using morphological operations [29].

3.6.2 Registration Methodologies

For retinal image registration, the main source of landmarks are the extracted or highlighted blood vessels extracted, as discussed the previous section.

Common problems to the registration of retinal images arise and have to be addressed. The first is that small overlaps between images tend to have too little information to estimate the translation between the images. This is especially true when calculating the higher order translation between the images, since the transformation model for the overlapping region might not be applicable for the whole image. Second is that high-resolution images contain large areas that do not contain significant landmarks [6].

The methodologies vary greatly for different retinal registration applications, with each tailored to cater for a specific need. In the following section we briefly discuss the use of dense- and sparse-feature registration.

Dense- and Sparse Methods Earlier studies in the registration of retinal images suggest the use of dense registration methods. [33] Discusses the registration of retinal images on what would be considered limited resources by current standards. Due to the increased computational power required, methods of optimisation when using correlation is also elaborated on. Minimization of error as a similarity measure is preferred over cross correlation. The reason being is that cross correlation entails more multiplication and division operations that are more processor-intensive than subtractions.

The other optimisation is the use of a multi-scale approach calculating the downscaled translation and using the result as an estimate for calculating the upscaled solution. The last optimisation, they suggest, is a threshold for early termination of calculating the similarity at a specific offset. If the cumulative error exceeds the threshold, the similarity calculation is abandoned for the position and the template is moved to the next location. This threshold could be a predefined value of be specified as the current minimum error found during the similarity calculations.

Recent methods tend to be more sparse feature based with literature covering various novel methods for the detection of salient features [5] [16] [51] [28].

Some methods operate on the highlighted blood vessels while other aim to isolate invariant salient features that can be identified regardless of variations in background illumination [14].

Because of the nature of retinal images, there is no general solution to all problems. [6] Suggests the use of a hybrid approach. They employ sparse feature methods for regions that contain easily identifiable areas, such as the regions around the optic disk and larger blood vessels, and employ correlation methods in areas that are low in features like the areas around the macula and between blood vessels. They employ dense feature methods to calculate the 0th order offset between and then calculate the higher order offset based on the offset of local features.

In most cases of sparse feature registration methods operate on images with a sufficiently large field of view. The areas covered by blood vessels are sufficient to calculate the offset between images and yield a estimation for areas that do not contain visible landmarks. The macula does not contain as many features, but is surrounded by enough blood vessels to achieve accurate registration for it.

Chapter 4

Proposed System

4.1 Setup

A system for the capturing and processing of an ocular fundus examination is proposed. For capturing, the Welch Allyn Pan optic ophthalmoscope is combined with a mobile phone. The camera of the mobile phone is aligned with the eye cup that the examiner would use, allowing the camera to capture imagery that is usually observed by the examiner. A live feed of the imagery is displayed on the screen of the device allowing the examiner to direct the device while the mobile phone records the procedure. The physical setup is discussed in greater detail in Appendix D.

A video of the examination is recorded on the mobile device and the videos transferred to a PC where relevant data can be processed. Sections of the video that contain valid fundus imagery are isolated analysed. The isolated sections are then registered with regards to one another and a composite image of the fundus is formed. The algorithms and their design are discussed in detail in Chapter 5. These algorithms and their tailoring for the specific application are the main focus of the research.

4.2 Motivation

Two main factors motivate the proposed implementation. The first is the ability of enhancing digital imagery from an analog device to aid diagnosis. This enhancement is not possible for a pure analog device. The second is the use of digitised imagery for remote diagnostic purposes.

The next two sections discusses the benefits of enhancing the data and the need for digitised data for use in remote diagnosis. Thereafter, the cost benefits of the proposed system are discussed.

4.2.1 Analogue Scope Enhancement

The direct handheld ophthalmoscope is a portable battery operated device that is often used in the field for monitoring patients. The main drawback is that these ophthalmoscopes are usually analog devices with a limited image quality compared the larger, stationary fundus cameras.

Digitising the imagery captured by the device, greatly enhances the diagnostic capabilities of the data captured as possessing a digital copy allows for highlighting and aggregation.

An example of this aggregation is to increase the field of view. The FOV refers to the total area visible in a single image. The FOV of the handheld device can be increased by collecting a number of images that cover a broader area of the fundus and combining them. Information from several images can also be combined to enhance the image taken for a specific area. When a signal has a large SNR, but several copies of the same signal exists, the copies can be super-positioned to yield a resulting signal with a lower SNR.

Another benefit of having digitised information is the ability to highlight certain features within the data. This can be done to emphasise areas of interest. As the case with retinal images, they are often enhanced to highlight the blood vessels as irregularities in their structure is used to diagnose disease.

Other benefits of obtaining digital data are the inherent benefits typically associated with digitised records. The sharing and storing of previously recorded data, allow the tracking of the development of certain conditions over time. Information can also be shared between healthcare practitioners operating in remote locations. This remote monitoring and diagnosis of disease, is often referred to as telemedicine.

The next section briefly discusses the benefits of telemedicine in a South African setting.

4.2.2 Remote Diagnosis

The South African public healthcare system is over burdened with a large number of patients being tended to by only a limited number of healthcare practitioners. Often the burden is shifted to primary-level practitioners to perform the first level of care and then refer patients to a physician if further assistance is required. These primary-level practitioners perform invaluable work, but often lack the skills to conduct proper monitoring of chronic deceases [40].

In these situations telemedicine, is suggested as a viable option. Telemedicine, per definition, is the use of information and communication technologies to relay consultation information to a physician capable of making a diagnosis of data collected by a primary-level practitioner [34]. Suggestion of the physician is relayed back to the patient or the primary-level practitioner using the same communication channels. It is employed in rural settings where access to proper medical care is difficult to obtain.

The proposed solution offers a lightweight, cost-effective device that can easily be used in the field and the proposed software algorithms offer a method of condensing the data before sending it to a physician to inspect.

The data is condensed to reduce the amount transmitted over a bandwidth restricted channel, since the mobile coverage in a rural setting is very limited. Since mobile coverage is often the only available connection to the internet, it is often the only method of communication for telemedicine applications in rural areas.

In rural areas the mobile data coverage is often restricted to the slower GPRS and EDGE service. In figure 4.1 the mobile data coverage of a leading South Africa cellular provider is shown. Note that this map indicates weak to no coverage in rural areas. The light-red areas indicate GPRS and EDGE connections. In these areas a stable data connection would exist, however, it would be slow when relaying multimedia content such as videos or images.

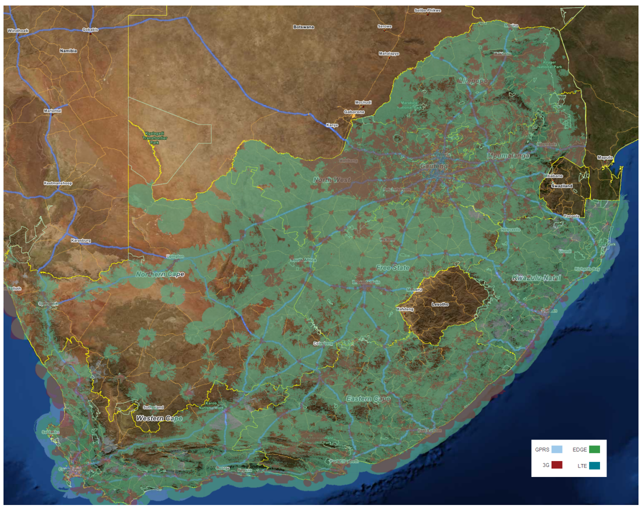


Figure 4.1: South African mobile coverage map of a leading cellular provider [44].

Although telemedicine is a possible use of the setup, the main focus of the research is on the development of the algorithms to enhance the quality and condense the captured data. The relay of the captured data is not addressed since numerous systems to achieve this is already the focus of a number of studies.

4.2.3 Cost Benefit

Field of view, is one of the important attributes of an ophthalmoscope or fundus camera. FOV indicates the maximum area of the eye that can be viewed through the device at a time. In practice, the FOV values only range between 5° and 60° with schematic in figure 4.2 demonstrating three commons FOV's.

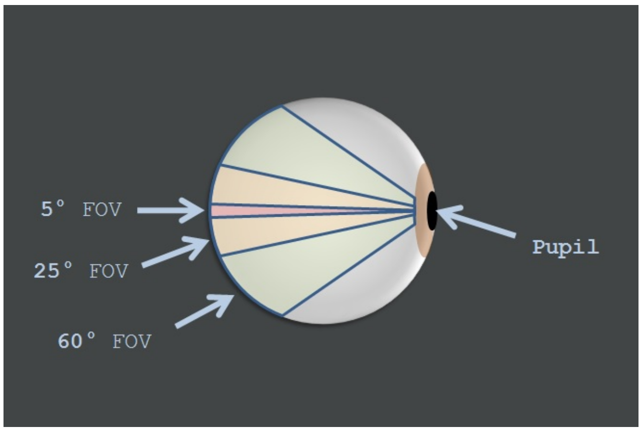


Figure 4.2: Typical FOVs from different fundus cameras and ophthalmoscopes.

Section 2 elaborated on the structures in the eye and their importance. Note that a larger FOV is more favourable for diagnostic purposes since it allows for a better view of these structures.

The limiting factor when selecting an appropriate device is that cost increases exponentially as the FOV increases. Figure 4.3 shows the plot of price versus FOV for ophthalmoscopes and fundus cameras. The prices in the plot are estimates for specific devices on the market compared to their advertised FOVs.

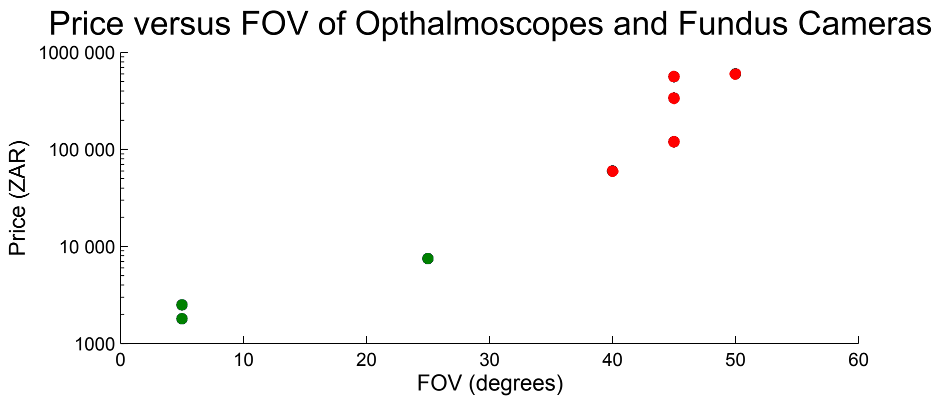


Figure 4.3: Price versus FOV of fundus cameras and ophthalmoscopes

The devices with a larger FOV (indicated in red), also have an image sensor to capture the visuals and often an electronically-driven lens system that can be guided by the physician. These two factors also attribute to the significant increase in the cost of the devices.

The proposed setup offers two distinct benefits in terms of cost. First is the benefit of digitising an analog device to obtain digital images at a reduced cost. Second is the ability to use the limited FOV images from a cheaper device to construct a larger FOV image. A handheld analog device with an image sensor attached would then emulate the results of the larger, and more expensive fundus camera.

Chapter 5

Processing and Registration of Captured Imagery

5.1 Overview

This chapter discusses the design of the image-processing algorithms. The basic flow of our implementation is illustrated in figure 5.1.

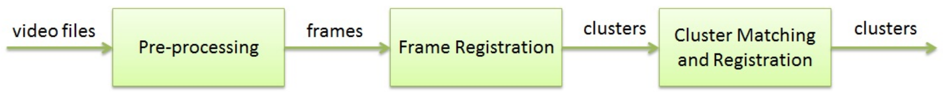


Figure 5.1: Steps in processing the examination video files.

The input to the system, is the videos captured during an examination. A typical examination video is described in the section 5.2. A retinal exam is a difficult process and a significant portion of the data captured, will not be usable and not contain valid retinal images. Section 5.2 also elaborates on obstructions typically encountered and the resulting quality of the imagery captured.

Due to the low quality of the imagery, an efficient selection process has to be devised to select frames from the video sequence that contain retinal images and can be processed further. Once candidate frames are selected, they have to be enhanced to highlight certain features. The features to be highlighted are the blood vessels as they will serve as the invariant features to align the frames by. Section 5.3 discusses these two aspects in detail. The output of the processing is a large number of frames. These frames correspond to a list of the *Frame* class of the code discussion in Appendix C.

Once the appropriate invariant features have been isolated, the video frames can be registered. The frames produced by the preprocessing stage are assumed to be consecutive and assumption is made that each frame has a close-to-zero offset with respect to its immediate predecessor. This yields an initial offset estimate and is leveraged to perform the bulk of the registration efficiently. The registration is conducted using spatial correlation methods and the frames are grouped into clusters. These clusters correspond to the *Cluster* class discussed in Appendix C. Section 5.4 discusses this approach in depth.

Once the frames have been grouped into clusters, the clusters are matched with each other and combined to form larger clusters. The set of clusters to be registered are in no particular order and it can thus not be assumed that consecutive clusters also contain mutual information. Other methods of optimising the searching for matches are devised. Section 5.5 discusses an effective method of matching random clusters in a large set. When a matching pair of clusters are isolated, they are combined using spatial cross correlation.

At each step of the registration the cross correlation has to be verified. Different methods for verification are suggested for the registration of sequential frames than for cluster registration. For the sequential-frame matching the strength of the correlation peak is utilised to split sequences of frames into clusters. When matching random clusters, the correlation is inspected to determine if a valid overlap of features have been found. Section 5.6 discusses this in further detail.

Throughout this chapter, the spatial cross-correlation method of determining the offset between two images is utilised. Refer to section 3.5.3 for a discussion on this method. This chapter, will rather, focus on methods of optimising and verifying the results of the spatial cross correlation for the specific application.

We assume no rotation and scale changes between images and do not develop the algorithms to take this into consideration. However, where applicable, it is mentioned how the algorithms can be modified to take rotational changes into consideration.

5.2 Captured Data

Inspecting the examination video is imperative for the design of algorithms that extract the data. Certain attributes of the video is caused by the physical setup and affects the way in which the data can be interpreted and extracted. Anticipating certain characteristics necessitates the use of custom registration methods. By tailoring these algorithms optimisation towards this specific application can be considered.

In this section, some of the characteristics of the video are discussed and in later sections refer to how the algorithms are adapted to suit the nature of the captured data. A full length video of a typical examination using our setup is available here [43].

The first attribute noticed in the recordings is that for a significant duration subjects other than the desired fundus, is captured. A portion is dedicated to aligning the camera with the patient's eye and an attempt to get a clear view. In other portions the view of the retina is obscured by interferences.

The long duration of the videos to be parsed prompted the addition of the push button (discussed in Appendix D) to allow the examiner to only record required sections of the process with little additional interaction with the system.

Another attribute that is noticed, is that a significant section of each video frame contains black region not recording any data. The information in each frame is contained within a clipped regions that is caused by the viewing field of the ophthalmoscope. These regions can be discarded without further inspection.

Vignetting is another phenomenon that causes a distortion. Vignetting is the curved illumination commonly seen in retinal images. This curved illumination is caused by the strong directional lighting and its strong falloff. The consequence is that the brightness, as well as the contrast, decreases from the centre of the frame towards the sides. This decrease in contrast and brightness increases the difficulty of distinguishing the blood vessels from the background.

In general, the video also has low contrast and therefore a low SNR. This is due to the overall low light exposure used in the examination. The low light exposure is used to avoid excessive blinking, tear formation and further pupil constriction. The low intensity of the signal poses a significant drawback since it prevents the identification of smaller blood vessels.

During the examination two, obstructions are typically present. The side of the pupil is often captured if the alignment between the ophthalmoscope and the patients eye is not perfect or the patient moves his eye. The other obstruction is caused by lens flare. Lens flares cause significant difficulties since even a perfect alignment of the opthalmoscope can be compromised if flare is still obstructing the view of the fundus. Lens flare occurs because of internal reflection between lenses. In this instance it is caused by reflection between the lens of the ophthalmoscope and the eye.

The last attribute noted is the limited FOV. The effective FOV is considerably lower for the captured video than for the examples in literature discussed in section 3.6. A limited FOV means fewer features visible at a time and increases the difficulty of registering the imagery considerably.

5.3 Preprocessing

5.3.1 Frame Selection

Image thresholding is a simple and commonly used technique in image processing. It is employed to segment an image and determine regions of interest. Thresholding is applied to a single channel of an image where its values are mapped to a binary image to yield a mask. Values above the threshold map to a 1 and values below the threshold to a 0.

Thresholding can be applied separately to all colour channels of an image and the results combined. This is referred to as multi-band thresholding. Consult Appendix A for an explanation of the colour thresholding process.

For the examination video, thresholding techniques can be used to segment frames into areas that possibly contain retina and areas that do not. Based on the size of the area returned, a distinction can be made on whether to keep or to reject a frame.

A simple observation of areas containing retina in a captured frame is that they are reddish in colour and are neither too bright nor to dark. Thresholding values should aim to isolate colours that fall within this range.

The YCrCb colour space is selected to specify the threshold values. Upon inspection, the CrCb plane (a slice of the YCrCb space) offers a continuous region that resembles the colour range of a typical ocular fundus. The HSV colour space also contains an appropriate colour range, however, the range is discontinuous and the YCrCb space thus takes preference.

A threshold range for the CrCb plane is suggested in figure 5.2. Here values in the Cr channel above 50% intensity and values in the Cb channel below 50% intensity are chosen. These threshold values are selected with slack to compensate for possible colour variations and avoid the over specification of parameters. The Luma channel's (Y) threshold values are also specified broadly with intensity values ranging between 10% and 50%, classified as valid regions. An upper limit to the Luma channel is imposed in an effort to remove glare.

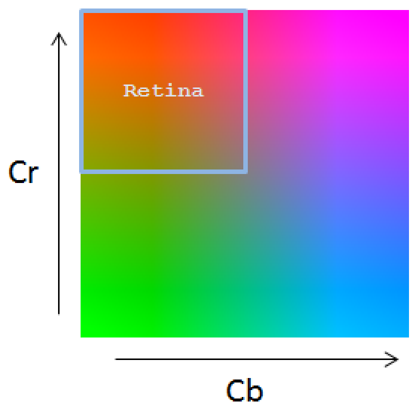


Figure 5.2: Threshold region on the CrCb plane to identify areas of retina.

The application of these thresholds yields three separate masks for each channel. These masks are combined to form a single mask with the use of an 'AND' operation. The resultant mask thus consists of the regions where all three separate masks overlap. In figure 5.3 a captured frame that contains retina and its resultant masks is shown.

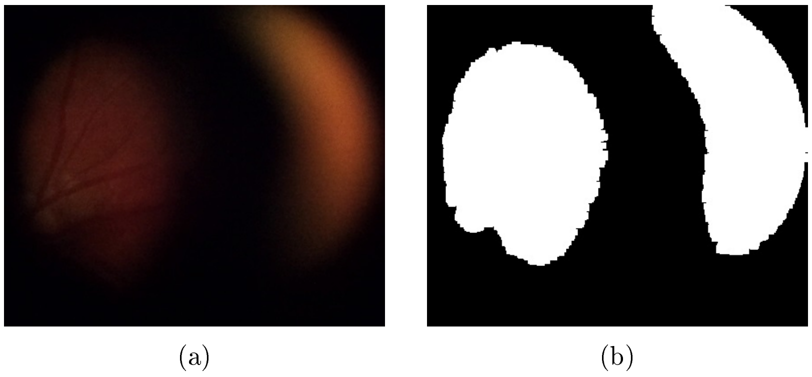
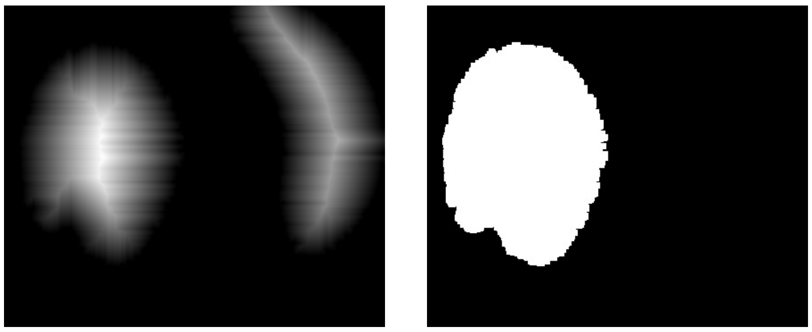


Figure 5.3: Thresholding of a frame containing retina and the resultant mask

Note that there are two discontinuous regions visible in the mask. Discontinuities are caused by an obstruction in front of the fundus. In this instance, the obstruction falls within the specified colour range and is also classified as fundus. It is caused by the side of the pupil (the iris) and the bright yellow region is the outer part of the eye (the sclera). A solution is to select only the best of the regions and to reject the others. The shape of an obstruction typically varies from the shape of usable section of retina. Note that the retina section has a rounded shape and that the unwanted structure is elongated and narrow. We can define a good section as one that covers a large, but compact area.

Compact shapes can be identified by applying the distance function. The distance function is applied pixel-wise and returns the distance of each non-zero pixel from a zero pixel. Figure 5.4a shows the results of applying this distance function to the mask in figure 5.3a. Note that the values are higher in the interior of the region. By searching for the maximum value in the results, a region is identified that is both large and compact. This region is selected as retinal candidate for this frame. Figure 5.4b shows the mask of figure 5.3a with only the desired section selected.



(a) Distance transform of a mask

(b) Section selected in mask based on peak in distance transform

Figure 5.4: Mask region selected based on the distance function.

With the possible area of fundus highlighted in a frame, a decision can be made on whether a frame will be processed further. The simplest approach is to specify a minimum size of the fundus to be present in a frame to classify the frame as usable.

The thresholding method is not the final decision on whether a frame is valid or not, but rather offers an efficient educated guess to reduce the bulk of unusable frames. If the examiner were to record any other red objects besides a valid fundus, the frames would also be selected as possible candidates. These false frames would be processed further. However, it will be found that these frames cannot be consolidated with other frames that do contain valid regions of fundus.

There will also be cases where the wrong section of a particular frame will be selected when the interference is reddish in colour. However, in these cases the viewable area fundus is typically small and of low quality.

For feature highlighting, a high-pass spatial filter is utilised. The motivation for not using more advanced techniques as outlined in section 3.6.1 is that the features in a single frame are often blurry and difficult to detect. More advanced techniques based on matched filtering would not be able to detect blood vessels in noisy data.

The high-pass filter serves two main functions: The first is the highlighting of blood vessels of a certain predefined size, and the second is the removal of the non-uniform illumination of the fundus.

The filtering is conducted in three steps on a grayscale image. First a Gaussian filter is applied to the image to determine the low frequency component of the image. The low-pass image is then subtracted from the original image to have only the high-pass components remaining. This is illustrated in equation 5.1.

$$I_{high\_pass} = I_{original} - I_{low\_pass} \tag{5.1}$$

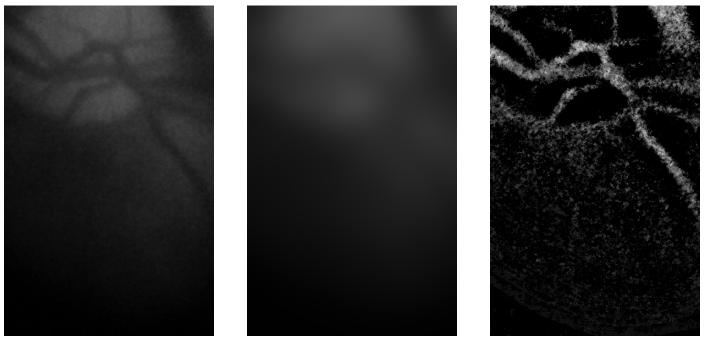
The remaining components are then inverted to have the blood vessels as bright features on a dark background. Equation 5.2 illustrates this inversion of a normalised image.

$$I_{high\_pass\_invert} = 1 - I_{high\_pass} \tag{5.2}$$

We combine equation 5.1 and 5.2 and ignore the constant component to yield equation 5.3

$$I_{high\_pass\_invert} = I_{low\_pass} - I_{original} \tag{5.3}$$

Figure 5.5 demonstrates the different steps in the highlighting.

