

# University of Kasdi Merbah - Ouargla

Faculty of New Technologies of Information and Communication  
Department of Electronic and Telecommunication



## FINAL STUDY DISSERTATION

In the aim of obtaining MASTER Degree - ACADEMIC

Domain: Science and Technology

Option: Automatic and Systems

Specialty: Automatic

Presented by:

*Haithem MEKHERMECHE*

*Bilal CHEBOUAT*

Topic:

**Points Based Features for Image Mosaicing and Template  
Object Tracking Algorithms**

Was Publicly Debated on: 25.06.2018 in front of The Examining Committee

Composed From:

Dr. F. CHARIF	MCB	President	UKM Ouargla
Mrs. W. BENZAOUI	MAA	Examiner	UKM Ouargla
Mr. B. BENHELLAL	MAA	Supervisor	UKM Ouargla
Mr. A. LATI	Doctorant	Co-Supervisor	USTHB Algiers

*Academic year : 2017 /2018*

# Acknowledgments

---

First of all, we thank **ALLAH** for having given us the courage and patience to arrive at this modest work.

We would like to thank our supervisor, Mr. BENHELLAL Belkhair, who has been always generous during all phases of the research, and co-supervisor Mr. LATI Abdelhai for his supervision, his follow-up and advice throughout this period.

Our most grateful thanks to all those who have contributed to the realization of this memory.

Finally, we would like to express our tremendous gratitude to our family and our friends for their support during these years.

# *DEDICATION*

---

*We dedicate this modest work to those who are the source of our inspiration and our courage.*

*To my dear mother, who always gives me hope to live and who has never stopped praying for me.*

*To my dear father, for his encouragement and support, And above all for his sacrifice so that nothing will hinder The course of my studies.*

*To all the professors and teachers who have followed me throughout my schooling and who have allowed me to succeed in my studies.*

*To my dear brothers*

*To my dear sisters*

*All my friends*

*Haithem MEKHERMECHE*

*Bilal CHEBOAT*

# Contents

List of Figures	iv
List of Tables	vi
List of Abbreviations	vii
General Introduction	1
<b>I State of the Art of Image Features Extraction</b>	<b>3</b>
I.1 Introduction	3
I.2 Image sources	4
I.3 Features extraction	4
I.4 Classification of features	5
I.4.1 General features	5
I.4.2 Domain-specific features (Robotic Vision Domain):	7
I.5 Conclusion	9
<b>II Points-Based Features Detectors</b>	<b>10</b>
II.1 Introduction	10
II.2 Points-based features detectors (State of the Art)	10
II.3 Classical detectors	11
II.3.1 FAST detector	11
II.3.2 Harris detector	12
II.4 Modern detectors	14
II.4.1 SIFT detector/descriptor	14
II.4.2 SURF detector/descriptor:	16
II.5 Binary Descriptors	18
II.5.1 BRISK descriptor	19
II.5.2 FREAK descriptor	21
II.6 Conclusion	22
<b>III Image Mosaicing and Template Object Tracking</b>	<b>23</b>
III.1 Introduction	23

III.2 Image mosaic issues . . . . .	24
III.2.1 Image registration . . . . .	24
III.2.2 Image Re-projection . . . . .	25
III.2.3 Image Blending . . . . .	26
III.3 Features extraction . . . . .	27
III.3.1 Properties of good features . . . . .	28
III.4 Features matching (Association) . . . . .	28
III.4.1 Correlation based features matching . . . . .	29
III.4.2 SIFT/SURF descriptors based features matching . . . . .	30
III.4.3 LBP Descriptors Based features matching . . . . .	31
III.4.4 BRISK, and FREAK Based features matching . . . . .	33
III.4.5 Features tracking based matching . . . . .	33
III.5 Geometric transformations . . . . .	34
III.5.1 Geometric transformations Models . . . . .	34
III.5.2 Homography estimation . . . . .	39
III.6 Image projection "Warping" . . . . .	41
III.6.1 Forward warping . . . . .	41
III.6.2 Backward warping . . . . .	41
III.7 Template object tracking . . . . .	42
III.8 Conclusion . . . . .	43
<b>IV Results and Evaluation</b>	<b>44</b>
IV.1 Introduction . . . . .	44
IV.2 Work environment . . . . .	44
IV.3 Evaluation and results . . . . .	45
IV.3.1 Image Mosaicing . . . . .	45
IV.3.2 Template object tracking . . . . .	54
IV.4 Conclusion . . . . .	56
<b>General Conclusion</b>	<b>57</b>
<b>Bibliography</b>	<b>59</b>