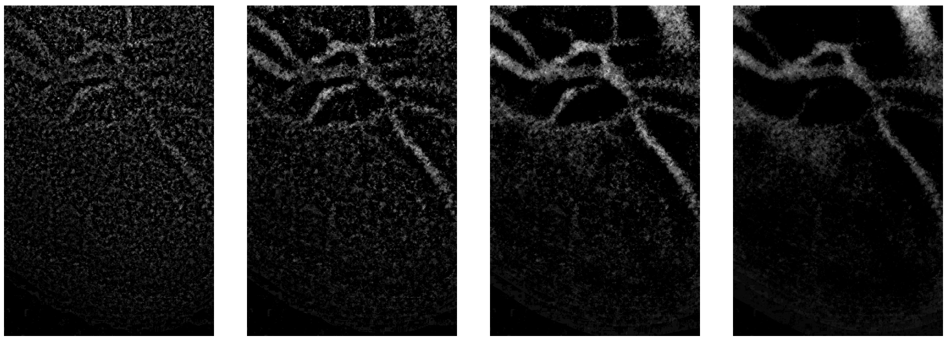


(a) I (b) I_{low_pass} (c) $I_{high\_pass\_invert}$

Figure 5.5: High-pass filtering of a retinal image.

The features are inverted to have the blood vessels highlighted instead of the shape created by the clipping region. If the clipping region were highlighted, the registration would fail as the clipping region would be registered instead of the continuously moving features.

To demonstrate the effect of the filtering for different frequencies, figure 5.6 shows the highlighted blood vessels when using different kernel sizes for the Gaussian smoothing. The size of the kernel (k) is expressed as a multiple of the estimated blood vessel diameter (d) in pixels. Note that the parameter is not extremely sensitive and that registration of the images can be achieved as long as the features are discernible and the constant pattern of the background illumination is eliminated. The robustness of cross-correlation registration will compensate for the vague or varying features that occur due to parameter choices.



(a) $k = 2 * d$ (b) $k = 4 * d$ (c) $k = 9 * d$ (d) $k = 18 * d$

Figure 5.6: Effect of kernel size on high-pass filtering

5.4 Consecutive Frame Registration

If given two consecutive frames in the video sequence, methods has to be devised to accurately determine the translation between their extracted data. This extracted data refers to the highlighted blood vessels.

Dense registration methods are typically used for medical applications; the reason being that the information contained in these images are of poor quality or limited resolution. The low quality of the images prevents the extraction of salient features typically required for sparse feature registration.

In this application the conditions are similar with captured retinal frames containing noisy data with low spectral resolution. Low spectral resolution is mainly caused by the low light exposure during the capturing process.

The method registering two dense feature sets, is typically determined by calculating the cross correlation between two images. In this application, the intensity maps of the highlighted blood vessels are cross correlated. The correlation can be calculated in two domains: the spatial- and the frequency domain. The spatial domain is selected due to the optimisations capable when calculating the cross correlation for a small region.

As mentioned, a key observation of the captured frames is that the translation between the blood vessels in consecutive frames approaches zero as the frame rate increases. This observation allows for great optimisation of the correlation calculations and is discussed with regards to the calculation of the spatial cross correlation in the following section.

5.4.1 Spatial Correlation

As described in section 3.4.1, spatial cross correlation can be expressed using equation 3.1. To utilise the benefit of a strong estimation of where the frames overlap, the correlation can be calculated for only a small region. The region for which the cross correlation is calculated is controlled by the amount of zero padding added to the reference image. Refer to section A.5 for the equations to determine the cross-correlation dimensions based on the dimensions of the reference- an sensed images.

As the reference image is zero padded, the cross correlation is calculated for a larger region and thus different translations between the images occur.

This is demonstrated with the correlation of two images of identical size. The images are windowed regions extracted from a larger image. The offset between the two images are exactly 10 pixels in the x-direction. In figure 5.7, the enlarged cross correlation for three cases are presented. In the first, the reference image is padded by 5 pixels on all sides. In the second, the reference image is padded with 10 pixels on all sides. Lastly the reference image was padded with 15 pixels on all sides. We note that the 5 and 10 pixel padding is not sufficient to identify the peak at the 10 pixel offset. Only when the padding of 15 pixels could the peak be located with certainty.

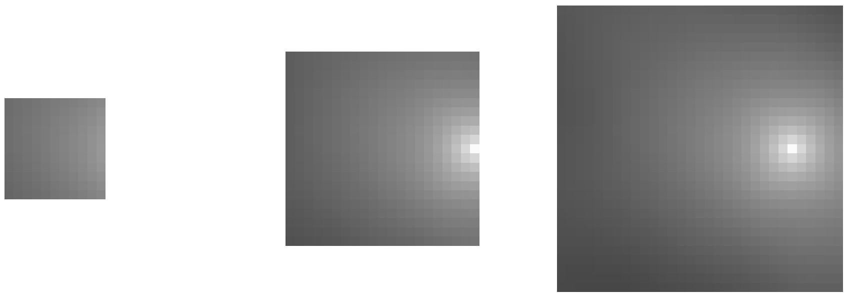

(a) 5 pixel padding (b) 10 pixel padding (c) 15 pixel padding

Figure 5.7: Sufficient padding to determine a 10 pixel offset between two images.

In figure 5.8 the full cross correlation contains the peak at the 10 pixel offset.

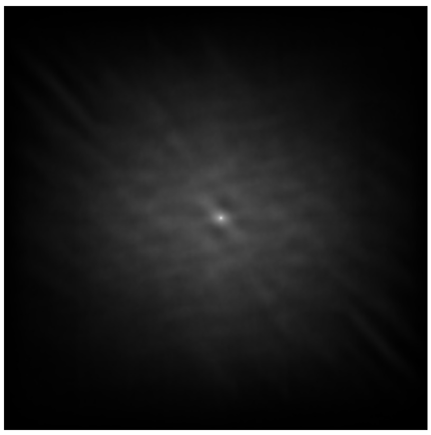
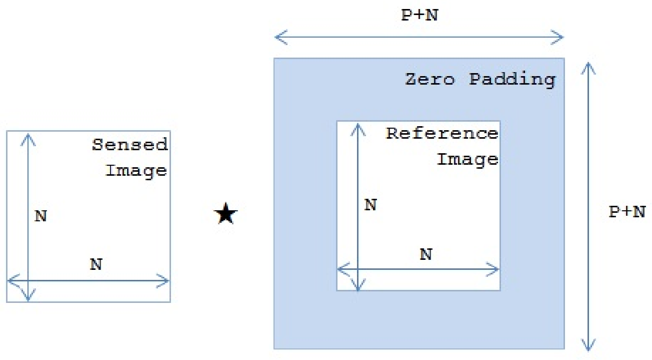
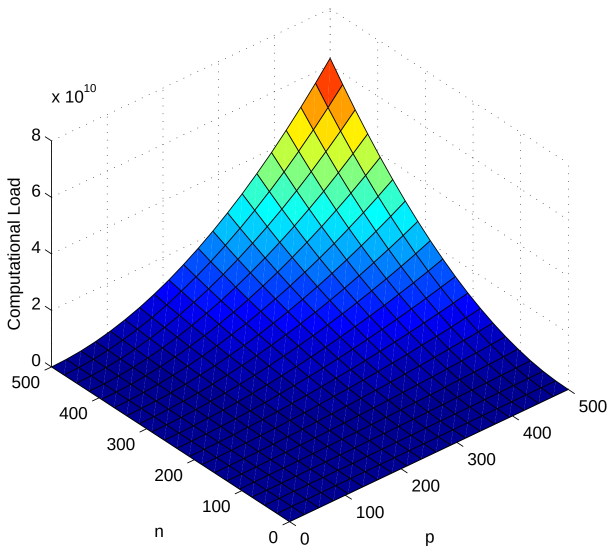


Figure 5.8: Full cross correlation map to enable location of peak.

Calculating the full cross correlation has the benefit of an increased possibility of locating the offset between two images, but comes with a penalty: as the size of the images increases so does the computational load. Figure 5.9 show the execution time of the cross-correlation of two $N \times N$ sized images in relation to the amount of zero padding (P) added to the reference image. Equation 5.4 describes the computational load as expressed by the number of individual pixel multiplications required to calculate the cross correlation. The derivation of the plot is given in Appendix I.



(a)



(b)

Figure 5.9: Cross-Correlation computational load relative to image padding (P) and image size (N).

$$multiplications = P^2 \times N^2 \tag{5.4}$$

We conclude that a low image size (N), low padding (P) or ideally both, should be maintained to avoid a drastic increase in the computational load of the calculation.

Two methods are suggested to achieve a reduction in computational load. The first is to keep the zero padding to a minimum and make a safe assumption that consecutive frames have a close to zero offset. The second optimisation is to crop away unnecessary content in the sensed images and only retain plausible areas of retina. Only these plausible areas of retina are cross-correlated opposed to the cross correlation of entire video frames. The masks obtained in section 5.3.1 would be used to identify these plausible areas and indicate which areas of a frame can be cropped away.

A problem with only cross-correlating a frame only with its immediate predecessor to determine its position, is the phenomenon of dead reckoning. The next section discusses this phenomenon and possible solutions.

5.4.2 Dead reckoning

With the small offset between consecutive frames comes the drawback of rounding errors. These occur because the offset is often in the order of a fraction of a pixel. By ignoring the sub-pixel offset, the error between two frames is hardly noticeable, but if the error accumulates over a larger set of frames, it would cause a drift between the real and estimated translation.

This is a phenomenon, referred to as dead reckoning, commonly associated with a navigational system where one's position is only based on an increment to a previous position and the process repeated for several iterations.

Dead reckoning can be avoided by determining a current position relative to a fixed reference, rather than from a previous position. The equivalent in our application is to estimate the position of a frame not only on its immediate predecessor's position, but also with regards to a single frame that serves as a fixed reference to all other frames in the cluster.

To demonstrate the concept, a synthesised set of frames were created where there is a 1.5 pixel x-offset between consecutive frames. The first frame in the set is considered to be the reference frame with a 0 pixel x-offset. The offset of all the other frames are described with regards to this reference frame.

The offset of each frame is calculated by using three methods. The first method simply uses dead reckoning to determine a frame's position based on its immediate predecessor. The second method correlates all the frames with the first frame in the set (the reference frame) to directly determine its offset. The last method correlates a frame with a running average of its predecessors translated to their estimated positions.

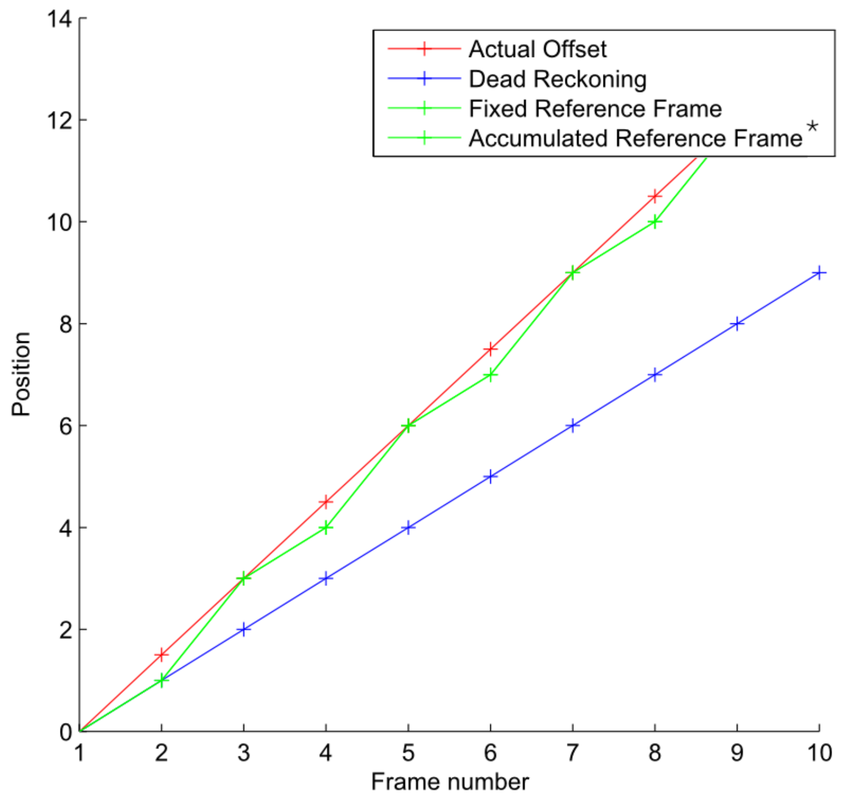


Figure 5.10: Comparison of frame to frame registration methods accuracy. ∗*The fixed- and accumulated reference frame approach offered identical results in this example.*

Figure 5.10 show the results of the estimation techniques. Note the large drift that is present when the only dead reckoning is used. At each stage of the correlation a rounding error occurs since the correlation calculates a whole-number offset. This rounding error accumulates and causes the drift between the estimate and actual position. Using the fixed reference solution offers an estimate closer to the actual offset, here the rounding error occurs, but does not accumulate.

The problem with the fixed reference solution is that, at a certain offset, the sensed image will not overlap with the reference image anymore. A solution is to select a new reference image once a certain offset has occurred. However, we propose a more continuous solution where each new frame is accumulated onto the reference image. Here the accumulated reference frame is calculated as described by equation 5.5.

$$reference_{new} = \alpha.reference_{old\_windowed} + \beta.frame_{new} \tag{5.5}$$

The windowed version of the reference frame refers to the section of the reference frame that is estimated to overlap with the new frame. Refer to section C.1 for a detailed explanation of windowed operations. The α and β constants indicate the speed at which the accumulation occurs with the sum of α and β being equal to one. In the example in figure 5.10, the accumulated reference-frame method and the fixed reference frame method returned the same offset.

The accumulated reference-frame method is favoured since features in a new frame is compared with an average of all other frames that overlap with the initial estimate of the new frame's position.

5.4.3 Sub pixel Registration

The cross-correlation based methods in the previous section offer a translation estimate in whole-pixel increments. These methods can be expanded to achieve registration with sub-pixel accuracy. By upsampling the sensed- and reference images, the estimate can be refined. Upsampling does not add additional information, but increases the resolution of the images and thus increases the spatial resolution of the cross correlation. The process entails upsampling the reference- and sensed images by a certain factor, estimating the offset between them and then downscaling the result by the same factor.

The difference here is that the input images are upscaled by a factor of two and yields a translation estimation resolution of 0.5 pixels. We note that in the example demonstrated in figure 5.10 the estimation obtained with this sub-pixel accuracy would follow the actual offset exactly. The resolution is sufficient to track the simulated 1.5 pixel offset between frames. This method yields a better estimated offset, but comes at the cost of increased calculation time.

5.5 Efficient Cluster Matching

Clusters are constructed from consecutive video frames with relative efficiency. However, a problem arises with the grouping of clusters since consecutive clusters would not necessarily cover the same region as expected with consecutive frames.

Clusters are often split up by interference while conducting the examination. This interference, be it blinking, eye movement or glare, would cause the physician to realign the device and would cause a discontinuity in the region being photographed. The clusters thus have to be sorted and aligned using a different approach and a method of rapidly locating matches in a large set of clusters is required.

In this section we discuss two possible approaches to optimise the matching of a large set of clusters. The aim is not to design a matching algorithm with high accuracy, but rather one that can group together a large number of clusters with great speed. The sorting algorithm only serves as a method to do the bulk of the matching rapidly and cross correlation and error checking is then conducted to join proposed matches.

The focus is on the development of an efficient pairwise matching of two clusters solution. This pairwise matching is then used iteratively to find matches in a larger set. The matching strategy utilising the pairwise matching is briefly outlined in the next section.

5.5.0.1 Matching Scheme

Given a pairwise-matching algorithm for two clusters, a method still has to be devised to apply it to a larger set. The approach outlined here aims to be simple in implementation and strategy.

Given a set of clusters the first step is to arrange the clusters from best to worst. The qualities that describe a good or bad cluster are subjective, and we simply sort them based on the amount of features they contain. A cluster with more content, i.e. a higher total intensity, is considered to contain more information in the desired frequency range originally filtered for.

Clusters with considerably lower content than that of the other clusters typically has too low content to be registered further and are discarded. The threshold of discarding clusters would be specified as a very low fraction of the total content in the best cluster. Discarded clusters typically contain content that vaguely correlates and managed to pass the first tier of thresholding, but does not contain visible blood vessels.

To do the pairwise matching, the first cluster is selected from the sorted list and is matched exhaustively with all the other clusters. The matching starts with the best clusters, those with the most content, and traverses to the worst, those with less content. This approach ensures that the best clusters are matched first and could allow for early termination of the matching if a sufficient match is found for a selected cluster.

Once a matching pair has been established the two clusters are removed from the set, their correlation verified; and combined to form a new cluster. The newly combined cluster is then added to the set and the process repeated. The matching process is considered to be complete once there are no good matches found in the remaining set of clusters. The remaining clusters are considered to be uncorrelated and impossible to combine.

In the next two sections, 5.5.1 and 5.5.2, we discuss possible pairwise matching methods and compare their performance in section 5.5.3.

5.5.1 Low resolution descriptor matching

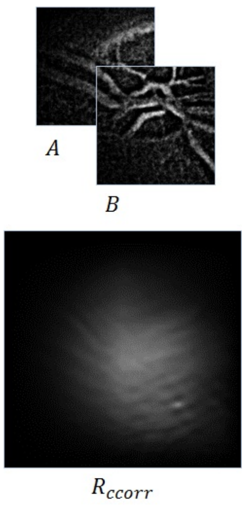
The first method we suggest for pairwise matching, is the use a downsampled representation of the cluster to serve as a low-resolution descriptor. When matching a pair of clusters their descriptors are then cross correlated to determine their similarity. The cross correlation peak is then used to gauge the similarity. A higher peak value indicates an overlap of more features.

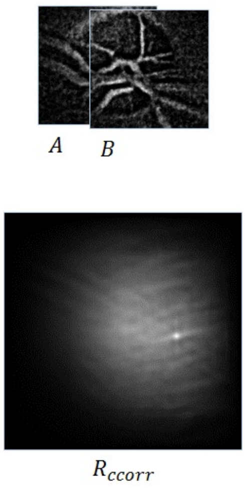
The cross correlation of the descriptors are conducted over a broader area than for the frames. A correlation over a broader area between two candidates has to be calculated and can cause a significant increase in the execution time. As demonstrated in Appendix I, the speed of calculating the cross correlation is proportional to the size of the images. By significantly reducing the size of the cluster images to form the descriptors, offers the benefit of increasing the speed of calculating the cross correlation.

A descriptor for each cluster is calculated once by smoothing and down-sampling the cluster significantly. This descriptor is stored and used for all matching operations. The matching entails calculating the full cross correlation for a pair of descriptors. For a smaller descriptor, the matching would be faster, but it has the possibility of reducing the accuracy too much. With a too low accuracy, slower speed would be induced since the matches would be rejected at the error-checking stage. A benefit of using these descriptors is that it offers an initial low-resolution estimate to where clusters overlap. This estimate can be used when performing the high-resolution correlation when combining two clusters.

Another optimisation, besides the reduction in size, is to specify a minimum required overlap between two descriptors. When conducting cross correlation, the strength of the peak is directly proportional to the size of the area where the images overlap. Checking clusters for a very small overlap would thus be redundant, because even a valid cross correlation that has a small overlap would be weak.

We demonstrate this concept with a simple example. Shown in figure 5.11 are two overlapping cluster pairs and their cross correlations. In both cases, the reference image is sufficiently zero padded. The two cross correlations are jointly normalised to highlight the intensity difference in the two peaks. The correlation for the 25% overlapping clusters is much weaker than for the 50% overlapping clusters, even though they both contain valid overlaps.





(a) 25 percent overlap (b) 50 percent overlap

Figure 5.11: Cross correlation of partially overlapping clusters.

5.5.2 Histogram of Oriented Gradients Matching

HoG Descriptor Histogram of oriented gradients is a method proposed by Dalal and Triggs [8]. They suggest it as a method to efficiently identify the presence of pedestrians within images. With this method, an edge detection is utilised to break an image up into a collection of gradients. The intensity and the orientations of the gradients are noted in a histogram. The histogram stores the total edge magnitudes over a range of orientations and is known as the Histogram of Oriented Gradients (HoG) descriptor. This descriptor is then classified using one of many mathematical classifiers to determine if a certain object is present in the image.

For the edge detection, a convolution kernel is used to identify edges and their orientation. A number of kernels were proposed and tested by Dalal and Triggs. Two pairs of these kernels are shown in equations 5.6 and 5.7

$$d_x = \begin{bmatrix} -1 & 0 & +1 \end{bmatrix} \tag{5.6a}$$

$$d_y = \begin{bmatrix} -1 \\ 0 \\ +1 \end{bmatrix} \tag{5.6b}$$

$$sobel\ d_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{5.7a}$$

$$sobel\ d_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{5.7b}$$

In both cases the kernels are used to calculate the derivative in both the x- and y-direction respectively. The kernels in equations 5.6a and 5.6b simply calculates the first order derivatives while the kernels in equation 5.7a and 5.7b calculates what is known as the Sobel derivative. The Sobel operator is a popular edge detection kernel often used in image processing.

To obtain the gradient components, the image is separately convolved with two kernels and the result is two intensity maps that represent the gradient in the x and y directions. The orientation and magnitude of the edges is calculated for each pixel's x-and y-gradient components. Equations 5.8 and 5.9 are used for the calculations, where G_x and G_y denotes the gradient components, θ, the orientation angle and G the total magnitude.

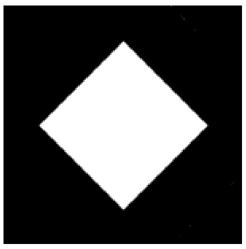
$$G = \sqrt{G_x^2 + G_y^2} \tag{5.8}$$

$$\theta = arctan(\frac{G_y}{G_x}) \tag{5.9}$$

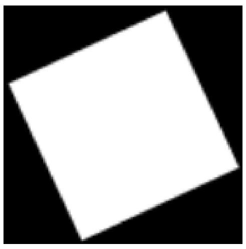
The magnitudes and orientations are used to construct a 1D histogram of magnitudes in different orientation bins. An appropriate bin size is chosen for the histogram and the magnitudes summed for the different orientations.

The bin size should be selected carefully to ensure enough data resolution is retained while not being too sensitive to noise. A bin that is too small would be sensitive to noise, while a bin that is too large would average out the information.

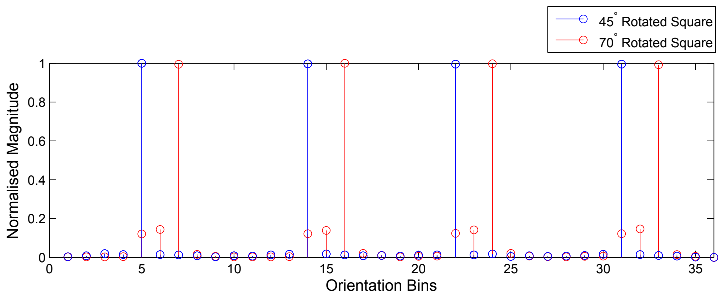
Figure 5.12c shows an example of the HoG descriptor constructed for a square rotated to 45° and 70° respectively. Note that this descriptor calculated for the shape, corresponds to our understanding of the shape's edges. The square has four large edges that are orientated in four directions. These four edges are clearly visible in the HoG descriptors, with the orientations of the edges corresponding with the peaks, at their respective angles, in the histogram. Any similar collection of edges would have a similar HoG descriptor.



(a) Square rotated to 45°



(b) Square rotated to 70°



(c) Descriptors for a Square Rotated to Different Angles

Figure 5.12: Rotated Square HoG Descriptors.

A number of methods are available to match histograms or determine the distance between them. In this setting two techniques prove beneficial. By interpreting two histograms as one-dimensional discrete signals and calculating the cross correlation between them, the matching can become rotation invariant. By normalising the histograms, a scale change in the shape would not affect the descriptors. Normalised HoG descriptors are used in figure 5.12c. We note that the scale change has no effect on the descriptor, while the rotation becomes a translation of the 1D signal. The translation can be estimated using 1D cross correlation as shown in equation 5.10

$$r_{ccorr} = \sum_{k=-\infty}^{\infty} h_1[k].h_2[x+k] \tag{5.10}$$

A feature of the HoG descriptor is that it is not affected by translation.

Modified HoG Descriptor The key benefit of the HoG descriptor is that it offers a short summary of the content of an image. This summary is considerably smaller than the original image and is thus much faster to compare to those of other images. In this section a modified HoG descriptor is proposed to suit our application.

The HoG descriptor is typically employed in situations where a decision has to be made whether an object is present or not present in a scene. A mathematical classifiers that operates in a N-dimentional space is typically used in such a situation. For this application, however, the descriptor is only used to find the descriptor most similar to it.

A proper comparator has to be selected to match descriptors. The assumption is made that there are no scale changes or any rotation present between clusters. The normalisation is thus not required and the peaks in similar descriptors, are not expected to shift. Not normalising the histogram, yields a more robust descriptor as it describes the absolute number of features in images and not a relatively scaled value.

This implies that a sufficient comparator could simply determine the squared distance between two histograms as described by equation 5.11. When comparing a histogram with a set of others, the best match is the one with the smallest error when calculating the square distance.

$$SquareDistance = \sum_{n=0}^{N}(h_1[n] - h_2[n])^2 \tag{5.11}$$

To extend the matching to include rotation between clusters, the cross correlation between two histograms can be calculated as described earlier in equation 5.10. The strength of the correlation peak would indicate how good the match is and the cluster with the highest peaks between their descriptors are the best matches.

The other modification made to the HoG descriptor is the method in which the gradients are detected. The typical process entails calculating the derivative in two directions and then determining the edges and their orientations from these derivatives.

For this application, the edges to be detected would be the edges of the blood vessels and the descriptor should mostly describe the layout of the blood vessels in an image. Given the resolution of the captured frames, a relative approximation to the size of the blood vessels can be made. With the blood vessel width determined, a cluster can be searched for blood vessels at different orientations rather than searching gradients at any resolution.

Searching for blood vessels is conducted by convolving the image with a simplified template that represents the structure of a blood vessel. An inverted section of blood vessel typically consists of a bright ridge with dark areas around it. This can be approximated with a white rectangle on a dark background. A number of the proposed convolution kernels, generated for eight different orientations, are shown in figure 5.13. These kernels operate in a similar manner to the Gabor filter discussed in section 3.6.1. In this application, the simplified kernels prove sufficient since the filter is not used to enhance the image quality, but simply to extract certain metrics.

Note that the kernels only range between 0° to 180° instead of the typical 360° used for the HoG descriptor. The reason for this is that a short section of a blood vessel is not discernible from its 180° rotated counterpart.

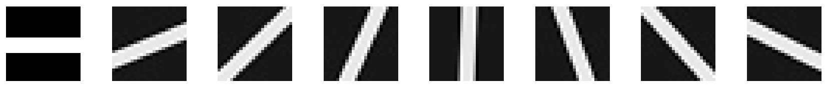

(a) 0° (b) 22° (c) 45° (d) 67° (e) 90° (f) 112° (g) 135° (h) 157°

Figure 5.13: Blood vessel detection kernels generated for eight orientations.

Another important feature of the kernels is that the dark areas represent a value of -1 instead of 0 and the white areas a value of 1 as shown in figure 5.14. This allows for the suppression of noise in the convolution process and prevents false highlighting of blood vessels in a area only containing interference, noise or other structures.

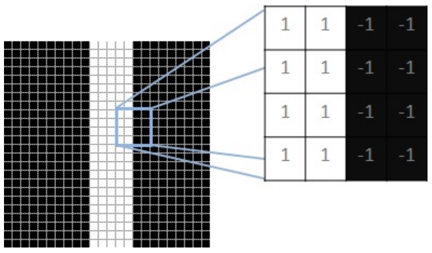
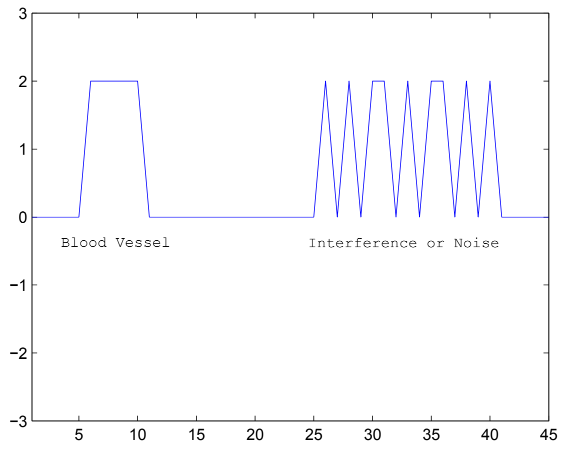
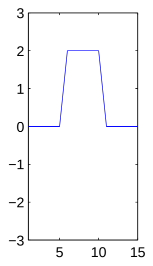


Figure 5.14: Enlarged view of detection kernel.

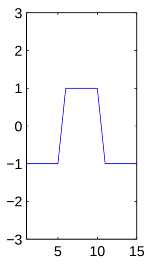
Figure 5.15 demonstrates the concept with a 1D analogy. A simulated input signal in figure 5.15a represents a 1D slice of a 2D image. It contains two significant regions; on the left is a possible blood vessel slice and on the right some interference. In figure 5.15b and 5.15c, two possible kernels are shown. The kernel in figure 5.15b is a mean shifted version of the kernel in figure 5.15c. Here the 'off' sections of the kernel is represented by -1 and not 0. To test the performance of these kernels, they are convolved with the input signal. With the results we note that the convolution using kernel, suppresses the interference, while the kernel in figure 5.15c also generates a peak for the interference. It is thus beneficial to use Kernel B and to reject the convolution results below 0.



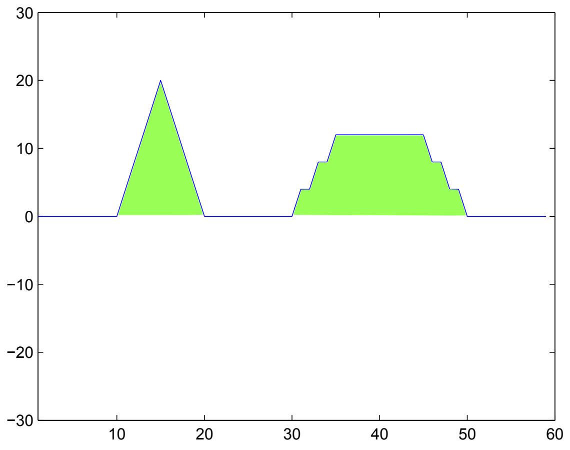
(a) Simulated 1D signal of 2D blood vessel image slice



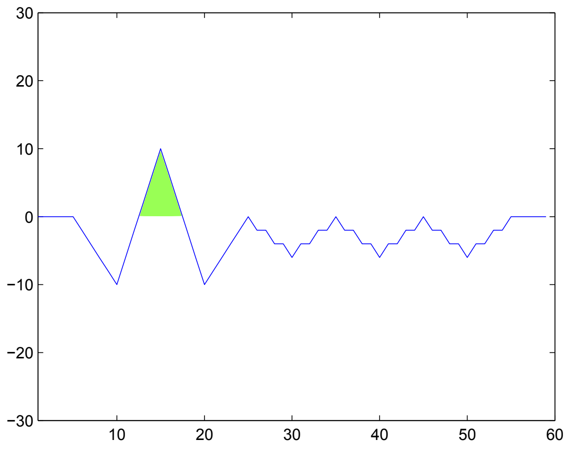
(b) Kernel A



(c) Kernel B



(d) Signal convolved with Kernel A



(e) Signal convolved with Kernel B

Figure 5.15: 1D analogy of proposed blood vessel detection kernels.

Convolving with each of the kernels in figure 5.13 yields a score for the total amount of blood vessels with a specific orientation. The total score for each kernel convolution is determined by the summing the convolution results. These scores is used to construct the histogram, where the score for each kernel is the value for a specific bin in the histogram.

Shown below, in figure 5.16, is a typical cluster convolved with kernels at different orientations. Bright areas in the result indicate regions where the blood vessel has the same orientation as the kernel. Also note that the noise visible in the cluster is not amplified by the convolution.

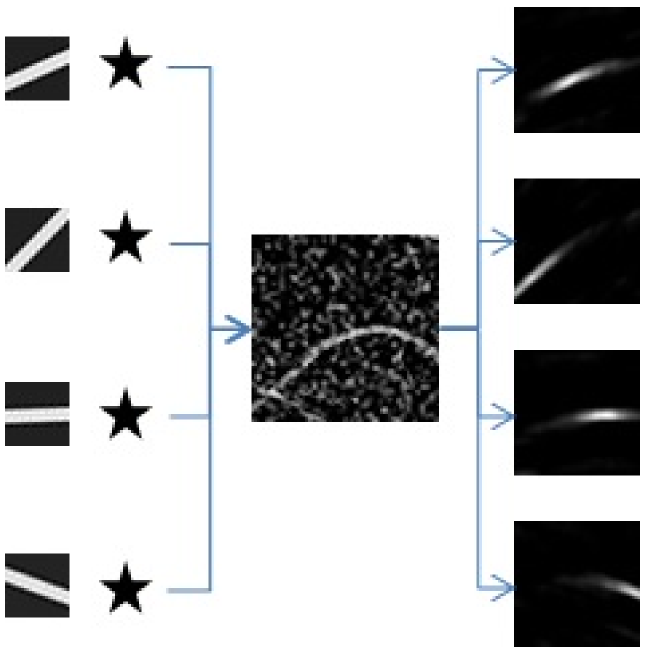


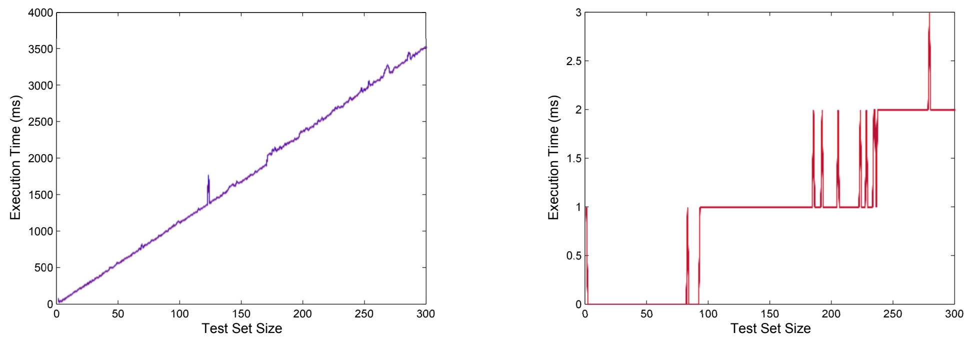
Figure 5.16: Convolution with detection kernels at different orientations.

5.5.3 Comparison

To test the performance of these algorithms, a scenario is set up where artificial clusters are used to measure the speed and accuracy of the algorithms respectively. Clusters are created by selecting a random window from the reference image and adding temporal noise to it. The temporal noise is scaled in size such that its spatial frequency is the same as that of the blood vessels. A large set of clusters is created in this manner and used as a test data set.

It is ensured that each cluster in the set has a match that covers exactly the same region. Note that these two corresponding clusters would cover the same region, but would differ slightly in appearance because of the temporal noise added. Several conditions are selected for the tests. The first is scaling the cluster images down to 20% of their original size. This scaled image is used as the low resolution descriptor and the HoG descriptor is also calculated from this smaller image. Further, the minimum overlap in the cross-correlation matching is set to 25%. The cross-correlation method would thus not check for overlaps smaller than 25% when matching the clusters. Testing the algorithms entails exhaustively searching the dataset for the best match. The best match would ideally be the cluster that covers exactly the same region. Since the clusters are all created artificially, their coordinates are known and the accuracy of a match can be determined. The accuracy is specified as the percentage by which two matched clusters overlap.

A test is first conducted to measure the speed of the two algorithms. In this scenario the size of the test set is incrementally increased and the time required to locate the best match for 10 different clusters is noted. Since the pairwise matching typically executes with a constant time, the total searching time is expected to increase linearly as the number of clusters that are to be searched, increases. The plots below show the execution time for the two methods. It is clearly visible that the execution of the cross-correlation method is considerably slower than that of the HoG descriptor matching.

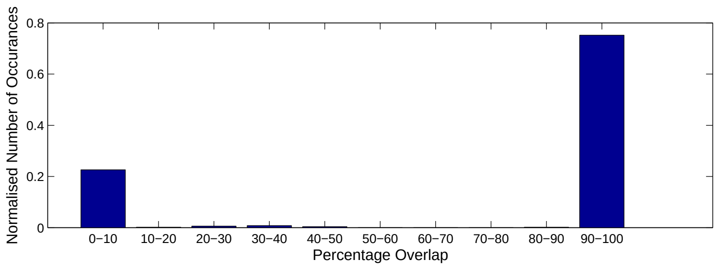


(a) Cross-Correlation Descriptor Matching (b) HoG Descriptor Matching

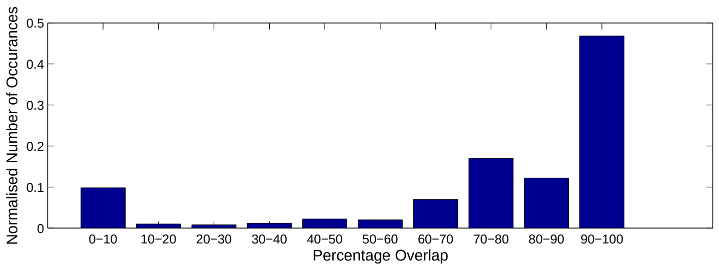
Figure 5.17: Execution times for finding matches in a large set of clusters.

Note that the linear increasing trend is visible for this with both plots. However, the execution time for the HoG descriptor matching is so low that this trend can be ignored. The typical number of clusters gathered from a single video is not expected to be higher than 300 as used in the test case and for the HoG descriptor, finding the best match in a set of 300 is executed ten times in under 3ms.

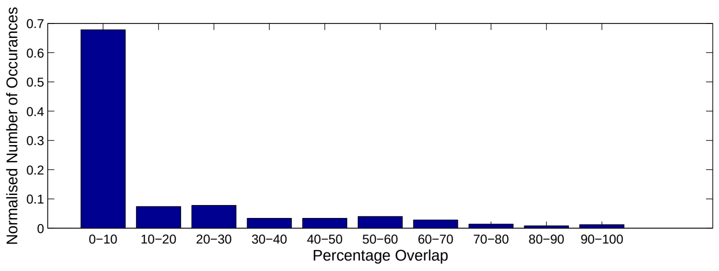
The second test is to measure the accuracy of the algorithms. Matching is conducted between a cluster and a 300 random-clusters test set. The matching is conducted 500 times and the normalised results noted in the respective histograms in figure 5.18a. Figure 5.18b shows the accuracy of the matching when using cross correlation of a low-resolution descriptor and figure 5.18b shows the accuracy of the matching when using the HoG descriptors. Lastly, the plot in figure 5.18c serves as a baseline and indicates the accuracy when a random cluster is selected form the test set as a best match. As expected, the accuracy in the last distribution is extremely low.



(a) Cross-Correlation Descriptor Matching



(b) HoG Descriptor Matching



(c) Random Matching

Figure 5.18: Accuracy of respective matching algorithms.

Note that the cross-correlation method is better at finding exact overlaps in the presence of the noise than the HoG descriptors. Regardless of this, the accuracy of the HoG descriptor is satisfactory, since a 70% overlap between clusters is still sufficient for the accurate combining of clusters. If we impose threshold of a 25% overlap to indicate a successful match between two clusters, the accuracies for the respective methods will be as shown in table 5.1.

Method	Accuracy
Cross-Correlation Descriptor Matching	76%
HoG Descriptor Matching	88%
Random Matching	20%

Table 5.1: Matching Algorithm Accuracy.

From these simulated test cases, it is deduced that the modified HoG Descriptor offers a robust and efficient method of matching a large number of clusters with each other. The overhead of calculating the HoG descriptors is overshadowed by the extensive gain in matching time.

We also note that the execution time for the HoG Descriptor matching is so significantly low that the inefficiencies of an exhaustive search is not of concern with a limited set.

The last apparent benefit of the HoG descriptor is that it requires considerably less memory for storage than the low-resolution descriptors. This feature makes it possible to store images in secondary memory and only retain their descriptors in main memory. The images would then only be retrieved if their HoG descriptor indicates that they will be a good match.

5.6 Cross-Correlation Verification

At various steps of calculating the transformation between the retinal images, a method of validation has to be established. In this section we discuss the analysis of the cross-correlation result to determine the accuracy of an estimation.

The validation at two distinct steps are discussed here. The first is monitoring the correlation peak value when grouping frames into clusters and the second is the validation of the cross correlation when two random clusters are matched.

5.6.1 Frame Grouping

When calculating the cross correlation of two images, the value of the peak suggests the number of features that correspond at a certain offset. The more features overlap, the higher the peak value.

Monitoring the peak value is a method we suggest for initial grouping of consecutive frames into clusters. This peak value is utilised to gauge the number of mutual features in consecutive frames.

When monitoring this peak value for consecutive cross correlations, a sudden decrease in the sequential peak values could indicate a loss of tracking. Loss of tracking would indicate that the features could not be correctly matched between frames. A sudden increase in the peak intensity would indicate the start of tracking between frames. Small fluctuations in the peak value are ignored as this could be caused by certain additional features moving in and out of the viewing field as well as noise present in the frames. Consecutive frames with similar cross correlation-peak values are then grouped together into clusters.

Figure 5.19 demonstrates a hypothetical plot of the cross correlation-peak value for a sequence of frames and how they will be grouped into clusters

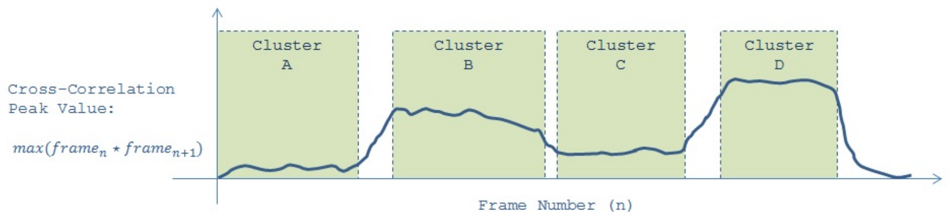


Figure 5.19: Peak values of the cross correlation of a sequence of frames utilised to group the frames into clusters.

Here it is possible that cluster B and C do contain mutual information, but C simply captures a region with fewer features. In this scenario, the clusters will be combined during the cluster matching.

5.6.2 Cluster Match Verification

When calculating the cross correlation between random clusters, the situation is slightly different than for that of the sequential frames. Sequential frames will always contain some mutual information. Clusters that are to be matched could possibly contain no mutual information if the descriptor matching was erroneous.

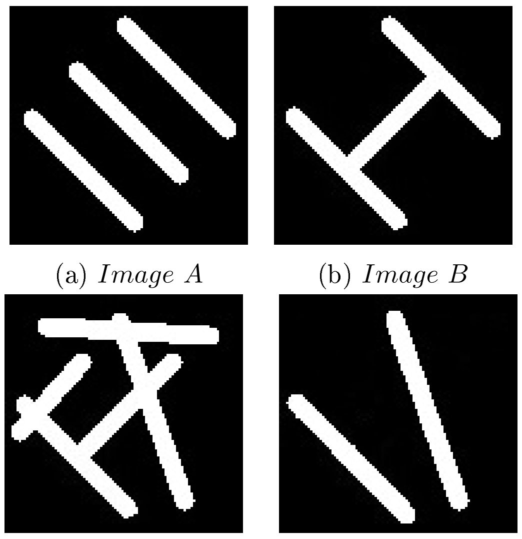
A method typically used to gauge the strength of the peak is to use a normalised cross correlation. With normalised cross correlation, the resulting map is divided by the theoretical maximum correlation value for each pixel. In this application the normalised cross correlation would indicate the number of overlapping blood vessels divided by the product of the total number of blood vessels in the two clusters. The normalised cross correlation has the benefit of a peak for a small valid overlap having a similar peak for a valid larger overlap.

In this section we discuss the classification of the peaks as being valid or invalid by calculating the normalised cross correlation as well as by characterising the peak's shape and uniqueness.

To characterise the peaks, some of the typical observations are demonstrated in the following section. Section 5.6.4 elaborates on the concept of normalised correlation versus regular correlation. Thereafter, section 5.6.5 discusses a method of classification based on the observations.

5.6.3 Peak Characterisation

The images in figure 5.20 shows a selection of images that contain elongated features similar to blood vessels. The cross- and auto correlation of these images will be used to illustrate certain characteristics.


(a) *Image A* (b) *Image B*

(c) *Image C* (d) *Image D*

Figure 5.20: Sample images.

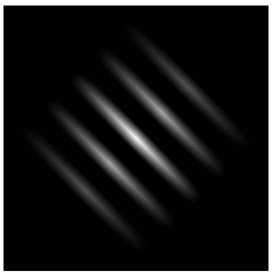


Figure 5.21: Auto correlation of *Image A*.

The first correlation is the auto correlation of *Image A* shown in figure 5.21. This would represent a perfect match between two images, however, note that in the correlation, the peak is elongated. This elongated peak could easily cause misregistration since a ridge of high-intensity values are present and the absolute peak, difficult to locate. With the presence of noise this task would prove even more difficult.

The reason for this appearance, lies in the nature of the features being correlated. The features are all orientated in one direction and not spread out over a range. In the situation where the ends of the line features are caused by clipping of the viewing field, the actual offset is difficult to extract. The true offset can by represented by any of the peak values on the ridge.

Figure 5.22 shows the auto correlation of *Image B*. Here the features are orientated perpendicular to each other. The correlation between the two identical images yields an easily locatable peak. Since the features are perpendicular, the exact offset between the images can be determined and there is no drift in any direction.

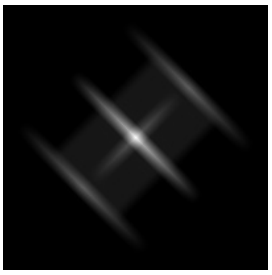


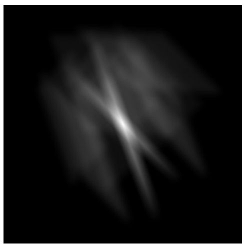
Figure 5.22: Auto-correlation of *Image B*.

In the last three examples the auto correlation of *Image C* and *Image D* and the cross correlation between them are noted respectively in figures 5.23a, 5.23c and 5.23b. The auto correlation for *Image C* yields an easily identifiable round peak. The roundness of the peak is attributed to the large range in feature orientations. For a mathematical discussion on the roundness and size of the peak as a result of the orientations present in the features, consult Appendix H.

The cross correlation for *Image C* and *Image D* indicate a strong peak where the features overlap. This causes a problem since the occurrence of a peak does not necessarily indicate an actual overlap of the features. The images do contain overlapping features, but are not a match.

For the auto correlation of *Image D* the peak is elongated and note the similarity with the cross-correlation of *Image C* and *Image D*. The reason for the similarity is that the features in *Image D* are wholly contained within *Image C*. The reason for the auto correlation of *Image C* differing from the cross correlation, is that *Image C* contains additional features not present in *Image D*.

(a) Auto correlation of *Image B*


(b) Cross correlation of *Image C* and *Image D*

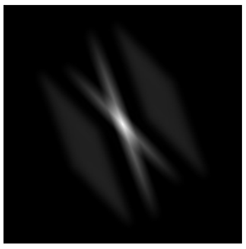

(c) Auto correlation of *Image D*

Figure 5.23: Cross correlations.