# *List of Figures*

I.1	Real images from different sources . . . . .	4
I.2	The RGB and HSV colours spaces. . . . .	5
I.3	Eye features extraction on colour image. . . . .	6
I.4	Typical iris recognition stages . . . . .	7
I.5	Localization using line and corner features . . . . .	8
I.6	The trajectories of the two UAVs in the X and Y axes . . . . .	9
II.1	Visualization of the FAST corner detection feature . . . . .	11
II.2	Nature of a point with Harris algorithm . . . . .	12
II.3	Applying corner operator. . . . .	13
II.4	Finding points with large corners response ( $R > \text{threshold}$ ). . . . .	14
II.5	Taking only the points of local maxima of $R$ . . . . .	14
II.6	SIFT detector generation. . . . .	15
II.7	SIFT descriptor generation. . . . .	16
II.8	Functionality of integral image. . . . .	17
II.9	Approximations of Gaussian second order partial derivatives by box filters . . . . .	17
II.10	BRISK sampling pattern . . . . .	19
II.11	Short distance pairs on the left and long distance pairs on the right. . . . .	20
II.12	Receptive fields distribution over the retina in a human eye . . . . .	21
II.13	FREAK sampling pattern. . . . .	21
II.14	FREAK's sampling pairs resulting from Coarse-to-fine analysis . . . . .	22
III.1	Mosaic rendering by re-projection onto a planar manifold . . . . .	25
III.2	Mosaic obtained by re-projection onto a calibrated cylindrical manifold . . . . .	26
III.3	Blurring effect after blending . . . . .	27
III.4	Blending based on distance. . . . .	27
III.5	Index pairs of the matched features. . . . .	31
III.6	The steps to create LBP Feature Descriptor ( $P=8, R=1$ ). . . . .	32
III.7	LBP descriptors for different values of points ( $P$ ) and radius ( $R$ ). . . . .	32
III.8	Two images of a planar scene connected by a rotation and translation. . . . .	39
III.9	Illustrating template matching. . . . .	42
IV.1	Flowchart for the proposed image mosaicing. . . . .	45

IV.2 The inputs images. . . . .	45
IV.3 Features detection using Harris detector. . . . .	46
IV.4 Features matching using correlation. . . . .	47
IV.5 Features matching using BRISK descriptor. . . . .	47
IV.6 Image mosaic by backward warping. . . . .	48
IV.7 Page 1: displaying Features detect in both images. . . . .	49
IV.8 Page 2: displaying the features matched between images. . . . .	49
IV.9 Page 3: displaying the estimated matrix . . . . .	50
IV.10Page 4: displaying the mosaiced image and the summary. . . . .	50
IV.11The method proposed for object tracking flowchart. . . . .	54

# *List of Tables*

III.1 The most known correlation criteria . . . . .	29
III.2 Comparison between affine and projective transformations. . . . .	38
III.3 Illustration of the projective linear group and its three subgroups . . . . .	38
IV.1 The effect of threshold value on the performance of the detector. . . . .	46
IV.2 The effect of the correlation window. . . . .	47
IV.3 Testing on Outdoor/Indoor phone images . . . . .	51
IV.4 Testing on Aerial and satellite images . . . . .	52
IV.5 Testing on remote sensing and medical imaging . . . . .	53
IV.6 Template object tracking results . . . . .	55



# *List of Abbreviations*

MRI	Magnetic Resonance Imaging
RGB	Red, Green, Blue
HSV	Hue, Saturation, Value
CCV	Colour Coherence Vector
CM	Colour Moment
SIFT	Scale-Invariant Feature Transform
DoG	Difference of Gaussians
SURF	Speeded Up Robust Features
FAST	Features from Accelerated Segment Test
CRF	Corner Response Function
SUSAN	Smallest Univalve Segment Assimilating Nucleus
BRIEF	Binary Robust Independent Elementary Features
ORB	Oriented FAST and Rotated BRIEF
FREAK	Fast Retina Keypoint
BRISK	Binary Robust Invariant Scalable Keypoints
FFT	Fast Fourier Transform
BCM	Brightness Constancy Model
SAD	Sum of Absolute Differences
ZSAD	Zero-mean Sum of Absolute Differences
LSAD	Locally scaled Sum of Absolute Differences
SSD	Sum of Squared Differences
ZSSD	Zero-mean Sum of Squared Differences
LSSD	Locally scaled Sum of Squared Differences
NCC	Normalized Cross Correlation
ZNCC	Zero-mean Normalized Cross Correlation