5.6.4 Normalised Cross Correlation

Normalised cross correlation is calculated in a manner manner to regular correlation, except that the peak values are normalised using a normalisation map. The normalisation map contains the theoretical maximum value for each value in the cross correlation. By normalising the correlation map, the peak value has a maximum of 1. With 1 indicating a perfect overlap of data. The equation for normalised cross correlation is given by equation 5.12. Here T is the template or sensed image, I the reference image and $Z(x, y)$ the normalisation map.

$$R_{ccorr\_normed}(x,y) = \frac{R_{ccorr}(x,y)}{Z(x,y)} = \frac{\Sigma_{x'y'}[T(x',y').I(x+x',y+y')]^2}{\sqrt{\Sigma_{x',y'}T(x',y')^2.\Sigma_{x',y'}I(x+x',y+y')^2}} \tag{5.12}$$

Figure 5.24 demonstrates the cross correlation and the normalised cross correlation of two images with the normalisation map is shown in figure 5.24.

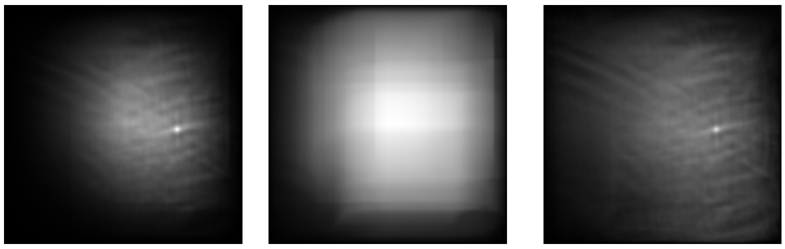

(a) Correlation (b) Normalisation map (c) Normalised

Figure 5.24: Normalisation of cross correlation.

The normalised versions of the cross correlations in figure 5.23 is noted below in figure 5.25. The normalised peak values are indicated in the figures. Note that figure 5.25a and 5.25c represent valid overlaps and that figure 5.25b represents an invalid overlap. The normalised peak value in figure 5.25b is lower since the peak only represents an overlap of a subsection of the data and is normalised with regards to all the data at the particular offset.

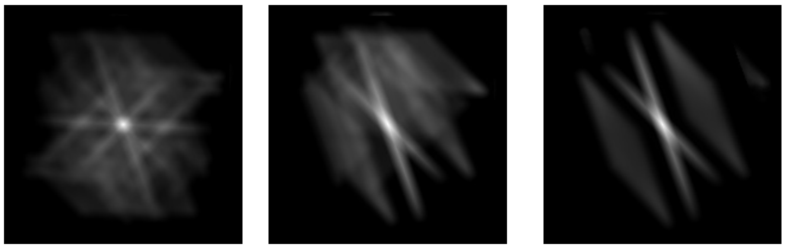

(a) Peak = 1 (b) Peak = 0.69 (c) Peak = 1

Figure 5.25: Normalised cross correlations.

5.6.5 Proposed Correlation Verification Method

A method for determining if the cross correlation results indicates a valid overlap based on the peak classification and the normalised cross-correlation peak value is proposed.

When a candidate peak is identified, the overlapping cluster sections are isolated and recorrelated. The peak is thus recalculated for only the overlapping sections and normalised by the features contained within in these sections. This normalised peak value is used to assign a score to the match. Shown below in figure 5.26 are the scores achieved for peak representing valid overlaps for artificially created clusters.

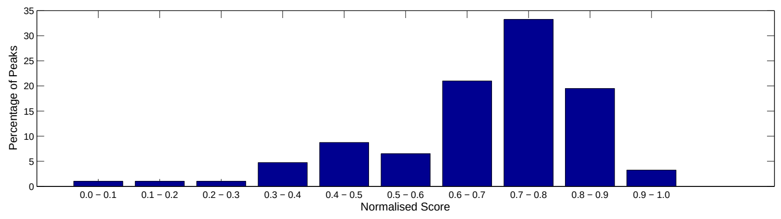


Figure 5.26: Scores obtained for valid peaks.

Figure 5.27 shows scores obtained for peaks representing invalid overlaps.

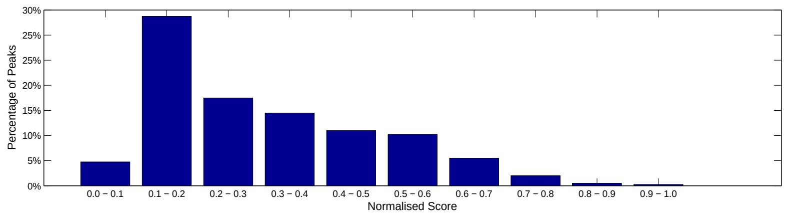


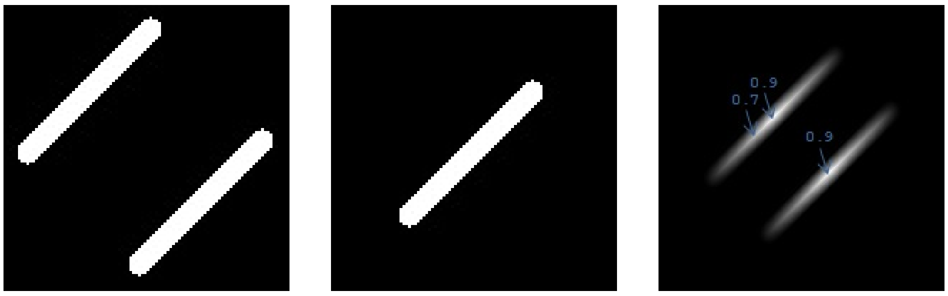
Figure 5.27: Scores obtained for invalid peaks.

The distribution in figure 5.26 has a easily identifiable mean with a small deviation. This score thus serves as a sufficient scoring system when combining clusters.

There are, however, cases that the scoring system would not account for. The first is the case of multiple or elongated peaks. The second problem is that a normalised peak for a small overlap is often erroneous since only a small section of data is compared between the images. The resulting score can thus be artificially high since the value is normalised for a small region.

A simple method to check for peak uniqueness and shape is proposed. When the cross-correlation map is inspected, the peak and an area around it is zeroed for a specific radius. The radius would typically be a value close to the blood vessels' diameter. The remaining values in the correlation map are then searched for any values that are similar to the candidate peak. If any values with a value similar to that of the peak is found, it will indicate an uncertainty in the identified peak.

Figure 5.28 demonstrates two cases with three normalised intensities indicated in a cross-correlation map. Here there are two peaks with equal values as well as a slow falloff in one direction for both of the peaks. This case would be rejected since no safe assumption on the true correlation peak can be established.



(a) Image A (b) Image B (c) Cross-correlation

Figure 5.28: Uncertain peaks of normalised cross correlation.

The second problem of peak verification is addressed by specifying the minimum overlap that a valid peak is allowed to represent. We use the test case illustrated in figure 5.26 and specify a score of 0.7 or higher to indicate a valid peak. Figure 5.29 shows the accuracy of the peak classification versus the overlap they represent. Note that peaks that represent a small overlap, are generally erroneous. In this scenario we would disregard peaks that represent an overlap of less than 30%.

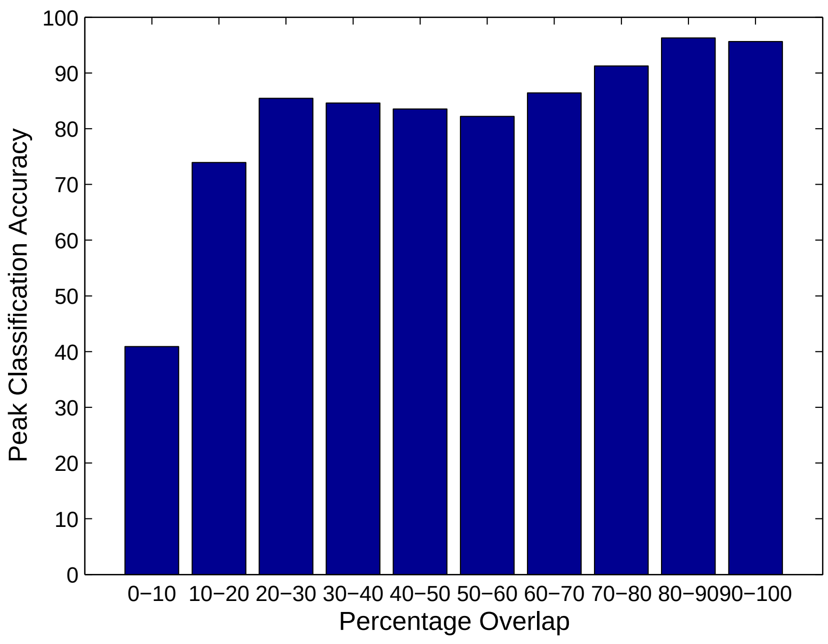


Figure 5.29: Peak classification versus the overlap that they represent.

Chapter 6

Results

6.1 Introduction

The proposed system was implemented in the form of an Android application for a mobile device and a C# .NET application running on a PC. A wrapper of the OpenCV library was used to perform the image processing. A discussion on this library is provided in Appendix B and a discussion on the code implementation in Appendix C. The typical use of these implemented systems are illustrated in Appendix F.

Besides the PC implementation, a proof of concept for processing the videos directly on the device was conducted. This is briefly described and the results shown in Appendix E.

This chapter discusses the performance of the system in terms of accuracy. Accuracy is difficult to quantify and the appearance of the results are compared with images obtained with another device.

6.2 Image Quality

To demonstrate the quality of the results obtained, they are compared with two control cases. The first comparison is made with a single frame for the video sequence and the second comparison is made with regards to images obtained with another commercial fundus camera, the Optomed SmartScope. The Smartscope is a digital fundus camera with a considerably larger FOV than the PanOptic Ophthalmoscope. Additional Information on the optomed smartscope is available here [31].

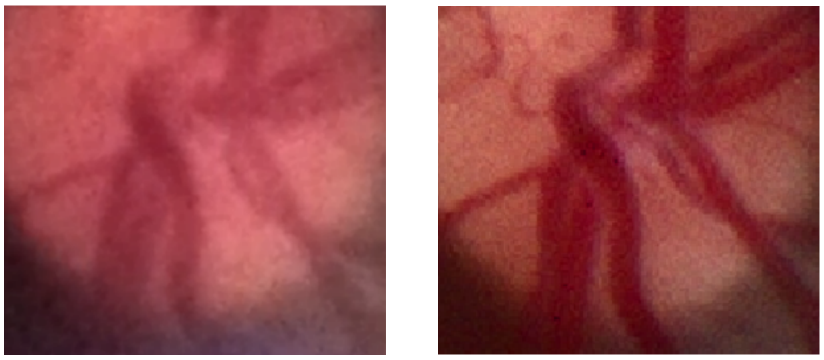
Two attributes of the results, with regards to its appearance, are discussed. The first is the spatial resolution of the obtained images and the second is the effective FOV obtained. Note that these features have a subjective element to them and are discussed in a partially subjective manner. The increased quality of the processed images are also case specific with the result being dependant on the amount of data accumulated. A large number of frames for a specific region offers an increased spatial resolution and a large sweep in the video sequence yields a larger FOV.

6.2.1 Increased resolution

The unprocessed images captured typically have a high SNR. This high SNR is due to the low light exposure. The camera on the mobile device increases the gain to compensate for the low amount of light and the result is a grainy image with the visibility of some features being compromised.

When the registration is conducted, a number of frames covering the same content is aligned and averaged. In signal processing the effect of combining multiple instances of the same signal is an increase in the SNR. It is thus expected that the average of several overlapping video frames will yield a clearer signal, with the signal in this case being the structure of the blood vessels.

Shown in figure 6.1, are two images of a patient's optic disc. Figure 6.1a is a single video frame from a captured video sequence using the proposed setup. Figure 6.1b is an image of the same optic disk taken with the Optomed Smartscope. Note that the image in figure 6.1a is considerably clearer with some features visible that are not visible in figure 6.1b. The image in figure 6.1b has a higher quality and can serve as a control to compare manipulated data against.



(a) Optic disk photographed with use of Pan Optic ophthalmoscope (b) Optic disk photographed with use of Smartscope fundus camera

Figure 6.1: Comparison of captured imagery.

The results of the processed video frames are shown figure 6.2. The composite consists of a number of images similar to the one in figure 6.1a that had its features highlighted and then averaged.

Note that there are several features clearly visible in the composite image that are not visible in the individual frame. Even though the frame in figure 6.1a and the processed image in figure 6.2 have the same image resolution their spatial resolution differs. The images have the same size (in terms of pixels), but the spatial resolution of the composite image is said to be higher in the composite since finer features are clearly observable. These finer features are checked against the control image to verify that they are not simply artefacts of the processing.

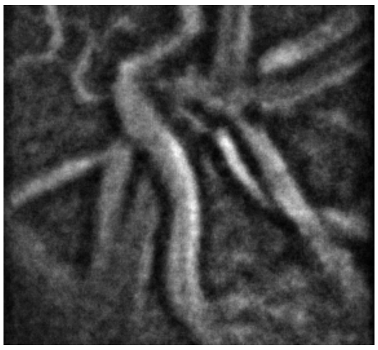


Figure 6.2: Composite image constructed from a number of video frames.

The mean horizontal diameter of a human optic disk is known to be 1.88mm with little variation. This measurement allows the addition of a scale to the images. The scale in turn allows the accurate measurement of the detail visible in a particular image that can be compared between any ocular fundus image.

Figure 6.3 shows the images from figures 6.1a, 6.1b and 6.2 on the same scale. The images of figures 6.1a, 6.1b have had simple preprocessing techniques applied to them to highlight their blood vessels. A highpass filter has been applied to remove the varying background and the contrasts adjusted with intensity mapping.

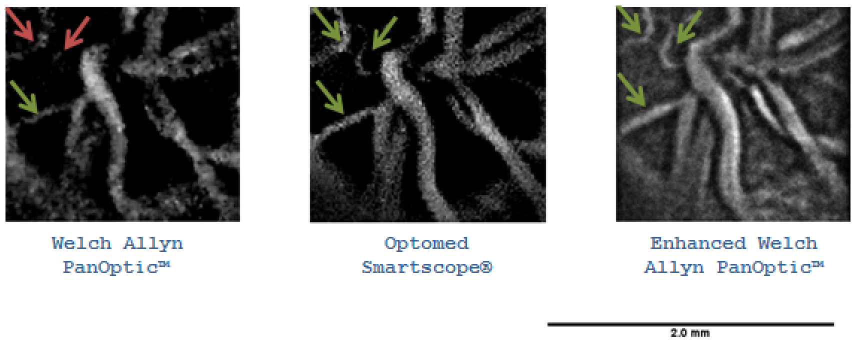


Figure 6.3: A scale comparison of optic disk images.

Note that the diameter of the vessels for the unprocessed Pan-Optic image appear slightly smaller than for the other two images. This is due to low contrast between the vessel and its background. This indicates that the edges of the vessel are at the cusp of being observable.

For an objective assessment of the composite image, a criteria for a vessel being observable, is established. We classify a section of vessel to be observable if its intensity is continuous for a length longer than the diameter of the vessel.

Using this criteria, a few vessels in figure 6.3 are classified. The red arrows indicate vessels that are not detectable versus green arrows indicating vessels that are. It is noted that the composite image has a spatial resolution closer to that of the Smartscope than to the unprocessed frame.

Using the scale to conduct measurements, the size of the smallest detectable vessels can be made. For the unenhanced images captured with the Pan-Optic, the minimum blood vessel diameter is 0.07mm while in the Smartscope and enhanced images, blood vessels with a diameter in the range of 0.045mm are detected. These measurements can serve as a comparable parameter between other fundus images.

As mentioned, the SNR of the composite image is expected to increase as the number of frames used to construct the image increases. Results are thus better for areas that are recorded for a longer duration. Figure 6.4 demonstrates this. Here the clusters are constructed by aggregating very high frequency content from the frames covering the same region. Note that as the number of frames increase, so does the image quality.

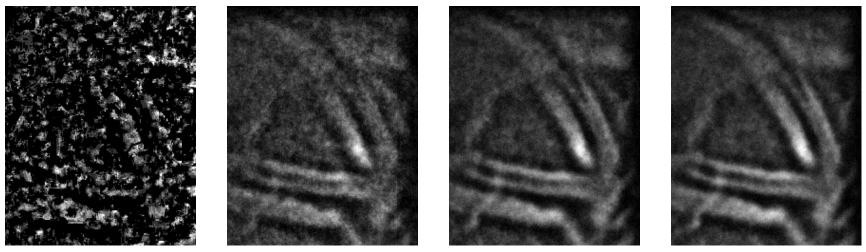

(a) 1 Frame (b) 40 Frames (c) 80 Frames (d) 120 Frames

Figure 6.4: Clusters consisting of an increasing number of frames.

6.2.2 Increased Field of View

The increase of the FOV is based on the area covered in the video sequence. If a larger region is captured, a larger effective FOV is achieved. A specific case an the achieved FOV is demonstrated. The FOVs are compared since this is an attribute typically specified for ophthalmoscopes and fundus cameras and is to be used as a benchmark here.

A method for determining the effective FOV is established to enable an objective measurement of the image. Figure 6.5 suggests a method of deriving the FOV given the measurements of the observed image and the known diameter of the human eye.

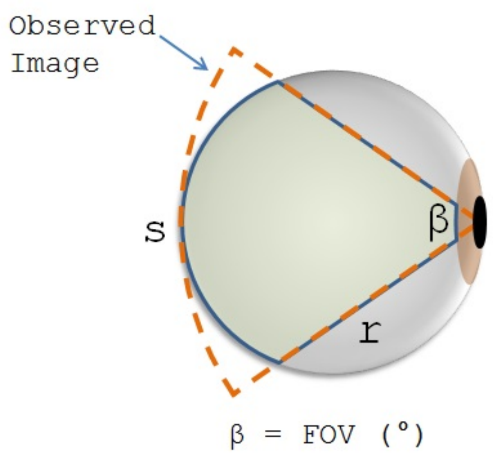


Figure 6.5: Calculation of FOV form a fundus image

The size of the image can be determined by using the size of the optic disk as reference. The FOV can be estimated by calculating the angle (β) in equation 6.1. Here the radius of the arc (r) is equal to the diameter of the human eye (25mm) and the arc length (s), equal to the diameter of the observed image.

$$\beta = \frac{s}{r}.\frac{180}{\pi} \tag{6.1}$$

An image from the Smartscope is used as a reference with a range of effective FOVs calculated using the described method. The results are indicated in figure 6.6.

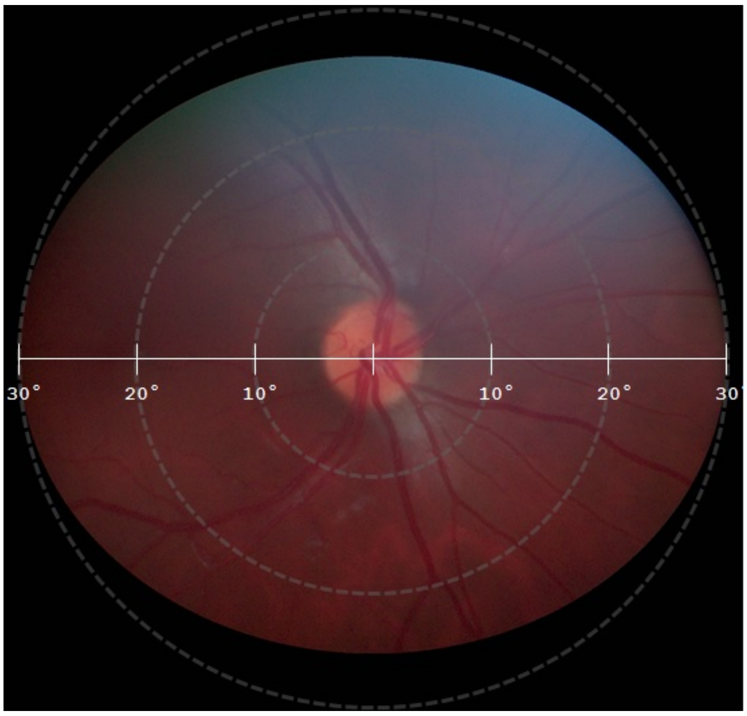
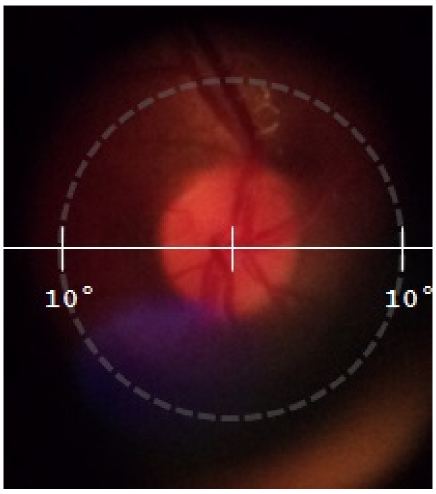
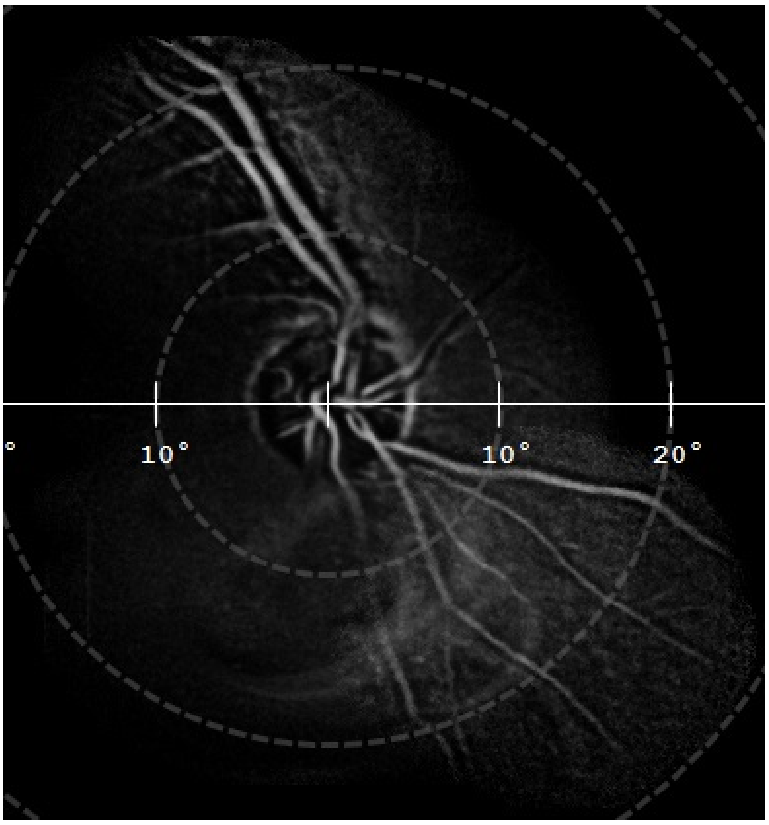


Figure 6.6: FOVs indicated on a Smartscope fundus image.

Figure 6.7 shows an individual frame and a composite cluster with their FOVs indicated. In this instance the FOV of the composite image does not match that of the Smartscope. However, it is still considerably larger than that of an individual frame.



(a) FOV of a single video frame



(b) FOV of a composite image

Figure 6.7: Illustration of the increased FOV obtained by combining a large amount of frames.

Chapter 7

Conclusion

Chapter 2 discussed the anatomy of the eye, the structures observed during a fundus examination, the retinal disease that is typically monitored and how the examination is conducted. An understanding of the examination process is imperative as this dictates the nature of the data to be captured and, in effect, the design of the registration algorithms.

The concepts of image registration were introduced in chapter 3. Here the different approaches to conducting image registration were discussed and emphasis put on the large range of applications as well as the extensive taylorization of methods for specific task. The different approaches used for registration of retinal images were also discussed. Again, it was noted that even for the very specific application, different approaches are developed based on capturing conditions. Chapter 3 also discussed typical approaches to isolating the blood vessels in fundus images and it noted that these isolated vessels are either used for registration processes or for diagnostic purposes.

Chapter 4 discussed the proposed setup. The physical setup and the desired outcome of the algorithms were briefly mentioned. The usefulness of the proposed system referring to economic factors as well as the diagnostic benefits were commented on.

The design of the software algorithms to achieve registration for this specific task, were discussed in Chapter 5. The initial focus was on the nature of the captured imagery. It was emphasised that understanding the characteristics that made the situation unique, were imperative to the design of the custom algorithms. Some of these characteristics were elaborated on and it was mentioned as to how they could hinder the process.

To conduct the registration, there was heavily relied on spatial cross correlation as it proved to be efficient in a setting where the imagery is of limited quality. It was noted that this is often the case for medical imagery.

The initial problem addressed was that of rapidly parsing the input videos to isolate frames to be processed further. This was achieved with the use of colour-thresholding techniques. This method of isolating candidate frames was not completely accurate but provided an efficient way of sorting the bulk of the frames.

With the candidate frames selected, a method of filtering was discussed. This filtering provided a method of determining the amount of content that a frame contains within the desired frequency range. The high frequency content was typically the blood vessels. These blood vessels proved ideal to register the frames by.

Spatial cross correlation was then discussed as it served two functions. Firstly, it was used to determine the offset between consecutive frames and secondly the peak value was used to gauge the number of features contained within a frame compared to its predecessors. This comparison allowed for efficient grouping of consecutive frames based on the relative amount of features they contained.

After the grouping of frames into clusters were discussed, methods of efficiently grouping clusters were investigated. Two methods for rapid matching of clusters were demonstrated and compared. The first was the use of a down sampled descriptor that allowed for faster conduction of cross correlation. The down sampled cross correlation offered a method of efficiently calculating a low resolution estimate to where clusters overlap and an indication of how many features overlap. In comparison to the down-sampled descriptor, the matching of HoG Descriptors was proposed. The HoG Descriptor offers a summary of the gradients contained within an image. The HoG descriptors were considerably faster to compare and required a fraction of the memory compared to the down-sampled descriptors. The HoG descriptor took preference in terms of speed with a small sacrifice in accuracy.

With matches compared within a set of clusters a method was devised to measure the accuracy of a match. Characterisation of the cross-correlation peak shape was discussed as well as the use of normalised cross correlation as a scoring method.

Registration of filtered frames were conducted and the averaging of the filtered frames were presented in Chapter 6. The results were compared to the quality of single frames captured with the setup as well as with images taken with a commercial digital fundus camera. It showed a clear increase in the captured spatial resolution of the combined images. A larger field of view was obtained by combining several videos frames that jointly covered a larger region.

The system was designed to be robust against variables as well as efficient in terms of computation power required. It was kept robust by designing algorithms to use relative parameters for sorting and registration. Frames and clusters were grouped according to relative feature similarities. This approach avoided the use of hard thresholds to classify data and allowed the better solutions to surface automatically. Here, the better solutions were the clusters with the most correlating high frequency data.

Efficiency was kept in mind throughout the design and implementation of the system. Where possible, simpler, but more efficient methods were favoured.

During the implementation, optimisation was maintained by retaining as little as possible resources in primary memory. Once a frame has been processed its data was released and only the descriptors of the clusters retained in primary memory. The clusters were sorted and the best ones were reconstructed by re-fetching the frame data from secondary memory.

These optimisations and the promising results in terms of the constructed imagery, makes the proposed system an attractive solution in resource constrained environments. The efficiency also makes the complete implementation of algorithms on the mobile device an attractive approach. With the ever-increasing mobile processing power, this could be realised in the near future.

Appendix A

Image Processing Concepts and Common Operations

A.1 Introduction

Image processing is signal processing methods applied to a 2D discrete signal. The signal is in the form of a digital image. The output of image processing algorithm can also be another image or it could be certain parameters extracted from the image. In most cases the input is converted to either an representation or a decision.

In this Appendix, we aim to give an overview of the common terms and concepts of image processing.

A.2 Image Representation

A digital image is described by a 2D array of pixels, with individual pixels each assuming a different colour. A pixel is the smallest spatial discretisation described within an image and an image processing algorithm typically operates on these individual pixels, where the pixel values are seen as discrete two dimensional signals.

A pixel, is in turn, is described by a number of intensity values that are combined to represent a specific colour. A pixel can consist of any number and range of intensity values depending on the colour space and depth of an image.

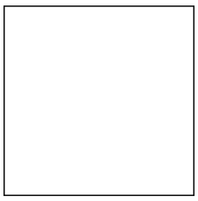
Intensity values can require different sizes, depending on the number of bits used to describe the pixel. This is referred to as the depth of the images. The higher the depth of a pixel, the greater its ability to reflect small changes in light intensities. Together with the image resolution, the depth determines the amount of memory an image requires to be stored. For a higher resolution and higher depth, more memory is required to load and process an image.

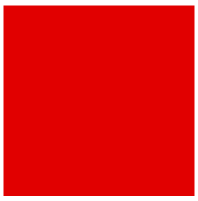
Image resolution is a term often used when working with digital images. Resolution commonly covers two concepts, namely spatial and pixel resolution. Pixel resolution refers to the size of the image where an image with a higher pixel resolution consists of a larger number of pixels. Spatial resolution is often a more subjective term and refers to the level of detail contained within the image. In most instances, the term resolution would refer to the pixel resolution of an image.

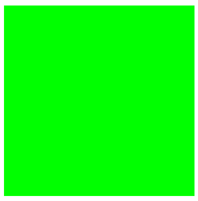
The typical representation of an image is 24-bit RGB. Here each pixel is represented by three components, red, green and blue and each of these components are described by an 8-bit value. The image is said to consist of three colour channels.

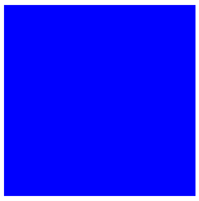
A 24-bit RGB pixel would thus consist of three intensity values that range from 0 to 255. With 0 indicating that a colour component is not present, while 255 indicates that the component assumes the strongest value for the pixel. Shown below, in figure A.1, are a selection of 24-bit colours with their respective RGB values, written in the format: (Red,Green,Blue)

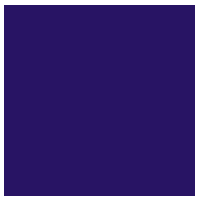
(a) (0,0,0)


(b) (255,255,255)


(c) (255,0,0)


(d) (0,255,0)


(e) (0,0,255)


(f) (40,20,100)

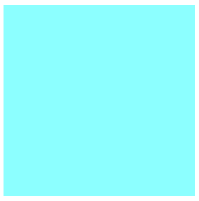

(g) (140,255,255)

Figure A.1: A selection of RGB colours.

The RGB Colourspace is one of many that is used. The next section describes some of the common colour spaces utilised in image processing.

A.3 Colour Spaces

Image processing algorithms benefit from using multiple colour spaces. A colour space describes the specific way in which a pixel's intensity values are combined to map a certain colour. Switching between colour spaces is often useful since a particular one might be better at highlighting certain features within an image than another.

Below some of the common colour spaces are described. Their origin or mathematical formulation are not discussed; their appearance is simply noted. Also note that switching between spaces can be conducted using specific mappings. For further reading on the development of these colour spaces, consult [39].

Figure A.2 demonstrates the contribution of the three colour channels in the RGB space that are combined to form the image. Usually when a single channels is extracted it is shown as a grayscale intensity map. However, in this example two of the three channels were zeroed to demonstrate the colour of the third. Grayscale intensity maps are discussed later in this section.

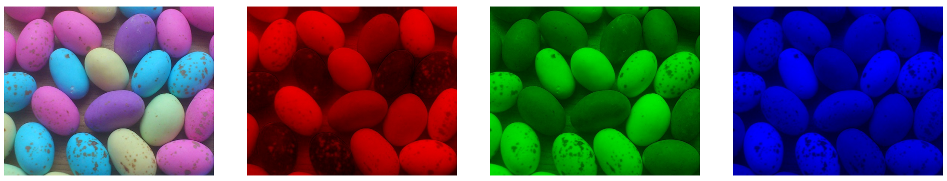

(a) Full colour (b) Red channel (c) Green channel (d) Blue channel

Figure A.2: An image split into its RGB colour channels.

The RGB space is modelled in the way in which the human eye perceives light with specific rods and cones sensitive to each of these components [39]. The values of the components are typically expressed as a value in a range between 0 and 255, but can be written in a normalized fashion as well. Figure A.3 shows a 3D representation of the colour space.

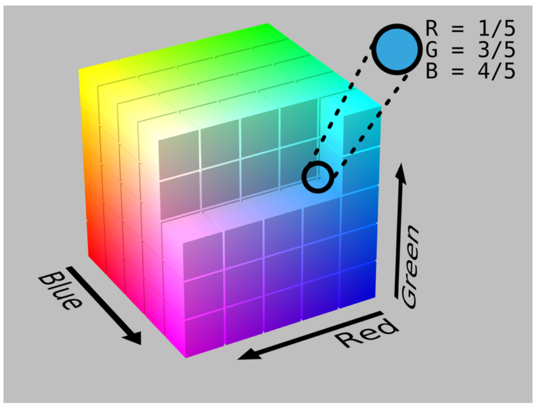
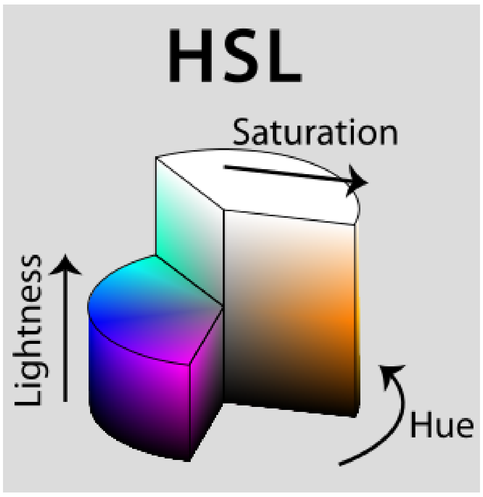


Figure A.3: The RGB colour space [22]

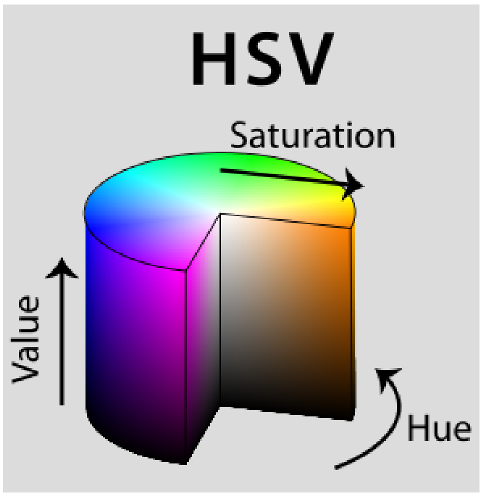
A shortcoming of the RGB space is that it does not directly describe the brightness of the colour. The brightness is implied from the combination of the three components with a normalised RGB value of (1,1,1) being the brightest colour, white, and (0,0,0), being the darkest, black. The following two colour spaces addresses this problem by dividing the colour space into separate colour and intensity components.

The HSV and HSL spaces are shown in figure A.4. Both are represented by a cylindrical coordinate system. HSV and HSL stand for Hue, Saturation, Value and Hue Saturation Lightness respectively. Note that the hue in both colour spaces represents the actual colour and that the brightness of the colours are determined by the saturation-, lightness- and value components.

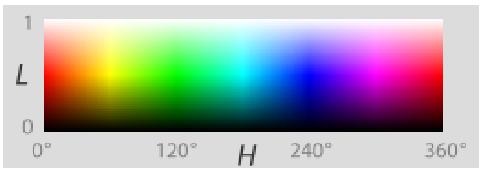
Hue is typically expressed in degrees of a 360° rotation. The hue at 0° is red, and moves through the colour spectrum and returns to red at 360° . This is illustrated in both figure A.4c and A.4d.



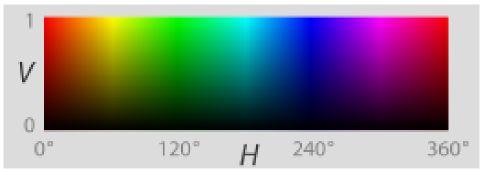
(a)



(b)



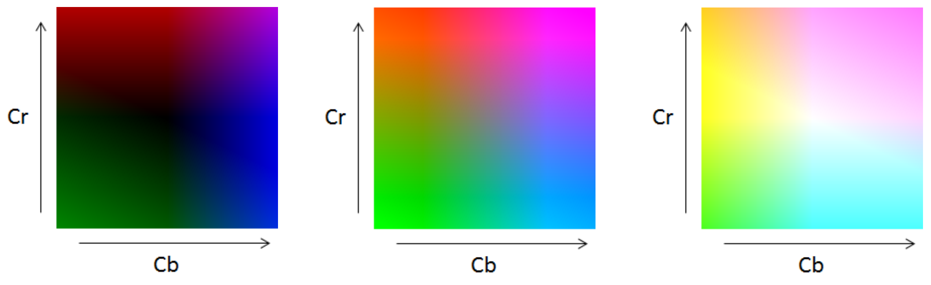
(c) HL colour space



(d) HV colour space

Figure A.4: HSL and HSV colour space representations [35].

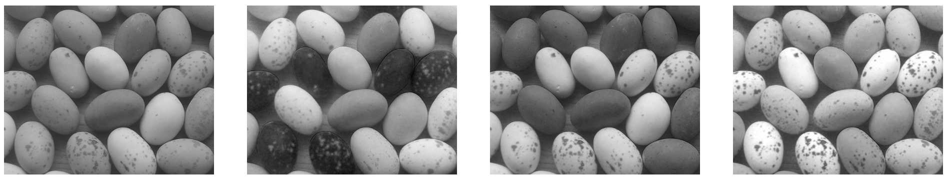
Another common colour space is the YCrCb space. Here a pixel is described by two colour components and the brightness by a third component. The brightness is described by the luma component (Y) and the colour by Chroma components; Chroma Red (Cr) and Chroma Blue (Cr). This colour space only has two colour components as it aims to eliminate redundancy present in the RGB colour space. Figure A.5 show slices of the YCrCb colour space for different normalised values of Y.



(a) Y = 0 (b) Y = 0.5 (c) Y = 1

Figure A.5: Slices of the YCrCb colour space.

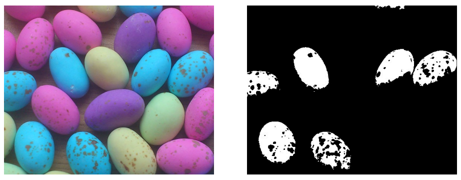
For most image processing operations, images are converted to a grayscale colour space to simplify the process. A grayscale image consists of only channel, so each pixel is only described by one intensity value. A colour image can be converted to grayscale by either performing some operation on the colours to yield a single gray intensity value or by extracting a single colour channel and using the intensities as gray-level intensities. Figure A.6a shows an RGB colour image from figure A.2a converted to grayscale by averaging the RGB values. Figures A.6b, A.6c and A.6d show the different channels extracted from the RGB channels to yield gray-level representations.



(a) Gray Averaged (b) Red channel (c) Green channel (d) Blue channel

Figure A.6: An image split into its RGB colour channels.

The last colour space, or rather image representation that is discussed, is variations of grey scale images. Binary images are a subset of grayscale images where the intensity is only described by one or zero, black or white. Binary images are typically used to mark areas of interest and are also referred to as binary masks. Areas of interest are isolated typically by using thresholding techniques with the result stored as a binary mask. Figure A.7 shows an image thresholded for blue areas and the resulting binary mask is shown in figure A.7b. This thresholding, as well as other intensity mappings, are described in the next section.



(a) (b)

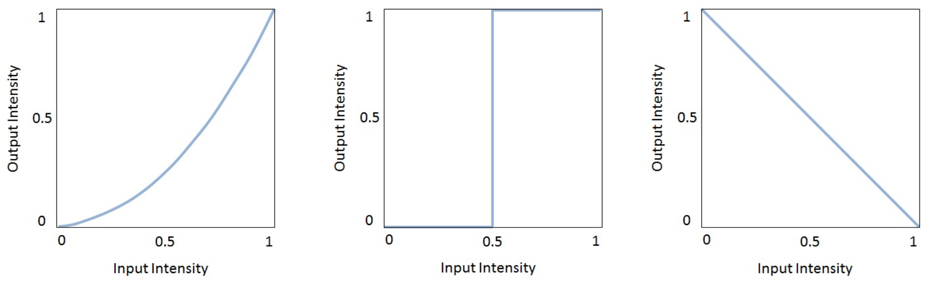
Figure A.7: Thresholding for blue regions in an image.

A.4 Intensity Transformations

Intensity transformations is the mapping of single-pixel intensity from an input image to a transformed one. Each of the pixel intensities in the input are subjected to some transformation (T) as illustrated in equation A.1.

$$g[x, y] = T[f[x, y]] \tag{A.1}$$

Figure A.8 shows three typical mappings between normalised intensity values. The first transform, shown in figure A.8a, is a power mapping where the input is raised to a number n. This is utilised for increasing the contrast of an image. The second transform, shown in figure A.8b, is an thresholding transform. Here values below a certain cut-off value is mapped to zero and the others to one. This transform is utilised as a simple method for isolating regions of interest in an image. The last transform, shown in figure A.8c, is an inverse transform and is used to perform an inversion of the input's intensity values.

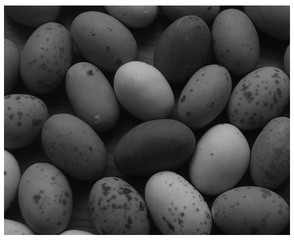


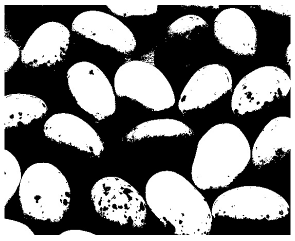
(a) (b) (c)

Figure A.8: Commonly used intensity transforms.

The results of applying these transforms to a grayscale image, are illustrated in figure A.9.

(a) Input image


(b) Power transformation


(c) Threshold transformation

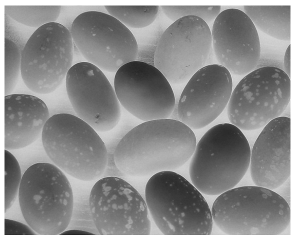

(d) Inverse transformation

Figure A.9: Examples of intensity transformations.

A.5 Spatial Filtering

Spatial filtering is a common operation in image processing. It is utilised for tasks such as smoothing an image, determining the cross correlation between images or highlighting and isolating certain features within an image. If intensity transformations, discussed in the previous section, describes the mapping of a single pixel's intensity from an input image to an transformed image, then spatial filtering describes the mapping of a group of pixels of an input image to the transformed image.

With spatial filtering, a 2D kernel is convolved or correlated with an input image. Figure A.10 illustrates the process where a 3x3 kernel is convolved over a input image. The kernel is slid over the input image with a one pixel increment at a time. At each position the kernel's coefficients are multiplied with overlapping pixel values in the input image. The sum of the multiplied coefficients at each position, yields the output image's values.

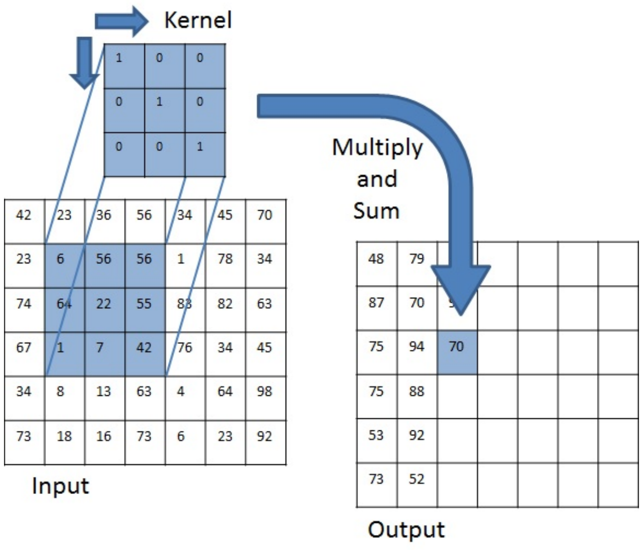


Figure A.10: Kernel convolution.

2D convolution is analogous to 1D convolution applied to a discrete signal. Equation A.2 demonstrates the process of a 1D discrete convolution while equation A.3 demonstrates the convolution equation expanded for a 2D discrete signal.

$$f[x] * g[x] = \sum_{k=-\infty}^{\infty} f[k].g[x-k] \tag{A.2}$$

$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2].g[x-n_1,y-n_2] \tag{A.3}$$

Similar equations can be written for image correlation. The difference for correlation, is that the kernel is not flipped as with convolution. Equation A.4 and Equation A.5 demonstrate 1D and 2D cross correlation between two discrete signals.

$$f[x] \star g[x] = \sum_{k=-\infty}^{\infty} f[k].g[x+k] \tag{A.4}$$

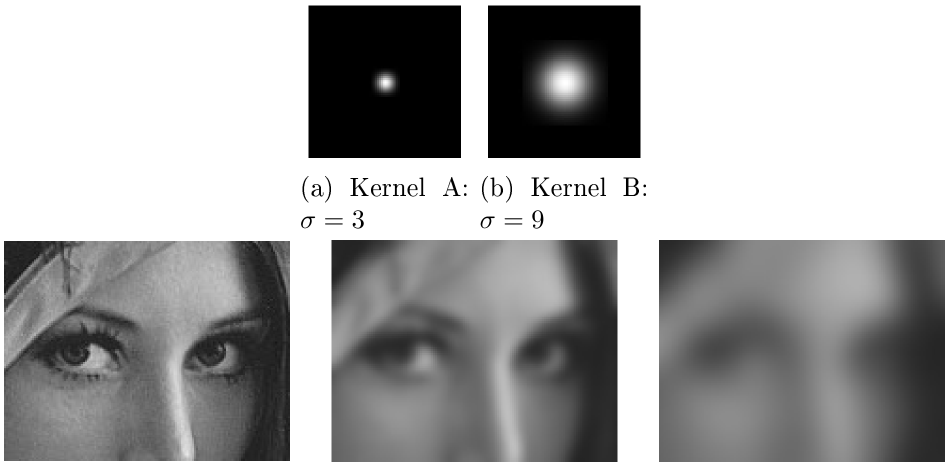
$$f[x,y] \star g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2].g[x+n_1,y+n_2] \tag{A.5}$$

Note that in practice the convolution and correlation can not be calculated for pixels on the border of the input images since all the values of the kernel would not overlap with input image values. Two methods of coping with this limitation are available: either the input image could be padded with additional pixels, or the output image would have a reduction in size. For padding the image, the following condition applies. The amount of pixels padded around the image, is equal to $(N-1)$, where N is an odd number that denotes the size of an $N \times N$ kernel. For typical applications, the size of N is chosen to be odd. Note that this is the amount of padding required to yield an output image with the same size as the original input. In the case of reducing the size of output image the size is as follow: $(I.width - K.width + 1; I.height - K.height + 1)$, where the kernel is denoted by K and the input image by I.

Image smoothing is the most common application filtering. A number of different kernels exist, but the principle remains the same: a pixel in the output image is based on the sum, or a weighted sum, of adjacent pixels in the input image. The patch wise average of groups of pixels in the input image, is thus calculated. This averaging causes the loss of high-frequency image components and yields a smooth output. For these reasons, the smoothing filters are also referred to as low-pass or averaging filters. A feature of these smoothing operations is that a larger kernel offers more smoothing.

A kernel commonly used for smoothing operations is the Gaussian kernel. Here the kernel consists of a 2D Gaussian distribution. The Gaussian distribution is given by equation A.6. The Figure A.11 illustrates the application of two Gaussian kernels with different values to an image. Note that the larger σ value yields more smoothing. Often the results are subjective and chosen for the desired result based on appearance.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}} \tag{A.6}$$

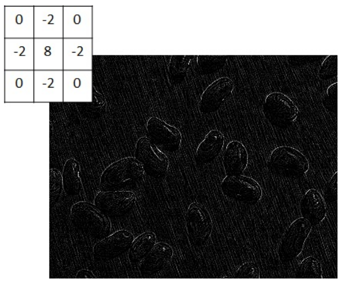

(a) Kernel A: $\sigma = 3$ (b) Kernel B: $\sigma = 9$

(c) Input image (d) Smoothed with kernel A (e) Smoothed with kernel B

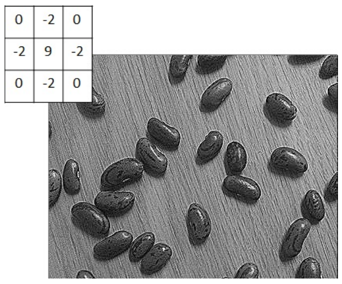
Figure A.11: Gaussian smoothing with different kernel sizes.