LBP	Local Binary Patterns
LBD	Local Binary Descriptor
KLT	Kanade-Lucas-Tomasi
UAV	Unmanned Aerial Vehicle
DLT	Direct Linear Transform
SVD	Singular Value Decomposition
RANSAC	RANdom SAmples Consensus
DSP	Digital Signal Processing
FPGA	Field-Programmable Gate Array
VI	Virtual Instruments
NI	National Instruments
VHDL	VHSIC Hardware Description Language

# *General Introduction*

The human is capable to see, to recognize, to infer the depth of the real world due to his vision system; that is basically composed of two eyes and the brain. The eyes take two images for the scene simultaneously then a complex and very fast processing is made to these images inside the brain in order to reproduce the real world. People have tried to understand this phenomenon from the early years, so first they thought to produce a camera that plays the role of the eye in order to capture the scene. The invention of digital computers in the late of 60th helped a lot the researchers for saving digital images and performing numerical calculations [1]. However, the problem of reproducing the real world from images is not simple and it requires doing hard calculations and implementing complex algorithms on images [2]. Researchers have been investigating methods to acquire three dimensions (3D) information from objects and scenes for many years.

In most of image processing algorithms, feature detection techniques are used for computing abstractions of image information and making local decisions at every image part whether there is an image feature of a given type at that part or not. The resulting features will be subsets of the image domain, often in the form of isolated points, continuous curves or connected regions, in which; type of features depends on the problem or the type of application [3]. Features are used as a starting point for many computer vision algorithms, since features are used as the starting point and main primitives for subsequent algorithms such as image registration, the overall algorithm will often only be as good as its used feature detector.

Image registration is the process of aligning two or more images, so that objects representing the same structures are eroded. Several branches of science have benefited from the applications of registration, to refine certain objectives. Among these applications, which appear, we find mainly the mosaic of images and object tracking. An Image Mosaic is a synthetic composition generated from a sequence of images and it can be obtained by understanding geometric relationships between images. The geometric relations are the coordinate systems that relate the different image coordinate systems [4]. Constructing image mosaics is an active area of research in the field of computer vision, and image processing. Since many years, image mosaics were used for various applications, and the most traditional application was and still until now; the construction of large aerial and

satellite photographs from collections of overlapped images [5]. Today, there exist more recent modern applications for image mosaicing including; scene stabilization, change detection, video compression, increasing the field of view and resolution of a camera.

Object tracking algorithms are designed to locate (and keep a steady watch on) a moving object (or many moving objects) over time in a video stream, object tracking is an important component of many computer vision systems, and it is widely used in video surveillance, robotics, 3D image reconstruction, medical imaging, and human computer interface.

The performance of the mentioned algorithms depends mainly on features detection and matching, because robust features detector and robust method for features matching ensure robust geometric image registrations. Registering images is based mainly on extracting the overlapping region between them, for large overlapping; correlation measures can be used to find coordinates of that common region, but for small overlapping; features description needs to be performed. In our work; we are going to implement these two image registration methods; and make comparison between them, then, we will choose the best one for implementing image mosaicing and objecting tracking algorithms on LabVIEW platform.

Our work is organized in four chapters as follows; in Chapter I, we start with a state of the art about types of features. The second chapter presents a review of the methods used for points based features detection. In the third chapter, we present the techniques specific to the mosaic of images and object tracking. In the fourth chapter, we will present the design of our algorithms, and describe the different steps and techniques used for their implementation.

# *State of the Art of Image Features Extraction*

---

I.1	Introduction . . . . .	3
I.2	Image sources . . . . .	4
I.3	Features extraction . . . . .	4
I.4	Classification of features . . . . .	5
I.5	Conclusion . . . . .	9

---

## **I.1 Introduction**

In the human vision system, the brain processes images (the scene) derived from the eyes. Similarly, the robot vision system when the computer (robot or machine) processes images which are captured from camera or optical system in general. Nowadays, most of automated industrial are using vision system for many purposes like:

- Manufacturing to check size, quality and present... of the products.
- Telescope images used in astronomy and satellites data analysing.
- Pattern recognition and Biometrics (way to recognize people).
- Robotic and machine learning.

Therefore, computer vision is rewarding and rich topic for research and study, and feature extraction from the images is one of the important of those researches because features are including in all images; also features extraction is a step or path we must to pass through it in most of studies and applications related to computer vision and image processing [6, 7, 8].

## I.2 Image sources

There are many sources of images such as X-ray, gamma-ray, infra-red and ultraviolet imaging. For example, the chosen images in [Figure I.1](#); we can easily recognize the face based on the image such as in [Figure I.1a](#), whereas, our brains able to discriminate and identify that is a face, according to the face's shapes (like shape of the eyes, the nose, the mouth...). But images (b) (c) and (d) in [Figure I.1](#) are unusual and it is difficult to recognize them. Actually [Figure I.1b](#) is an ultrasound image of the carotid artery, taken as a cross section through it. The top region of the image is near the skin; the bottom is inside the neck. And [Figure I.1c](#) shows a remote sensing image, this imaging technique is often used for analysing the content texture. The perceived texture is different between the road junction and the different types of foliage. Finally, we see in [Figure I.1d](#) a magnetic resonance image (MRI) of a human body. The chest is at the top of the image, and the lungs and blood vessels are the dark areas, the internal organs and the fat appear gray.

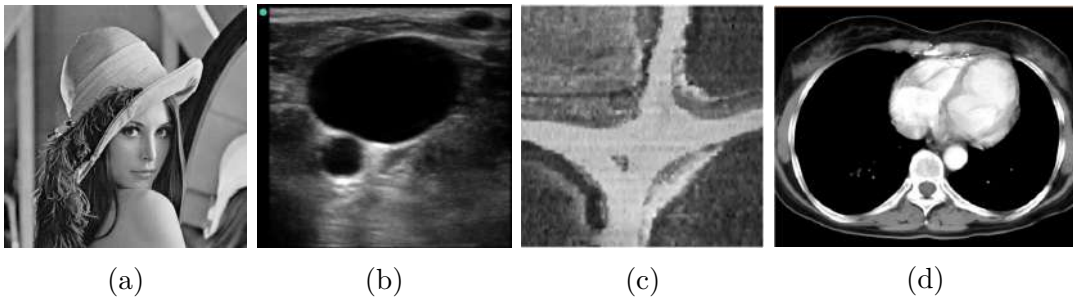


Figure I.1: Real images from different sources [6].

As shown above, there are different sources of images specially in medical studies. But computer vision techniques are used to analyse any form of data, not just the images from cameras [6, 9]. In our case, we consider these images comprise a set of points or picture elements (usually concatenated to pixels) stored as an array of numbers in a computer. Or we can say, those are digital images represented by a two-dimensional matrix of numerical values  $f(x, y)$  where:  $x$  and  $y$  are spatial coordinates and the amplitude  $f$  at any pair of coordinates  $(x, y)$  is called the intensity or gray level. And our work, focused on the automatic extraction (or description) of the features from images, features such as shape, texture, colour, etc. are used to describe the content of the image.

## I.3 Features extraction

Feature extraction is a special form of dimensionality reduction. The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space. When the input data to an algorithm is too large to be processed and it is suspected to be redundant (much data, but not much information) then the input data will be transformed into a reduced representation set of features (also named features vector) [8].

## I.4 Classification of features

Types of features depend on the type of system in which they are going to be implemented. In pattern recognition the types used most often can be divided into colour, shape, and texture features. Yet in robotic vision the types are divided into regions, lines and points. In [10] they classify the various features currently employed as follows:

- ❶ **General features:** Application independent features such as colour, texture, and shape. According to the abstraction level, they can be further divided into:
  - Pixel-level features: Features calculated at each pixel, e.g. colour, location.
  - Local features: Features calculated over the results of subdivision of the image band on image segmentation or edge detection.
  - Global features: Features calculated over the entire image or just regular sub-area of an image.
- ❷ **Domain-specific features:** Application dependent features such as human faces, fingerprints, and conceptual features. These features are often a synthesis of low-level features for a specific domain.