Another application of spatial filtering, is sharpening and edge detection. With image sharpening, the high frequency components of an image are highlighted, yielding pronounced edges. An edge detection spatial filter highlights the high frequency components and discards the lower frequencies, yielding just the sharp edges contained within the image. Figure A.12 illustrates the use of a sharpening- and an edge-detection filter applied to an image. Note the similarity of the two kernels with only the centre coefficient varying.

(a) Original image



(b) Edge detection



(c) Image sharpening

Figure A.12: Spatial filtering for edge detection and sharpening.

Morphological operations applied with spatial filters is the last application to be discussed discuss. With a morphological operations, the kernel differs from the convolution kernel. The morphological kernel simply specifies a logic rule as the kernel is convolved over the image and does not specify numerical values for each element in the kernel. Morphologic operations can typically be used to isolate or merge certain regions or objects in a boolean image. The four main morphological operations are: dilate, erode, open and close.

Dilation is used to enlarged structures in a binary image, while erosion is used to shrink them. Dilation is typically achieved with the use of a 3×3 kernel where the output value is determined by the maximum value in the area covered by the kernel. Erosion is achieved by using the minimum value in the area covered by the kernel. Figure A.13 demonstrates the dilation of an image using a 3×1 kernel where the mapping of two positions of the kernel is illustrated.

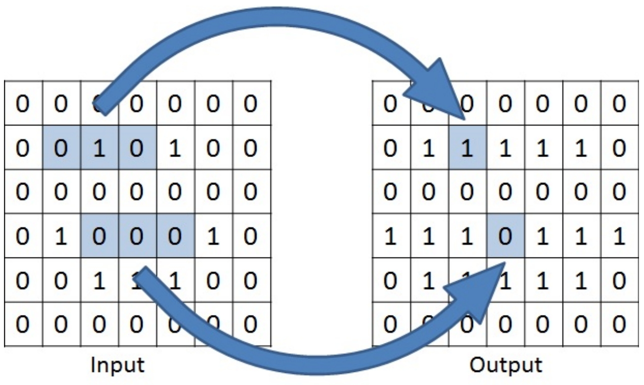
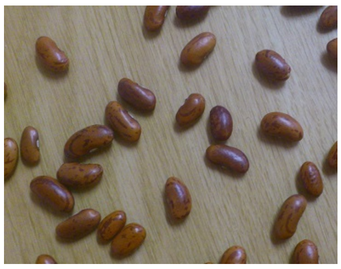


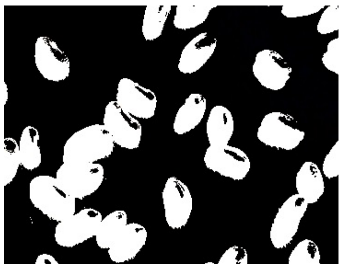
Figure A.13: The operation of morphological filters.

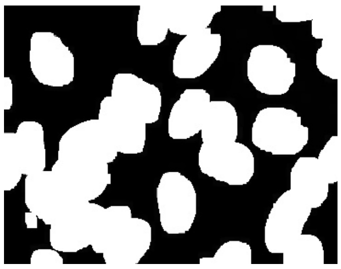
The open operation is achieved with a erode operation followed by a dilate operation. The closing operation is achieved with a dilate operation followed by a eroding operation. These two operations are useful for separating or joining structures of an image whilst maintaining their size.

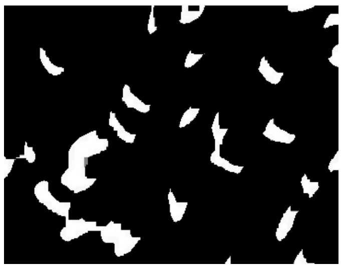
Figure A.14a shows an image of beans and a thresholded version of the image in figure A.14b. The the four morphological operations are applied to the thresholded image to demonstrate their effects. In figure A.14d, we note that the erosion operator decreases the size of the rice grains. The size of the grains are increased with the dilate operation shown in figure A.14d.

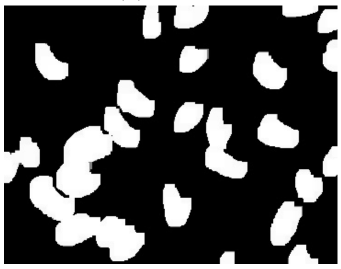
The more useful operations are the open and close operations as illustrated in figure A.14. Note that the open operation separates adjacent entities in the input image, while the close operation joins them. However, these operations maintain the original size of the entities.


(a) Original


(b) Mask


(c) Dilate


(d) Erode


(e) Open

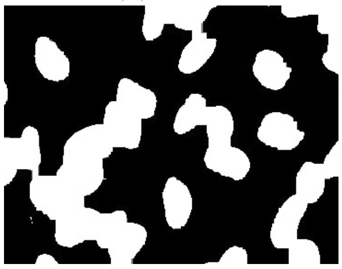

(f) Close

Figure A.14: The effects of different morphological operations.

Appendix B

OpenCV

OpenCV, as the name suggests, is an open source library for computer vision. It is written in C and C++ with builds for a number of systems, including Windows and Android. OpenCV has a strong focus on realtime video processing, but is also employed for other image processing tasks. The library is free for academic and commercial use under the BSD license agreement. Visit the OpenCV website for additional information and download links to the various library builds at: `http://opencv.org`. Full documentation, with API and tutorials, is available here: `http://docs.opencv.org`.

Apart from the standard builds for OpenCV, there exists ports and wrappers that allow use on other platforms not officially supported. A wrapper of particular interest is a .NET wrapper known as EMGU. The EMGU wrapper allows the use of OpenCV's optimised image processing algorithms while leveraging .NET's capabilities for the rapid development of interactive Windows desktop applications.

For the implementation of the proposed algorithms in the text, the EMGU library is selected and development conducted in C#. The IDE used is Microsoft's Visual Express that allows free use for non-commercial purposes. Using this environment, Windows desktop applications were created that use OpenCV functions at its core.

The translation of the algorithms for use on any other OpenCV supported platform is conducted with relative ease since the EMGU documentation specifies which base OpenCV functions are utilised. This allows for the easy implementation of the algorithms directly on a mobile device, running Android or IOS, once they are fully refined on a higher-level testing platform.

The EMGU installation, API documentation and Wiki is available here: `http://www.emgu.com`. The EMGU download offers an installation file that installs the SDK. To use the libraries in a .NET application, one simply references the EMGU DLLs provided in the SDK.

In the following three paragraphs we illustrate the simple use of the EMGU library in C#. First we discuss how an image is represented, then it is demonstrated how a video file is iterated and lastly how an individual frame is processed. For an in depth guide to the algorithms available in OpenCV, their use, practical implementation and mathematical equations, consult Learning OpenCV [3]. Learning OpenCV offers the C++ code for a vast array of applications, however, these examples are found to be interpreted and translated easily for use with EMGU.

Image Representation Images can be represented in multiple ways. The image is declared based on its colour space and depth. The colour space can assume any of those discussed in Appendix A.3 as well as a multitude of others. The depth can assume any of the number depths available in the C# runtime.

Shown below is the declaration of images with different colour spaces and depths. To instantiate these images, a parameter has to be passed to indicate the size of the image. This allows for the allocation of sufficient memory to store the image.

```
Size size = new Size(50, 50);

Image<Bgr, byte> image_a = new Image<Bgr, byte>(size);
Image<Ycc, double> image_b = new Image<Ycc, double>(size);
Image<Gray, float> image_c = new Image<Gray, float>(size);
Image<Hsv, int> image_d = new Image<Hsv, int>(size);
```

The number of channels assigned to an image depends on the colour space. Most, besides the gray space, consist of three. The code section below demonstrates the separation of the channels of a colour image. The extracted channel is inherently a grayscale image. Since the middle channel is extracted, the grayscale image represents the green channel of the original image. Note that the depth of the two images have to remain the same during the extraction since only a pointer to the existing data is used for the extracted image.

```
Image<Bgr, byte> image = new Image<Bgr, byte>(size);
Image<Gray,byte> green_channel = image.Split()[1];
```

Converting between colour spaces is conducted as demonstrated in the following section.

```
Image<Bgr, byte> image = new Image<Bgr, byte>(size);
Image<Hsv,float> converted_image = image.Convert<Hsv,
   float>();
```

Here a new depth can also be specified, since a new image is created in memory for the converted intensity values.

Capture A video file consists of a series of still images captured at a certain frame rate. To perform operations on a video sequence, the frames can be accessed sequentially from the start of a video file. This is achieved using the EMGU capture object. For processing files, a *Capture* object is created that points to the video file. The capture object is then queried repeatedly to retrieve images from the video file. The code below shows a typical implementation.

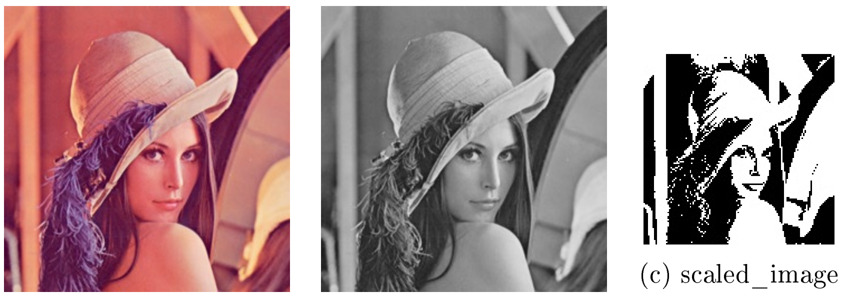
```
Capture capture = new Capture("c:\test.avi");
Image<Bgr, byte> video_frame = capture.QueryFrame();

while (video_frame != null)
{
        ProcessVideoFrame(video_frame);
        video_frame = capture.QueryFrame();
}
```

This section shows the common approach to processing a video using OpenCV or EMGU. The file is processed sequentially until the *Capture* object returns a null, indicating that the end of the file has been reached. Random access to the video file is allowed, but is considerably slower. Video files are typically compressed and the content of a specific frame is compressed along with the content of its predecessor. The accessing of a random frame would thus require the decompression of several of its predecessors.

Image Processing Once an image has been loaded into memory, it can be subjected to a vast number of operations. The operations, or types of operations, are too vast to discuss here. In the section below we give a simple example of some operations performed on an image. This image could be part of a video sequence or a still image obtained through a number of ways.

```
Image<Bgr, byte> image = new Image<Bgr,
   byte>(@"C:\image.jpg"); //Creates a new image form
   content located on the disk

Image<Gray, byte> gray_image = image.Convert<Gray, byte>();
   //Converts the RGB image to a Gray image by averaging the
   channels

Image<Gray, byte> scaled_image = gray_image.Resize(0.65,
   INTER.CV_INTER_CUBIC); //Resizes the Gray image using
   bi-cubic interpolation

scaled_image._ThresholdBinary(new Gray(128), new Gray(255));
   //Thresholds the gray image for pixels with an intensity
   greater than 128 and sets their intensities to 255

gray_image.Save(@"C:\gray_image.jpg");
scaled_image.Save(@"C:\scaled_image.jpg");
```

Figure B.1 shows the input and output files of the algorithm. Note the in-place application of the thresholding to the scaled image. In place operations are considerably more efficient since the image does not have to be copied from one memory location to another processing can start.



(a) image.jpg (b) gray_image.jpg (c) scaled_image

Figure B.1: Image processing example.

Appendix C

Code Discussion

In this section we discuss key points in the implementation of the algorithms and give code samples where appropriate. Two key structures are designed, namely the *Frame*- and *Cluster* class as illustrated in the UML diagrams in figure C.1. Each instance of the *Frame* class maps to a specific frame in the video sequence while the *Cluster* class consists of a selection of *Frame* objects that correlate and cover a specific region of the retina.

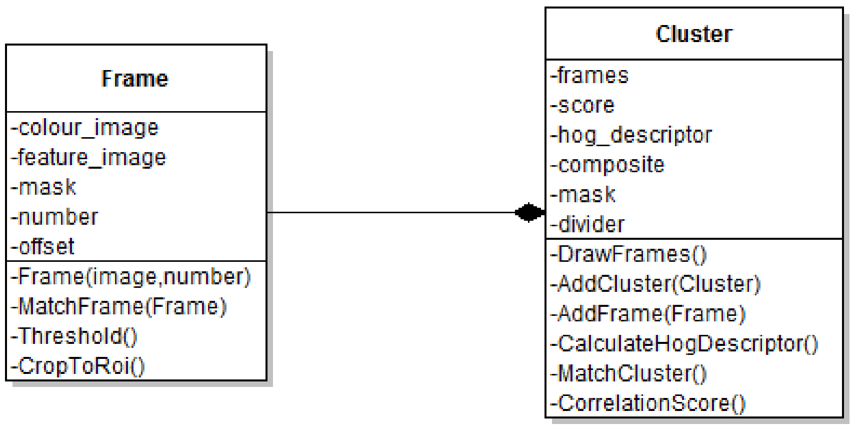


Figure C.1: UML diagram illustrating two main classes used for the implementation.

We discuss the two classes separately and aim to give a description of their key properties and methods. First, however, we note the leverageing of image windowing when utilising EMGU.

C.1 Windowing

OpenCV(and EMGU) offers the ability of having two images reference the same underlying data. Typically, an image would be created in memory and another smaller image pointer would be created to point to a subsection of this larger image. The area that the smaller images references is referred to as an ROI (region of interest) of the larger image. The ROI is specified as a rectangle with an offset and a specific length and width. The code section below demonstrates the use of an ROI rectangles to reference a subsections of a larger image. Operations that are performed on the on the subsections, are reflected in the larger image since they reference the same data.

```
Image<Gray, byte> image_a = new Image<Gray, byte>(new
   Size(60, 60));
image_a.SetValue(255);

Rectangle roi_b = new Rectangle(new Point(10, 10), new
   Size(30, 30));
Rectangle roi_c = new Rectangle(new Point(20, 20), new
   Size(30, 30));

Image<Gray, Byte> image_b = image_a.GetSubRect(roi_b);
Image<Gray, Byte> image_c = image_a.GetSubRect(roi_c);

image_b._Mul(0.5);
image_c._Mul(0.5);

image_a.Save(@"C:\image_a.jpg");
```

The result of the above code is shown in figure C.2. Here the halving of the intensity of *image_a* is performed on two sections.

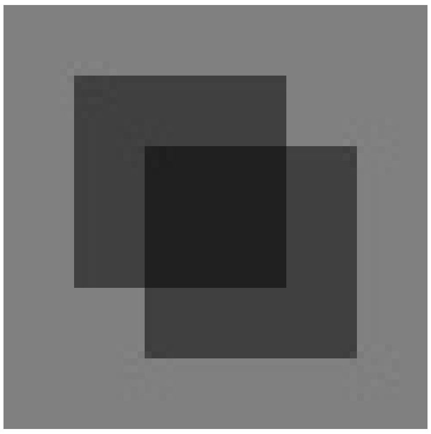


Figure C.2: Example of an operation applied to two sections of a larger image.

The benefit of using the subregions is that certain operations can be performed on a small region of an image with little overhead while keeping the larger image intact. For the proposed application it proves beneficial since cross correlation can be conducted on small sections of larger images. Another benefit is that by referencing the subsection directly the costly operation of copying the image to another memory location is avoided.

C.2 Frame Class

As mentioned, each instance of the *Frame* class maps to a specific video frame. As a video is parsed, a *Frame* object is created for each video frame in the sequence. Many of these frames are quickly removed if they are not deemed fit for use. After the *Frame* objects have been created, operations are performed to highlight and extract data and they are then grouped together to create a cluster.

Figure C.1 outlined the key methods and properties for a frame object. The following paragraphs below describes each of these:

colour_image This image contains the actual content grabbed from the video. From this colour image an ROI and mask are extracted. This images is also thresholded to extract the blood vessels and form the *feature_map*.

feature_map This grayscale image contains the highlighted features from extracted from $colour_image$.

mask The mask indicates the area of the frame possibly containing retina and is calculated using thresholding methods on *colour_image*.

offset This is a point structure indicating the offset of a frame relative to a fixed reference in the *Cluster*.

roi The ROI indicates the region of the image that contains retina and is equal to the bounding box around the area marked by the mask.

frame_number This number indicates the frame in the video sequence to which the object corresponds. It allows for re-fetching of data at a later stage.

Threshold As discussed in the text, thresholding is conducted in the YCrCb colour space. The image is initially split into the three channels and the tresholding is applied to each of them separately. The results are then combined using an "AND" operation. The listing below demonstrates how the thresholding is executed. Note that the mask is calculated on a down-sampled image to increase the speed of the operation.

```
Image<Bgr, Byte> thumbnail;
Image<Gray, Byte> mask_temp;
Image<Ycc, Byte> ycc;

mask = new Image<Gray, byte>(color_image.Size);

mask.SetZero();

thumbnail = color_image.Resize(color_image.Width /
   SCALE_FACTOR, color_image.Height / SCALE_FACTOR,
   INTER.CV_INTER_CUBIC);

ycc = thumbnail.Convert<Ycc, Byte>(); //Convert RGB image to
   YCrCb colour space


Image<Gray, Byte>[] channels = ycc.Split();

Image<Gray, Byte> y_channel = channels[0];
Image<Gray, Byte> cr_channel = channels[1];
Image<Gray, Byte> cb_channel = channels[2];

cr_channel._ThresholdBinary(new Gray(128), new Gray(255));
   //In place thresholding of the Cr channel

Image<Gray, Byte> tmp;

cb_channel._ThresholdBinaryInv(new Gray(128), new
   Gray(255)); //In place thresholding of the Cb channel

tmp = y_channel.ThresholdBinary(new Gray(10), new
   Gray(255)); //Thresholding of the Y channel to eliminate
   dark colours

y_channel._ThresholdBinaryInv(new Gray(128), new Gray(255));
   //Thresholding of the Y channel to eliminate bright
   colours

y_channel._And(tmp); //Combining Y channel masks

mask_temp = cr_channel.And(cb_channel);  //Combining Cr and
   Cb masks

mask_temp._And(y_channel);  //Combining all masks
```

```
mask = SelectLargestRegion(mask); //Selects largest region
   in mask

mask = mask_temp.Resize(color_image.Width,color_image.Height,
INTER.CV_INTER_CUBIC); //Resize mask to original size
```

CropToRoi Once the mask is calculated, the ROI is determined, indicating the area containing retina and the *feature_map* is cropped to only include this region. This is achieved by calculating the bounding rectangle around the area indicated by the mask and discarding areas in *colour_image* that do not fall within this region.

```
this.roi  = FindRoi(); //Obtains the bounding box from the
   frame's mask

mask = mask.GetSubRect(roi);
color_image = color_image.GetSubRect(roi);
```

MatchFrame The *MatchFrame* method aims to determine the offset of a frame's content relative to another frame. This method is reserved for consecutive frames and when calling it, a close-to-zero offset between the two frames are expected. It is important to note that all frames in a cluster will have an offset calculated in a shared coordinate system. The offset would point from the origin of the cluster to the top left corner of a cropped section of frame as illustrated in figure C.3 where the, V_1 and V_2, describe the positions.

A problem arises when the frames are cropped according to their ROIs, since the close-to-zero offset is compromised. This is demonstrated in figure C.3 where the overlapping ROIs for two consecutive frames are shown. If there is a large shift in ROIs positions, the close-to-zero translation estimate does not hold true, even though the actual scene did not experience any translation.

In figure C.3 the actual offset has not been determined yet, but in this instance we estimate it to be zero ($V_{SceneTranslation} = 0$). The frames are expected to overlap with a zero offset, however, the overlap of the ROIs will differ.

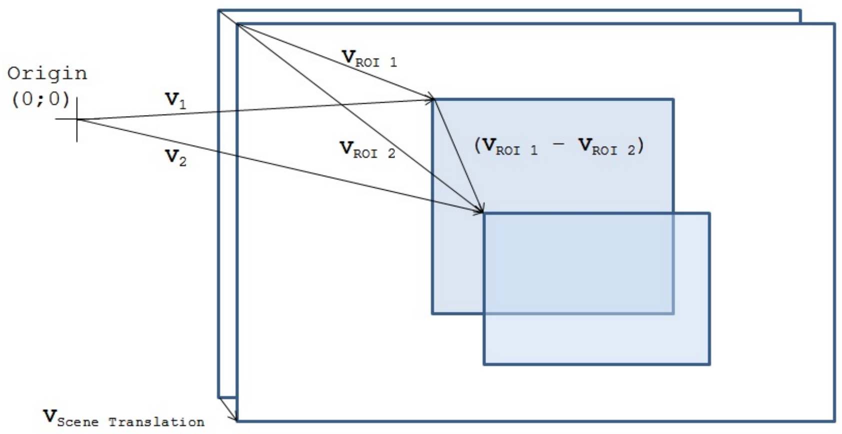


Figure C.3: Vectors describing the positions of overlapping consecutive frames and their ROIs.

In this scenario the goal is to determine the vector V_2, given all the other information. If the scene translation is ignored, V_2 can be expressed as shown in equation C.1.

$$V_{Frame2} = V_{Frame1} + (V_{ROI2} - V_{ROI1}) \tag{C.1}$$

In the code implementation, a rough estimate of $V2$ is calculated and $V_{SceneTranslation}$ is then estimated by calculating the cross correlation of the overlapping sections of ROI. The code listing below show the initial calculation of V_2.

```
frame.offset.X = this.offset.X + (frame.roi.X - this.roi.X);
frame.offset.Y = this.offset.Y + (frame.roi.Y - this.roi.Y);
```

After the translation for a cropped section is estimated, the original concept of a subregion in a larger frame is discarded and the frame simply treated as a clipped section with an absolute offset in a cluster.

Using the initial estimate of V_2, the overlap of the two sections are extracted. $Frame1$'s image is padded with a predefined amount of pixels (i.e. 5 pixels). The cross correlation between the padded section for $frame1$ and the section for $frame2$, is calculated. The listing below shows the padding as well as the correlation calculation.

```
this.CalculateOverlap(frame); //Calculates the overlapping
   sections of the frames. The overlap is stored as a
   private member for both frames.
```

```
Image<Gray, Byte> image_a = new Image<Gray,
   byte>(this.overlap.Size); //Creates an image for the
   frame's overlapping section
Image<Gray, Byte> image_b = new Image<Gray,
   byte>(frame.overlap.Size);//Creates an image for the
   frame's overlapping section

frame.feature_map.GetSubRect(frame.overlap).CopyTo(image_b);
   //Copies the data for the frames overlapping section
this.feature_map.GetSubRect(this.overlap).CopyTo(image_a);
   //Copies the data for the frames overlapping section

Image<Gray, Byte> image_a_padded = new Image<Gray,
   Byte>(image_a.Width + window, image_a.Height + window,
   new Gray(0)); //Creates an image slightly larger than
   image_a

CvInvoke.cvCopyMakeBorder(image_a, image_a_padded, new
   Point(window/2, window/2), BORDER_TYPE.CONSTANT, new
   MCvScalar(0));//Pads image_a

Image<Gray, float> cross_correlation = new Image<Gray,
   float>(image_a_padded.Width - image_b.Width + 1,
   image_a_padded.Height - image_b.Height + 1); //Creates an
   image for the cross-correlation result

cross_correlation = image_a_padded.MatchTemplate(image_b,
   TM_TYPE.CV_TM_CCORR);
```

Once the correlation is calculated, $V_{SceneTranslation}$ is extracted from the correlation result by locating the peak. The listing below shows the locating of the peak and the addition of $V_{SceneTranslation}$ to the initial V_2 estimate.

```

//Output parameters for finding max location in
   cross-correlation (Only using max_val)
Point max_loc = new Point(0, 0);
Point min_loc = new Point(0, 0);
double min_val = 0;
double max_val = 0;

CvInvoke.cvMinMaxLoc(cross_correlation.Ptr, ref minVal, ref
   maxVal, ref minLoc, ref maxLoc, IntPtr.Zero);//Search
   image for minimum and maximum points

```

```
//Adds the value of Vscene translation to V2
frame.offset.X += maxLoc.X - window/2;
frame.offset.Y += maxLoc.Y - window/2;
```

As discussed in the text, in order to avoid the phenomenon of dead reckoning, each new frame is averaged with the previous frame. The listing below demonstrates the accumulation of the data.

```
this.CalculateOverlap(frame);//New overlap is calculated for
   refined offset

image_a = feature_image.GetSubRect(this.overlap);//Select
   overlapping region
image_b = frame.feature_image.GetSubRect(frame.overlap);
   //Select overlapping region

image_a._Mul(0.5); //Adjust accumulating weight
image_b._Mul(0.5); //Adjust accumulating weight

image_a = image_a + image_b;
```

C.3 Cluster

A *Cluster* object consists out of a number of *Frame* objects that are registered to a fixed reference point. In this section we describe the methods and properties as outlined in figure C.1.