### I.4.1 General features

#### I.4.1.1 Colour Features

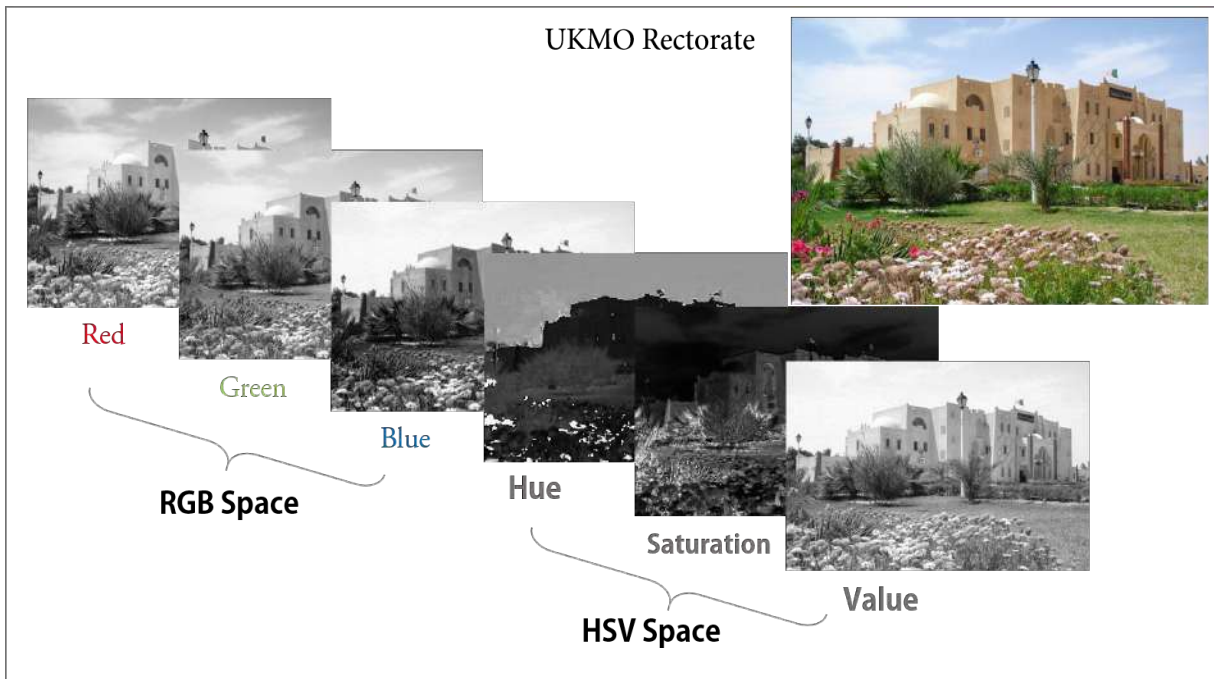


Figure I.2: The RGB and HSV colours spaces.

Colour is very important feature in colour images. Colour features represent subject to a particular colour space or model, there are many colour spaces used in colour imaging such as red, green, blue (RGB), hue, saturation, value (HSV) and luminance and chrominance (Y, Cb, Cr). When the colour space is specified; colour features can be extracted from images or regions. The extraction of colour features could be done by using many techniques (colour descriptors), including colour histogram, colour coherence vector (CCV) and colour moment (CM) [10, 11, 12].

In [13], *Z. Zheng et al* developed a robust and accurate algorithm to extract eye features from colour images, he could detect the centre of the pupil in H channel of HSV colour space as shown in Figure I.3a; Then they estimated and refined the radius of eyeball. After that they detected the eye corner by using a proposed filter which is Gabor eye-corner, a sample of results shown in Figure I.3b

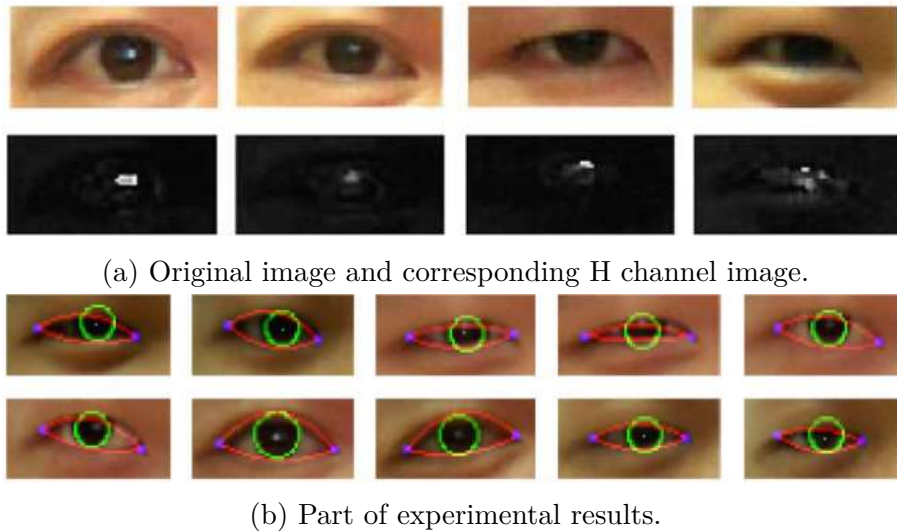


Figure I.3: Eye features extraction on colour image [13].