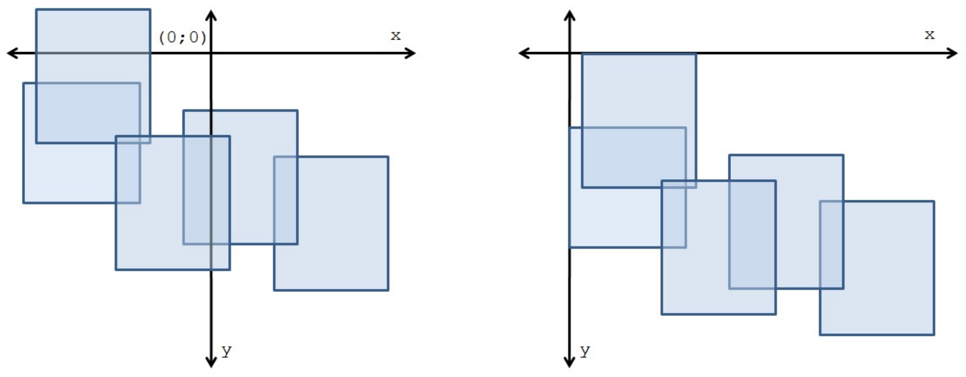
frames - A list of *Frame* objects that together form the content of the *Cluster*.

hog_descriptor - The histogram of gradients descriptor for the cluster.

composite - An image composed from the $feature_maps$ of the individual *Frame* objects.

divider - A map of the amount of data accumulated for specific areas of the *composite*.

RealignFrames When the frames are registered, the location of the first frame in the cluster is regarded as the origin. This has the effect of a scene translation in the western direction to yield frame offsets that are negative. The realign method shifts the frames such that they all have a positive offset. Figure 3.11b demonstrates this.



(a) Frames with a negative offset (b) Realigned frames

Figure C.4: Offset shift of a set of frames.

After the frames are shifted, the required size for the composite image is determined by locating the furthest data points on the x- and y-axis.

The listing below demonstrates the shifting and size estimation.

```
int x_min = Int32.MaxValue;
int y_min = Int32.MaxValue;
int width = 0;
int height = 0;

foreach (Frame frame in this.frames)//Iterate all frames
{
        if ((frame.offset.X < x_min)) //Determine smallest
           x-offset
        {
                x_min = frame.offset.X;
        }

        if ((frame.offset.Y < y_min))  //Determine smallest
           y-offset
        {
                y_min = frame.offset.Y;
        }

        if (((frame.offset.X + frame.roi.Width) >
           width))//Determine biggest x-offset
                {
                        width = frame.offset.X +
                           frame.roi.Width;
        }
        if (((frame.offset.Y + frame.roi.Height) > height))
           //Determine biggest y-offset
        {
                height = frame.offset.Y + frame.roi.Height;
        }

}

width -= x_min; //Determine collective width
height -= y_min; //Determine collective height

foreach (Frame frame in frames)
{
        frame.offset.X -= x_min; //Remove total x-offset
        frame.offset.Y -= y_min; //Remove total y-offset
}

return (new Size(width, height)); //Return the size
   collectively spanned by frames
```

DrawFrames The *DrawFrames* method creates a composite image of the frames contained in the cluster. A pixel's intensity in the composite image is equal to the average of the various frame intensities at that point.

When the frames are to be drawn, the first task is to realign the them and to determine the size of the composite image using the *RealignFrames* method.

Thereafter, the frames are iterated and accumulated at their respective positions. There will be areas of the composite image that has more frames accumulated than others. After the accumulation, the composite image is divided by a map that indicates how much data is accumulated at each pixel. The *divider_map* is constructed by accumulating the mask of the frames in a similar manner as the data is accumulated. Areas of the cluster that contain more accumulated data would thus consist of bright regions on the *divider_map*.

The listing below demonstrates the accumulation of the data for the *composite*, as well as the accumulation for the *divider_map*. Once the images have been constructed, the *composite* is divided by the *divider_map*. Note that the depth used for the images is float, not byte, since the accumulated values are expected to exceed the 255 range limit for bytes.

```
foreach(Frame frame in frames){
        Rectangle drawing_region =
        new Rectangle(frame.offset.X,frame.offset.Y,
        frame.feature_image.Width,frame.feature_image.Height);

        matching_region_composite =
           composite_float.GetSubRect(drawing_region);
        matching_region_divider =
           divider.GetSubRect(drawing_region);


//      Image<Gray, Byte> mask_sub =
   mask.GetSubRect(drawing_region);
        Image<Gray, float> mask_sub_float =
           mask_sub.Convert<Gray, float>();


        Image<Gray, float> d = new Image<Gray,
           float>(mask_sub.Size);
        Image<Gray, float> divtmp =
           matching_region_divider.Add(d); //Was  new Gray(1)

         mask_sub._Or(frame.mask);

        img = frame.feature_image.Convert<Gray,float>();

        Image<Gray, float> tmp =
           matching_region.AddWeighted(img, 1, 1, 0);

        tmp.CopyTo(matching_region);
        divtmp.CopyTo(matching_region_divider);

}


 Image<Gray,float> composite_float_divided =
    composite_float.Mul(1/divider);



 composite = composite_float_divided.Convert<Gray, Byte>();


}
```

AddCluster The *AddCluster* method is used to combine two clusters after their relative offset from one another has been determined. The *frames* of one of the cluster is removed, the relative offset added to them and then they are added to the other cluster.

After the frames are combined in one *Cluster*, a new *composite* image for the is created.

CalculateHogDescriptor A discussion on the calculation and use of the hog descriptor, is given in the text. The first step is the construction of the kernels for a predefined number of orientations and the known blood vessel width. These kernels only have to be constructed once not and for every descriptor calculated. The listing below demonstrates this construction.

```
int kernel_width = bloodvessel_width * 3;

Image<Gray, float> line = new Image<Gray,
   float>(kernel_width * 2, kernel_width * 2);

line.SetValue(-1);

Point point_1 = new Point(kernel_width,0);
Point point_2 = new Point(kernel_width,kernel_width*2);

line.Draw(new LineSegment2D(point_1, point_2), new Gray(1),
   bloodvessel_width); //Draw a line segment that serves as
   a blood vessel representation

double angle_increment = 180 / directions; //Rotation
   increment between consecutive kernels

Image<Gray, float>[] kernels  = new Image<Gray,
   float>[directions];//Create an array to store kernels

for (int i = 0; i < directions; i++)
{
        Image<Gray, float> kernel = new Image<Gray,
           float>(kernel_width, kernel_width);//Create new
           kernel image
        kernel = line.GetSubRect(new Rectangle(c.X -
           kernel_width / 2, c.Y - kernel_width / 2,
           kernel_width, kernel_width)); //Select sub-region
           from the line segment to serve as kernel
        double average = kernel.GetAverage().Intensity;
           //Calculates average intensity
        kernels[i] = k1;//Adds kernel to array
        line = line.Rotate(angle_increment, new
           Gray(0));//rotates line section with a increment
}
```

Once the kernels are in place, they are employed to determine the amount of features that are orientated in the various directions. The listing below demonstrates the implementation. Here an image is correlated with a specific kernel and the sum of the resulting intensity map is noted. This sum is considered to be the weight of the features that share the same orientation as the kernel. This process is repeated to calculate the weight in all directions an the result is noted in an array of integers.

```


for (int i = 0; i < directions; i++)
{

Image<Gray, float> composite_float = composite.Convert<Gray,
   float>();

Image<Gray, float> correlation =
   composite_float.MatchTemplate(k1,
   TM_TYPE.CV_TM_CCORR);//Correlation image with kernel

Image<Gray, float> mask = correlation.ThresholdBinary(new
   Gray(0), new Gray(1)); //Create mask representing
   positive values

correlation._Mul(mask); //Reject negative values

hog_descriptor[i] = correlation.GetSum().Intensity; //Enter
   weight into histogram
}
```

MatchHogDescriptors The matching of descriptors is conducted in a simple method by calculating the total square difference between the weights of the different orientations. The smaller the distance; the better the match. This distance is returned by the methods and can be used as a relative score.

MatchCluster *Cluster* matching is implemented in a manner similar to the method described for *Frame* matching. The only difference is that the clusters are correlated over a larger region since there is no initial estimate to where two random clusters overlap. The *composite* images of the two *Cluster* objects are used for the matching process.

Appendix D

Physical Setup

To digitise the analog ophthlamoscope, an image sensor has to be attached to it. Various solutions exist, for instance, the use of a webcam or DSLR camera, however, a mobile phone takes preference. The use of a mobile phone and its onboard camera, is proposed. A mobile phone is selected over any other device containing an image sensor for the following reasons:

- The ubiquitous use of mobile phones.
- The interactive capabilities of the device. The user can interact with the data as it is being captured.
- The ease of writing custom software. The custom software can guide the examiner during the capturing process or process the data before transmitting it.
- The direct visual feedback on the screen.
- The telemedicine capabilities of sending the data directly from the device.
- A mobile device is lightweight and does not interfere with the examinations.

Figure D.1 shows the prototype used for the study. It consists of a mobile phone attached to an ophthalmoscope. The phone is placed in the same position as the examiners eye would be. Any light that would normally be observed by the examiner is now captured by the device's camera and displayed on the screen.

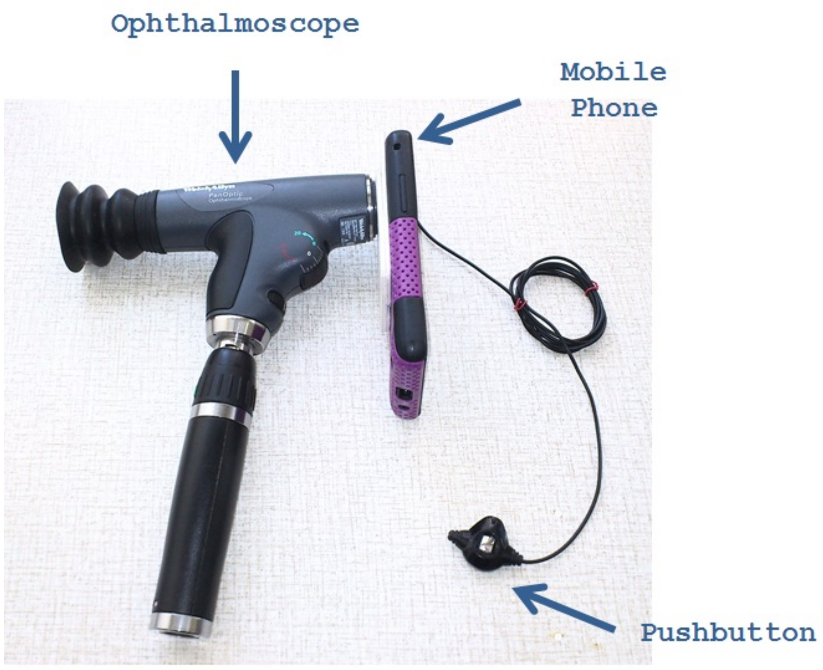


Figure D.1: The prototype used to conduct the research.

A trigger button connected to the phone's audio jack, allows the examiner to only record sections of the process. The button eliminates the need to interact with the device using the touch screen, as this can hinder the precise positioning of the scope to achieve a clear view of the fundus. The examiner would thus hold the scope in one hand while controlling the recording with the other.

The ophthalmoscope used, is the Welch Allyn Pan Optic. It is a hand-held, analog, battery operated ophthalmoscope that offers an increased FOV of 25° over the 5° typical for smaller handheld ophthalmoscopes. The battery operated feature means it is portable and ideal for use in the field while the increased field of view yields better visuals of the fundus for a slight increase in price.

Appendix E

Android Application Proof of Concept

A proof of concept was developed to illustrate the possibility of processing the images directly on the mobile device. The OpenCV build for Android was utilised and some of the basic functionality of the algorithms discussed in the text, were implemented.

Due to the limited resources available on the device, certain compromises were made. Firstly the videos were recorded at a very low frame rate, to reduce the data to be parsed. And secondly, there was more reliance placed on the user to guide the process.

To classify the frames, they are sorted based on the amount of content they contain in the desired colour range and the user prompted to select the best from the collection. Shown in figure E.1 are the frames as sorted by the application. The frames with the white borders are those selected by the user.

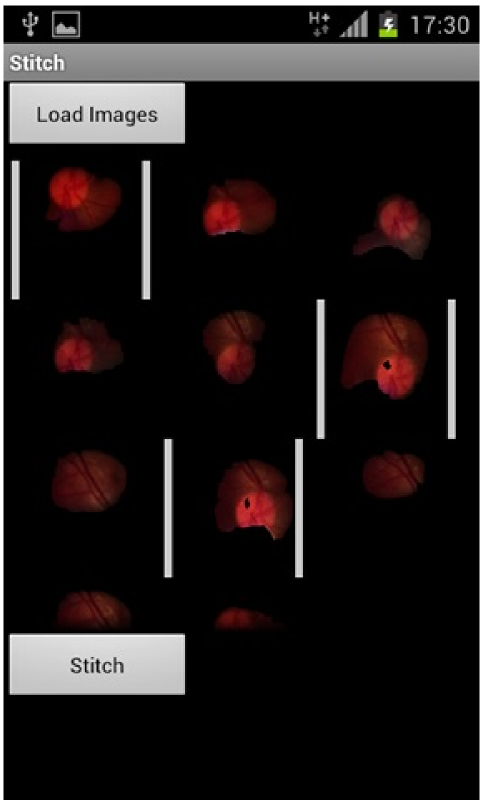


Figure E.1: A selection of frames presented to the user.

Once the desired frames are selected, the user instructs the application to stitch them together. To register the frames exhaustive cross-correlation matching is performed between them to find their relative offsets. This exhaustive search is not scalable, but proved sufficient for the proof of concept. In this example only five frames were used. Shown below, in figure E.2, are the combined frames.

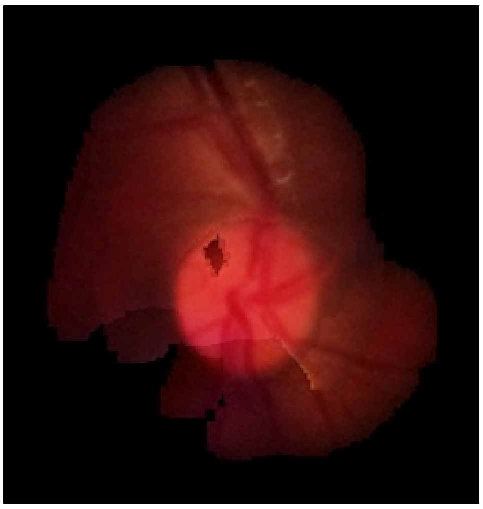


Figure E.2: Retinal images as combined on the device.

It was found that the resources were to limited to implement and execute the designed algorithms in full. Problems arose with the speed of parsing the video and the large amount of memory required for image processing in general. Also, the OpenCV build for Android is relatively new and lacks a lot of the functionality offered by the full build. Even though these problems exist, a full implementation of the algorithms could be realised on a mobile device in the near future. The capabilities of mobile devices are also increasing rapidly and the OpenCV build for Android is actively being developed.

Appendix F

Proposed Workflow

The proposed workflow utilises the physical setup, as demonstrated in Appendix D, and two software packages to implement the algorithms discussed throughout the text. The first software package is an application that runs on the mobile device that captures a video from the ophthalmoscope. The second package runs on a PC and conducts the processing the captured data. The following two paragraphs discuss the use of the Android capture application and the processing of the captured videos on a PC.

F.1 Android Capture Application

The Android application is a custom application that allows the examiner to create a record for each session conducted with a patient. The launch screen is shown in figure F.1. Here, the examiner is prompted for an optional identifier for the patient. A record for the patient along with a timestamp of the session is stored in an XML sheet on the device. An example of an XML sheet containing session records, is shown in Appendix G.

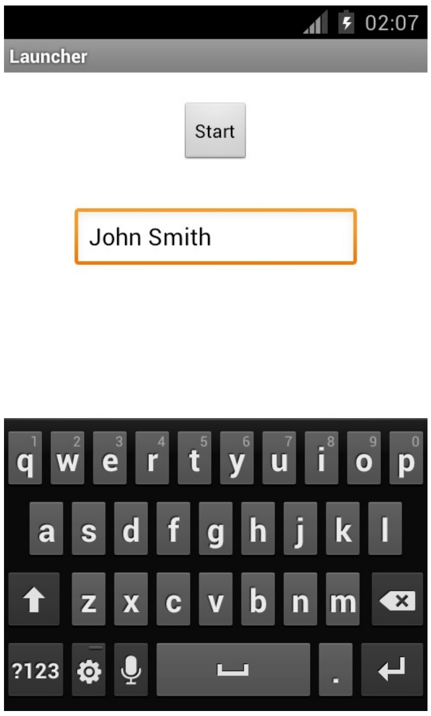
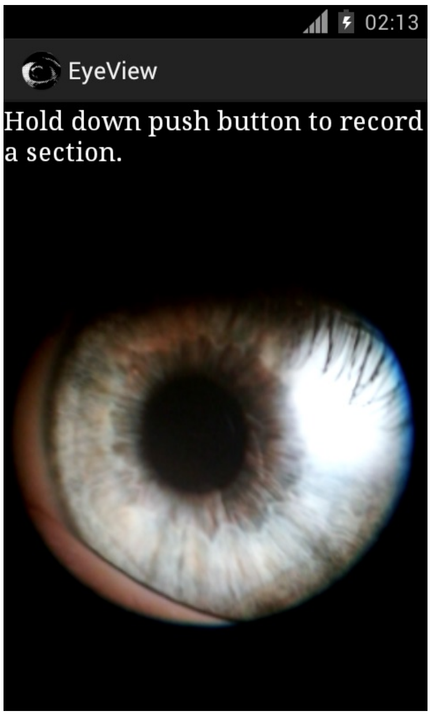


Figure F.1: Patient identifier prompt screen.

Once the identifier is entered, the start button is clicked. A live feed from the camera is shown on the screen of the mobile device. Figure F.2 shows a screenshot of the video feed and further operator instructions.



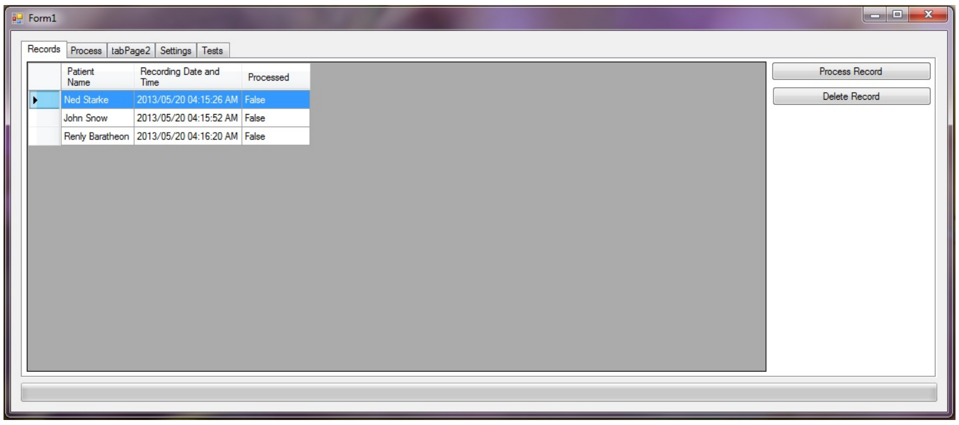
Hold down push button to record a section.

Figure F.2: Video feed on Mobile Device.

The examiner can now use the video feed to adjust the focus of the ophthalmoscope and conduct the examination as usual. A push button is provide that allows the examiner to record sections of the examination is provided. The examiner uses the button to start recording when a clear view of the fundus is obtained. The push button offers an unobtrusive manner for the examiner to do the first level of classification of the data obtained. The alternative is to record the entire session and have a much longer video to parse. The added push button thus significantly reduces the amount of data to be classified by the software algorithms and greatly increases the performance of the system. As many sections as desired can be recorded in a session with all the captured data saved to the device's memory as segmented video files. Once the examination is complete, the application is be exited. At this stage, a new examination can be started or the ones already conducted, can be processed on a PC.

F.2 Processing the Data

The data can either be processed directly by the examiner or at a later stage by another user. To start, the mobile device is connected to a PC via a USB cable. The software is started and all the sessions recorded with the mobile device are displayed in the startup screen as shown in figure F.3. The information displayed here is derived from the XML records stored on the device. A session is selected to be processed. At this stage, the relevant video files are copied from the device to the PC to start processing.



Patient Name	Recording Date and Time	Processed
Ned Starke	2013/05/20 04:15:26 AM	False
John Snow	2013/05/20 04:15:52 AM	False
Renly Baratheon	2013/05/20 04:16:20 AM	False

Figure F.3: A list of records to be processed is displayed to the user.

The video files are all parsed and their frames are sorted into clusters. The videos are processed at a reduced resolution to increase the speed of the processing. Larger images require more time to be copied into memory and the correlation operations get slower as the size increases. Another method to conserve memory, is not to retain the frame content in memory. This would quickly result in the application running out of memory. Instead pointers to the specific frames in the relevant videos are retained. The only data retained in primary memory, is the composite images of each cluster. After the clusters have been created, they are displayed to the user as shown in figure F.4.

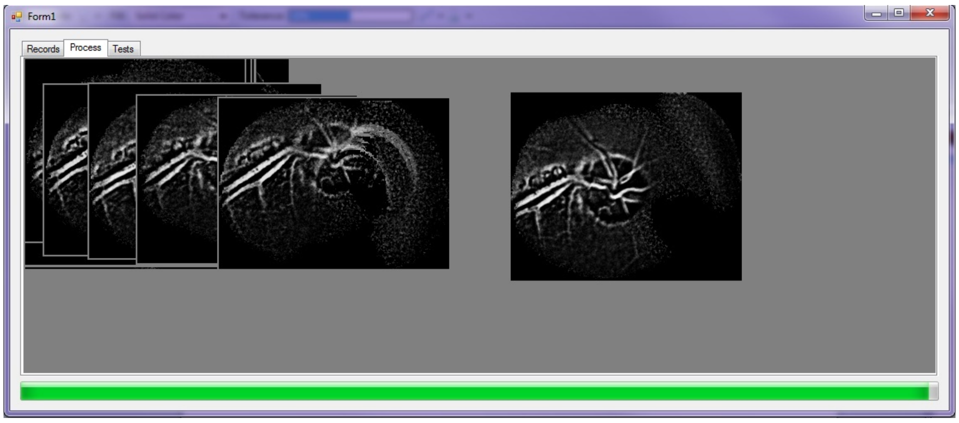


Figure F.4: The initial clusters constructed by parsing the video files.

At this stage the user can manually sort and join the clusters or initiate the automatic grouping. To join the clusters manually the user simply holds down the '+' character and drag a cluster over another. The best offset estimate between the clusters is calculated and the frames of the two clusters are joined. A new composite image for the cluster is calculated.