#### I.4.1.2 Texture Features

Texture is one of the very useful characterizations of images. In fact, human visual systems use texture for interpretation and recognition. Usually the colour is a pixel property (could be one point) while texture can only be measured from a group of pixels. A large number of techniques have been proposed to extract texture features; such as Fourier power spectra and multi-resolution filtering techniques such as Gabor and wavelet transform, all of these techniques characterize texture by the statistical distribution of the image intensity. In [10] Gabor functions analysis was used in order to extract iris image features which consists of convolution of the image with complex Gabor filters, the detected features were used as personal identities for recognition purpose as shown in Figure I.4;



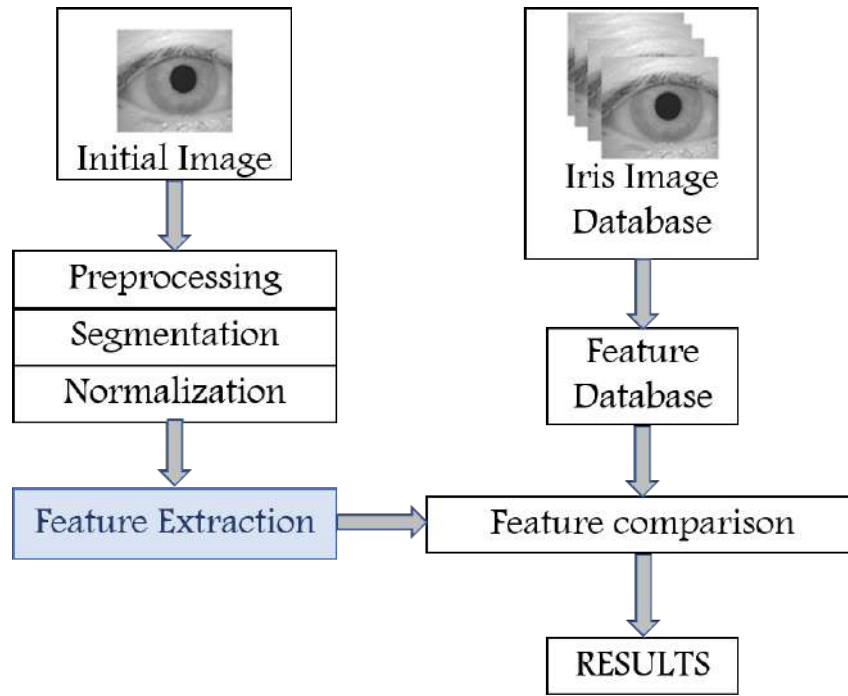


Figure I.4: Typical iris recognition stages[10].

### I.4.1.3 Shape Features

Shape is known as an important visual feature and it is one of the primitive features for image content description, whose purpose is to encode simple geometrical forms such as straight lines in different directions. Shape feature extraction techniques can be divided into two main categories: region based and contour-based methods [10, 11, 12]. These types of features will be discussed in more details in the next section.

## I.4.2 Domain-specific features (Robotic Vision Domain):

### I.4.2.1 Regions (or Surfaces)

They can be projections of closed areas, water tanks, lakes, buildings or shades. They are often represented by their gravity centres, which are invariant to rotation, dilation and to deviation, and stable under a random noise and variation of gray level. Those regions are detected by means of some segmentation methods [14] ; therefore the precision of the segmentation can influence the result of detected features. Recently, researchers are interested in the selection of regions invariant to scaling. For example, Alhichri and Kamel proposed the idea of virtual circles, by using the distance transformation [15].

### I.4.2.2 Lines (or Curves)

They can be representations of general segments of lines, contours of objects, borders of regions, roads or rivers. For their detection, standard methods of edges detection like Canny detector, or a detector based on Laplacienne of Gaussian, are used. The lines are often represented by pairs of points of extremities, or by their points of medium [16][17]. In [18], they presented a method to localization and navigation the state of the

robot on the football field by using line-based features. The results of the work shown in Figure I.5. The top-figure shows an image taken from the robot's front camera. The purple line denotes the detected field boundary, red (green) lines show field lines (not) used for localization. Detected corners are marked as "X" or "T". Bottom left: egocentric view with everything used for localization. Bottom right: resulting localization using the particle filter.

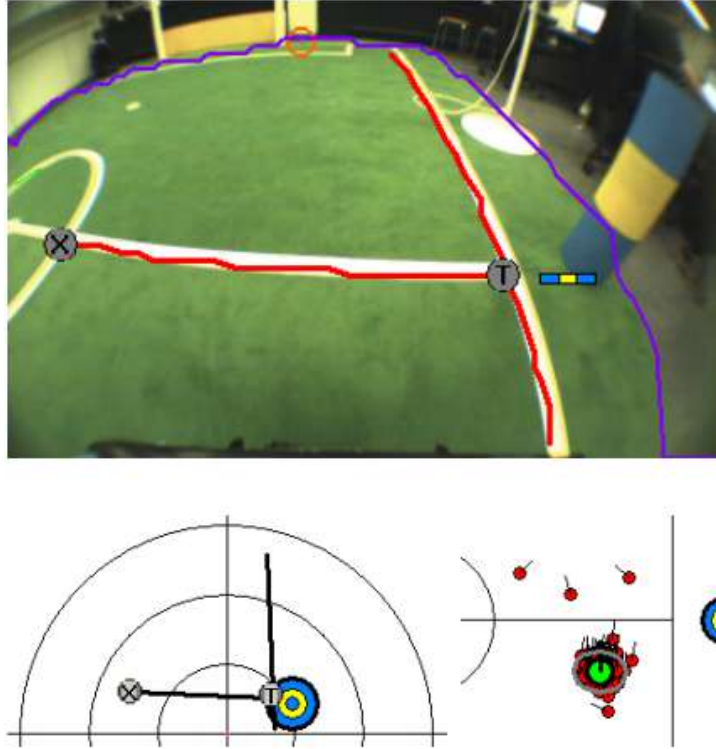


Figure I.5: Localization using line and corner features[18].

### I.4.2.3 Points

Points are ideal for image registration because their coordinates can be used directly to determine the parameters of the transformation function, and also due to their invariance to the image geometry and their facilities to detect by a human observer. This type of primitives are the most desired features in computer vision, because they can be easily visible and can be detected using simple detectors [19][20].

The proposed technique in [21] was for airborne enabling unmanned aerial vehicles to construct a reliable map of an unknown environment and localise themselves within this map without any user intervention, building of this map is based on detecting distinguished points-based features on all captured images using SIFT detector.

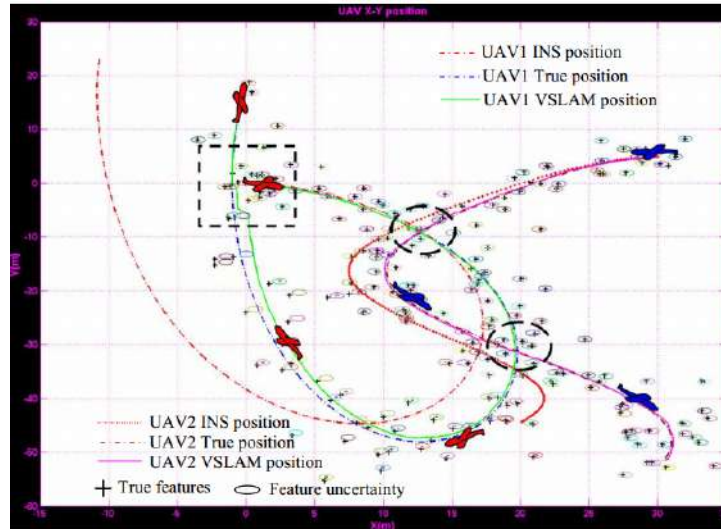


Figure I.6: The trajectories of the two UAVs in the X and Y axes[21].

## I.5 Conclusion

From this chapter we can conclude that digital images are used in many domains (surveillance, traffic, military, biometry and robotics etc.) from different imaging sources, that makes image processing rich topic and reward for study.

The digital images contain a lot of information, such as textures, colours, and points..., those later are one of the most important features are used in robot vision. Feature retrieval techniques help to make the processing faster and more reliable.

In the next chapter, we will see many techniques and algorithms which allowed us to extract the corners point features, and we are going to see the propriety of those extraction techniques.

# *Points-Based Features Detectors*

---

II.1 Introduction . . . . .	10
II.2 Points-based features detectors (State of the Art) . . . . .	10
II.3 Classical detectors . . . . .	11
II.4 Modern detectors . . . . .	14
II.5 Binary Descriptors . . . . .	18
II.6 Conclusion . . . . .	22

---

## **II.1 Introduction**

The concept of features has been widely used in order to solve many problems in computer vision domain such as image registration, and visual tracking. The main advantage of features detection is selecting special parts in the image and doing the desired analysis on them. The most desired features are points, because their coordinates can be directly used to determine the parameters of a transformation function that registers the images. In some images it may not be possible to detect point features; however, lines or regions may be detected. In such situations points are derived from the lines and regions. For example, both intersections of corresponding line pairs produce corresponding points. In this chapter we will present a review of the methods used for points based features detection.