When grouping the clusters automatically, the matching scheme (discussed in the text) is used to sort the bulk of the clusters. The result is several large clusters and numerous low quality clusters that could not be joined as shown in figure F.5.

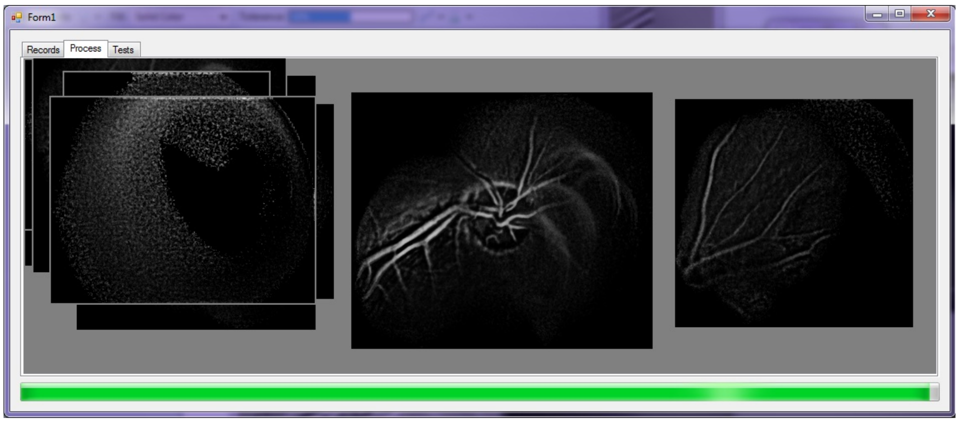


Figure F.5: Clusters constructed by joining several smaller ones.

The user can also manually join any remaining clusters missed by the sorting algorithm.

The last step is the investigation of the clusters. The user double clicks a cluster to bring up the cluster scaled to the original capture resolution. Here the user can select a section of the cluster to be reconstructed. The frames constituting the selected section are fetched from the video files and recorrelated to form a high-resolution estimate. The new estimate consists of numerous frames and contains more detail than an individual frame.

Appendix G

XML Record Storage

The examination records are stored in a XML file on the mobile device. The listing below shows a typical XML sheet containing three session records. Each record contains four fields. The first is an identifier unique to each record. This identifier is used to indicate a unique folder that contains the video files specific to the record. The second is the name associated with the record. The name does not have to be unique. This allows for multiple records for a specific patient. The third property is a timestamp. The timestamp indicates the time of the examination and allows for the monitoring of the progression of a condition. The last field indicates whether a record has been processed. This is set after a record has been processed on a PC.

```xml
<?xml version='1.0' encoding='UTF-8'
   standalone='yes' ?>
<records>

<record>
        <id>fc952342-6516-4cd9-a1fb-29eb33d10261</id>
        <name>Renly Baratheon</name>
        <timestamp>2013-01-01T03:49:05</timestamp>
        <processed>false</processed>
</record>

<record>
        <id>458956d7-ab5f-4678-8149-aeeea5c59f7e</id>
        <name>John Snow</name>
        <timestamp>2013-01-01T03:49:49</timestamp>
        <processed>false</processed>
</record>

<record>
        <id>a94e1b38-2a0c-436e-8f14-a4505df537e7</id>
        <name>Ned Starke</name>
```

```
                <timestamp>2013-01-01T04:02:19</timestamp>
                <processed>false</processed>
        </record>

        </records>
```

Appendix H

Ideal Cross-Correlation Peak Shape

The shape of the cross-correlation peak, yields information on the data being correlated. Typically an elongated peak indicates the predominant orientation of the features correlated. This attribute is suggested for use in flow estimation of particles, where a long-exposure rate causes a blur in the direction of the flow. The flow direction is then extracted from the cross correlation of consecutive frames [24].

In this application, the elongated peak is not desired since it indicates an uncertainty to the exact location of the peak. Much similar to the flow of particles, an elongated peak indicates the flow of features, an undesired observation.

As mentioned in the text, the profile of a blood vessel is often estimated with a Gaussian curve. We use this curve to prove certain concepts and attributes of the cross-correlation peak. A two-dimensional Gaussian curve is defined in equation H.1 that offers an elliptical Gaussian distribution as shown in figure H.1. The shape of the curve is specified by the variance in the x- and y-direction (σ_x^2 and σ_y^2 respectively).

$$f(x,y) = Ae^{-\left(\left(\frac{(x-x_0)^2}{2\sigma_x^2}\right)+\left(\frac{(y-y_0)^2}{2\sigma_y^2}\right)\right)} \tag{H.1}$$

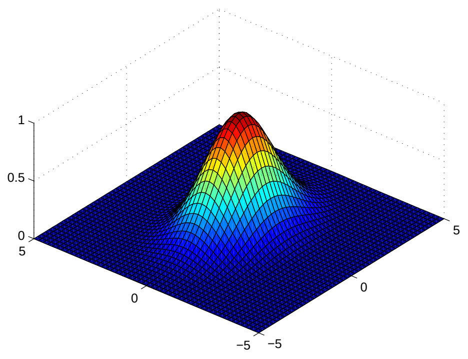


Figure H.1: 2D Gaussian curve.

To resemble the appearance of a blood vessel, variance in the y-direction (σ_y^2) is set the to infinity. For simplicity the x-offset (x_0) is set to zero. Equation H.2 demonstrates the result while figure H.2 shows the resulting curve.

$$f(x,y) = Ae^{\frac{-x^2}{2\sigma_x^2}} \tag{H.2}$$

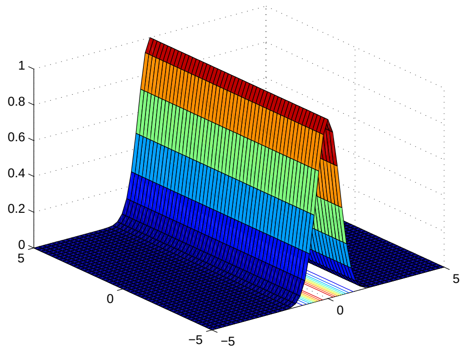


Figure H.2: Elongated 2D Gaussian curve.

To demonstrate the effect of cross correlation of blood vessel the auto-correlation for equation H.2 can calculated along the x-axis. The y-component is ignored since the function is independent of y. Two identical blood vessels are assumed and thus calculating the auto correlation of the blood-vessel profile is equivalent to calculating the cross correlation.

Continuous cross correlation is given by equation H.3.

$$(f \bigstar g)(x) = \int f(\tau)g(x+\tau)d\tau \tag{H.3}$$

The function in equation H.2 is substituted into equation H.3 as given by equation H.4.

$$(f\bigstar f)(x) = \int_{-\infty}^{\infty} Ae^{\frac{-\tau^2}{2\sigma_x^2}} Ae^{\frac{-(x+\tau)^2}{2\sigma_x^2}}\, d\tau \tag{H.4}$$

The simplification of the correlation function is given by the following equation (equation H.5).

$$\begin{aligned}\int_{-\infty}^{\infty} Ae^{\frac{-\tau^2}{2\sigma_x^2}} Ae^{\frac{-(x+\tau)^2}{2\sigma_x^2}}\, d\tau &= \int_{-\infty}^{\infty} A^2 e^{\frac{-\tau^2-(x^2+2x\tau+\tau^2)}{2\sigma_x^2}}\, d\tau \\ &= A^2 e^{-\frac{x^2}{2\sigma_x^2}} \int_{-\infty}^{\infty} e^{\frac{-2\tau^2}{2\sigma_x^2}} e^{\frac{-2x\tau}{2\sigma_x^2}}\, d\tau\end{aligned}$$

Note that,

$$\int_{-\infty}^{\infty} e^{-a\tau^2} e^{-2b\tau} d\tau = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{a}}$$

therefore,

$$\begin{aligned} a &= \frac{1}{\sigma_x^2} \\ b &= \frac{x}{2\sigma_x^2}\end{aligned}$$

and thus,

$$(f\bigstar f)(x) = \sigma_x \sqrt{\pi} A^2 e^{\frac{-x^2}{4\sigma_x^2}} \tag{H.5}$$

An important attribute of the resulting function in equation H.5 is that it yields a Gaussian curve with a higher peak and a larger variance. In figure H.3 the resulting correlation curve is shown.

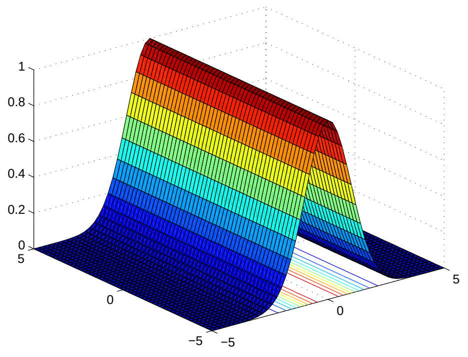


Figure H.3: Auto correlation of elongated Gaussian curve.

Note that the peak of the cross correlation is simply a ridge and that the y-offset between the two functions can not be determined. In a similar manner to the example above, the cross correlation for identical blood vessels positioned along the x-axis, can be determined.

A useful property of the cross-correlation operation, is that it holds valid against the distributive law. Equation H.6 demonstrates this law.

$$f\bigstar(g+h) = f\bigstar g + f\bigstar h \tag{H.6}$$

Using the distributive law, the cross correlation for an image containing features along the x-axis as well as the y-axis, can be calculated by summing the two individual results. The resulting cross correlation is shown in figure H.4. Note the strong and easily locatable peak. The ‘flow ’of the peak is significantly reduced.

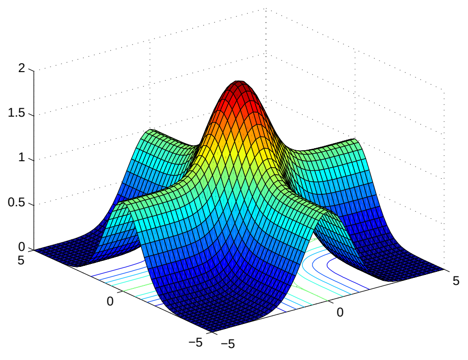


Figure H.4: Combination of two elongated Gaussian curves at different orientations.

From this result, we deduce that the wider the range of orientations in the features that are to be correlated, the more certain the cross-correlation peak is.

The ideal solution would be to have features in all possible directions. This is simulated by integrating the correlation result over a 180° rotation. The integration yields the sum of the cross correlation in all directions. The calculation of this sum is allowed by the distributive law.

A variation of the Gaussian function that allows for the specification of the orientation of the curve is given in equation H.7. The orientation is specified by the angle θ.

$$f(x,y) = Ae^{-(a(x-x_0)^2)+2b(x-x_0)(y-y_0)+c(y-y_0)^2)}$$

where:

$$\begin{aligned} a &= \frac{\cos^2\theta}{2\sigma_x^2} + \frac{\sin\theta}{2\sigma_y^2} \\ b &= \frac{\sin 2\theta}{4\sigma_x^2} + \frac{\sin 2\theta}{4\sigma_y^2} \\ c &= \frac{\sin^2\theta}{2\sigma_x^2} + \frac{\cos^2\theta}{2\sigma_y^2} \end{aligned} \tag{H.7}$$

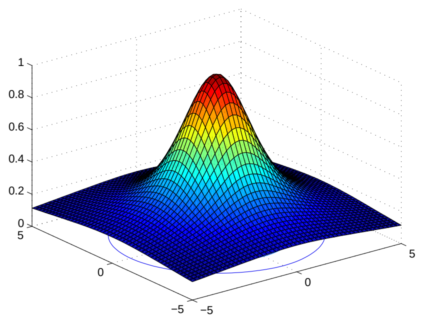
Again, the variance in the y-direction is set to infinity. The variance is set to the variance obtained in equation H.5 such that $\sigma_x^{2\prime} = 2\sigma_x^2$. The offsets, $x0$ and $y0$, are set to zero. The amplitude is specified as A', where $A' = \sigma_x\sqrt{\pi}A^2$, the amplitude calculated in equation H.5. The resulting integral is given by equation H.8.

$$\int_0^\pi A'e^{-(x^2\frac{\cos^2\theta}{2\sigma_x^{2\prime}}+2xy\frac{\sin\theta\cos\theta}{2\sigma_x^{2\prime}}+y^2\frac{\sin^2\theta}{2\sigma_x^{2\prime}})}d\theta$$

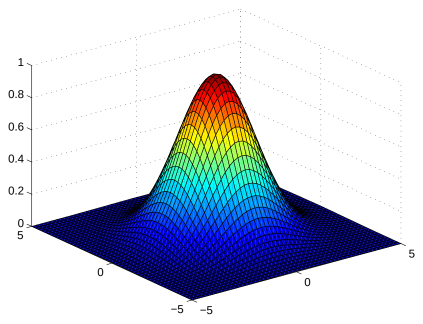
which simplifies to:

$$\int_0^\pi A'e^{-\frac{1}{2\sigma_x^{2\prime}}(x\cos\theta+y\sin\theta)^2}d\theta \tag{H.8}$$

Equation H.8 is solved using numerical methods with the result shown in figure H.5a. The resulting curve resembles a modulated radially symmetrical Gaussian curve. If a Gaussian curve is fitted to the function it is found to have variance three times that of the original blood vessel in equation H.2. Figure H.5b shows a Gaussian distribution with a matching variance of $3\sigma_x^2$.



(a) Numerically integrated function



(b) Matching Gaussian curve

The deduction can be made that the ideal cross correlation peak for a scene containing blood vessels with a Gaussian profile with a variance of σ_x^2, is a function resembling a radially symmetrical Gaussian distribution with a variance of $3\sigma_x^2$.

Appendix I

Cross-Correlation Execution Time Derivation

When matching two images using cross correlation, the template image is slid over the reference image and the sum of the multiplication of all overlapping pixels, are noted at each position.

If the reference image and the template is the same size, the reference image has to be padded to allow the sliding of the template. Figure I.1 notes the two images with both dimensions $N \times N$ and the reference image padded by $P \times P$ pixels.

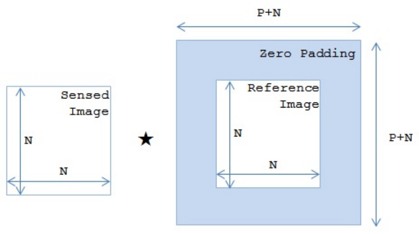


Figure I.1: Parameters of the sensed and reference images.

Adapting the cross-correlation equation given in equation A.3, the two finite signals yields equation I.1 where $g[x, y]$ is the reference image and $f[x, y]$ is the template image. Figure I.2 demonstrates this process.

$$f[x,y]\bigstar g[x,y] = \sum_{p1=0}^{P}\sum_{p2=0}^{P}\sum_{n1=0}^{N}\sum_{n_2=0}^{N} f[n_1,n_2].g[x+n_1+p1, y+n_2+p2] \quad (I.1)$$

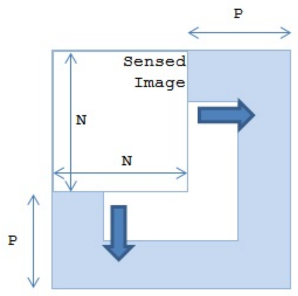


Figure I.2: Sensed image being slid over reference image.

At each position that the template image assumes, all its pixel values are multiplied with the corresponding reference image pixels. This operation would require $N \times N$ individual multiplications. The total amount of possible positions that the template image can assume, is equivalent to the padding around the reference image. For a padding of P, $P \times P$ positions can be assumed.

Since the number of multiplications at each position is equal to $N \times N$ and $P \times P$ positions can be assumed, the total number of individual multiplication operations required to calculate the cross correlation between two images, can be described by equation I.2.

$$multiplications = P^2 \times N^2 \tag{I.2}$$

List of References

[1] UR Acharya, CM Lim, EYK Ng, C Chee, and T Tamura. Computer-based detection of diabetes retinopathy stages using digital fundus images. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 223(5):545–553, 2009.

[2] Welch Allyn. Welch allyn panoptic ophthalmoscope user guide. `http://www.welchallyn.com/documents/EENT/Ophthalmoscopes/PanOptic/SM3001EU%20RevB%20072208%20PanOptic%20Ref%20Card.pdf`, 2007.

[3] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[4] Beau B Bruce, Cédric Lamirel, Valérie Biousse, Antionette Ward, Katherine L Heilpern, Nancy J Newman, and David W Wright. Feasibility of nonmydriatic ocular fundus photography in the emergency department: Phase i of the foto-ed study. *Academic Emergency Medicine*, 18(9):928–933, 2011.

[5] Philippe C Cattin, Herbert Bay, Luc Van Gool, and Gábor Székely. Retina mosaicing using local features. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006*, pages 185–192. Springer, 2006.

[6] Thitiporn Chanwimaluang, Guoliang Fan, and Stephen R Fransen. Hybrid retinal image registration. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):129–142, 2006.

[7] Subhasis Chaudhuri, Shankar Chatterjee, Norman Katz, Mark Nelson, and Michael Goldbaum. Detection of blood vessels in retinal images using two-dimensional matched filters. *IEEE Transactions on medical imaging*, 8(3):263–269, 1989.

[8] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. 1:886–893, 2005.

[9] E De Castro and C Morandi. Registration of translated and rotated images using finite fourier transforms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (5):700–703, 1987.

[10] Disabilities Department of Health Directorate: Chronic Diseases and Geriatrics. Prevention of blindness in south africa. http://www.kznhealth.gov.za/blindness.pdf, 2002.

[11] Manjusha Deshmukh and Udhav Bhosle. A survey of image registration. *International Journal of Image Processing (IJIP)*, 5(3):245, 2011.

[12] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[13] Centre for Vision Research. Detecting the risk of stroke using retinal imaging. http://sydney.edu.au/news/84.html?newsstoryid=715, 2005.

[14] Yann Gavet, Mathieu Fernandes, and Jean-Charles Pinoli. Quantitative evaluation of image registration techniques in the case of retinal images. *Journal of Electronic Imaging*, 21(2):021118–1, 2012.

[15] Sara Gharabaghi, Sabalan Daneshvar, and Mohammad Hossein Sedaaghi. Retinal image registration using geometrical features. *Journal of Digital Imaging*, pages 1–11, 2012.

[16] Sara Gharabaghi, Sabalan Daneshvar, and Mohammad Hossein Sedaaghi. Retinal image registration using geometrical features. *Journal of Digital Imaging*, pages 1–11, 2012.

[17] Google. Googlemaps. http://maps.google.com/?ll=-33.928424,18.865972&spn=0.005047,0.010568&t=h&z=17, 2013.

[18] A.A. Goshtasby. *2-D and 3-D Image Registration: for Medical, Remote Sensing, and Industrial Applications*. Wiley, 2005.

[19] Mikael Häggström. Fundus photograph of a normal left eye. http://en.wikipedia.org/wiki/File:Fundus_photograph_of_normal_left_eye.jpg, 2012.

[20] Chris Harris and Mike Stephens. A combined corner and edge detector. 15:50, 1988.

[21] Adam Hoover and Michael Goldbaum. Locating the optic nerve in a retinal image using the fuzzy convergence of the blood vessels. *Medical Imaging, IEEE Transactions on*, 22(8):951–958, 2003.

[22] Frank Horst. Rgb cube show. http://upload.wikimedia.org/wikipedia/commons/thumb/8/83/RGB_Cube_Show_lowgamma_cutout_b.png/800px-RGB_Cube_Show_lowgamma_cutout_b.png, 2010.

[23] Edwars S. Harkness Eye Institute. Retinal vascular diseases. http://dro.hs.columbia.edu/vr3.htm, 2013.

[24] KD Jensen. Flow measurements. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 26(4):400–419, 2004.

[25] Karin Kneöaurek, Marija Ivanovic, Josef Machac, and David A Weber. Medical image registration. *Europhysics News*, 31(4):5–8, 2000.

[26] V Vijaya Kumari and N Suriyanarayanan. Blood vessel extraction using wiener filter and morphological operation. *Int. J. Comput. Sci. Emerg. Technol*, 1(4):7–10, 2010.

[27] Sangyeol Lee, Joseph M Reinhardt, Philippe C Cattin, and Michael D Abràmoff. Objective and expert-independent validation of retinal image registration algorithms by a projective imaging distortion model. *Medical image analysis*, 14(4):539–549, 2010.

[28] Hao Li, Hansheng Yang, Guohua Shi, and Yudong Zhang. Adaptive optics retinal image registration from scale-invariant feature transform. *Optik-International Journal for Light and Electron Optics*, 122(9):839–841, 2011.

[29] Mai S Mabrouk, Nahed H Solouma, and Yasser M Kadah. Survey of retinal image segmentation and registration. *Graph. Vis. Image Process. J*, 6:1–11, 2006.

[30] Jagadish Nayak, P Subbanna Bhat, Rajendra Acharya, CM Lim, and Manjunath Kagathi. Automated identification of diabetic retinopathy stages using digital fundus images. *Journal of medical systems*, 32(2):107–115, 2008.

[31] Optomed Oy. Smartscope m5. `http://www.optomed.com/smartscope+m5+/`, 2013.

[32] C.W. Oyster. *The Human Eye: Structure And Function*. Sinauer Associates Incorporated, 1999.

[33] Eli Peli, Reed A Augliere, and George T Timberlake. Feature-based registration of retinal images. *Medical Imaging, IEEE Transactions on*, 6(3):272–278, 1987.

[34] Douglas A Perednia and Ace Allen. Telemedicine technology and clinical applications. *JAMA: the journal of the American Medical Association*, 273(6):483–488, 1995.

[35] Jacob Rus. Hsl-hsv models. `http://en.wikipedia.org/wiki/File:Hsl-hsv_models.svg`, 2010.

[36] Nirmala Saini and Aloka Sinha. Optics based biometric encryption using log polar transform. *Optics Communications*, 283(1):34–43, 2010.

[37] PC Siddalingaswamy and K Gopalakrishna Prabhu. Automatic detection of multiple oriented blood vessels in retinal images. *Journal of Biomedical Science and Engineering*, 3(1):101–107, 2010.

[38] GigaPan Systems. Gigapan. `http://www.gigapan.com`, 2013.

[39] Marko Tkalcic and Jurij F Tasic. *Colour spaces: perceptual, historical and applicational background*, volume 1. IEEE, 2003.

[40] Health System Trust. South african health review. `http://www.hst.org.za/sites/default/files/sahr2008.pdf`, 2008.

[41] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[42] Stefan Johann Van der Walt. Super-resolution imaging. *P h D Dissertation, Stellenbosch University*, 2010.

[43] Christo Carel Van Der Westhuizen. Fundus examination. `http://www.youtube.com/watch?v=Sob3-ar70x8`, 2013.

[44] Vodacom. Vodacom coverage map. `http://www.vodacom.co.za/personal/main/coveragemaps`, 2013.

[45] Wikipedia. Human eye. `http://en.wikipedia.org/wiki/Human_eye`, 2013.

[46] Wikipedia. Macular edema. `http://en.wikipedia.org/wiki/Macular_edema`, 2013.

[47] Yanxiong Wu and Xiling Luo. A robust method for airborne video registration using prediction model. In *Computer Science and Information Technology, 2008. ICCSIT'08. International Conference on*, pages 518–523. IEEE, 2008.

[48] Lili Xu and Shuqian Luo. A novel method for blood vessel detection from retinal images. *Biomedical engineering online*, 9(1):14, 2010.

[49] Mohamed S Yasein and Pan Agathoklis. A robust, feature-based algorithm for aerial image registration. In *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pages 1731–1736. IEEE, 2007.

[50] Wong Li Yun, U Rajendra Acharya, YV Venkatesh, Caroline Chee, Lim Choo Min, and EYK Ng. Identification of different stages of diabetic retinopathy using retinal optical images. *Information sciences*, 178(1):106–121, 2008.

[51] Jian Zheng, Jie Tian, Yakang Dai, Kexin Deng, and Jian Chen. Retinal image registration based on salient feature regions. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 102–105. IEEE, 2009.

[52] Qinfen Zheng and Rama Chellappa. Automatic registration of oblique aerial images. In *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, volume 1, pages 218–222. IEEE, 1994.

[53] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003.