## **II.2 Points-based features detectors (State of the Art)**

Many types of detectors were developed to extract features from an image, Harris corner detector [22] which was proposed in 1988 is the most used detector. It is based on the eigenvalues of the second moment of the intensity matrix. But the Harris corners are not invariants to large scale change. In [19] *Goshtasby* tried to detect interest points in the image, each with its own characteristic scale by introducing the concept of the automatic scale choice. In [14], for creating robust detectors and which are invariant to

the change in scale; Harris-Laplace and Hessian-Laplace detectors were invented. They used the measure of Harris (adapted scale) or the determinant of the Hessian matrix to choose the place, while the Laplacian is used to choose the scale. Depending on the Difference of Gaussians (DoG), other type of features detector called SIFT was proposed by Lowe as discussed in [23]. From the review of the existing features extractor, H. Bays developed a new type of key point's detector called SURF for improving the speed and the precision of detection [16, 24].

## II.3 Classical detectors

### II.3.1 FAST detector

The Features from Accelerated Segment Test (FAST) corner detector was developed by Rosten and Drummond in 2006; it has a simple and fast corner detection algorithm to find local invariant points. It finds corners in the image by comparing pixel-gradients in a neighbourhood of pixels. FAST algorithm defines corner point as: (In the neighbourhoods of a pixel, there are enough pixels in different region and their gray values are greater than or less than the central pixel's [25]). The Corner Response Function to judge whether a pixel is a corner point is defined as CRF as follows:

$$N = \sum_{x \in \text{circle}(p)} |I(x) - I(p)| \prec \varepsilon \quad (\text{II.1})$$

Where:

- $p$ : means the central pixel;
- $I(p)$ : means the gray value of pixel  $p$ ;
- $I(x)$ : means gray value of the neighbourhood;
- $\varepsilon$ : is a given threshold value.

If  $N$  is greater than a given threshold, this pixel point is considered as a corner point. However, some pseudo corner points can appear with this algorithm. To extract

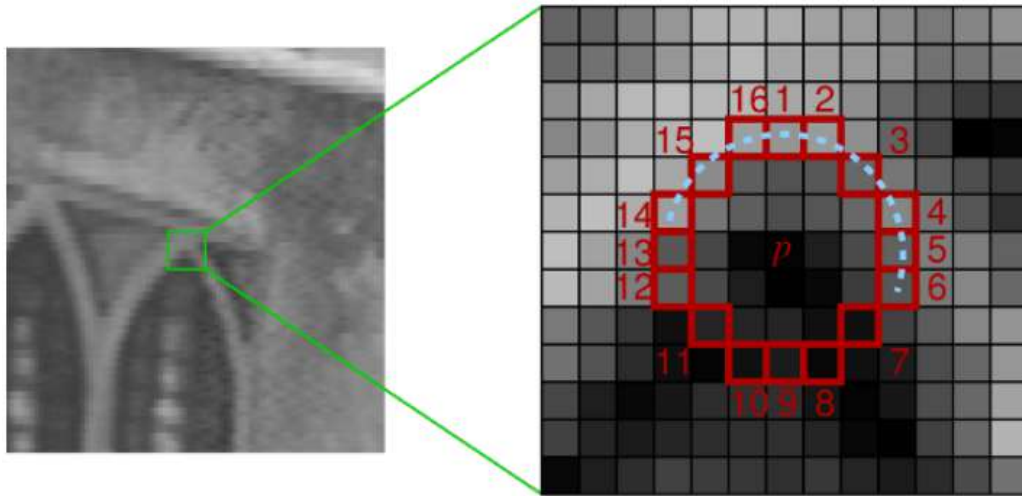


Figure II.1: Visualization of the FAST corner detection feature [26].

Fast corners, a grey scaled image is sufficient and allows much faster extraction than a coloured one. In order to detect an existing corner, the grey scale of the pixels lying on the discrete circle is compared with the centre pixel  $p$ . If a certain consecutive number of differences lie above or below a certain threshold  $t$ , the considered pixel is marked as corner. The chosen threshold serves as parameter for controlling the total numbers of extracted corners in a given image [26, 27].

FAST detects sometimes more than one corner in a certain neighbourhood. In order to reduce the number of found corners only the corner with the highest cornerness is kept. This is as well called non-maximal suppression [28]. Optimizing the calculation cost can be done by examining the pixels 1, 9, 5 and 13 because a feature can only exist when three of them are beyond the threshold. Due to low cost of comparing a small amount of pixels, the FAST feature is much faster than the SUSAN or the Harris feature. The amount of potential frames per second depends on the threshold. The higher the threshold, the less feature get detected and the higher the speed of detection.

### II.3.2 Harris detector

The Harris corner detector was proposed by Harris C and Stephens MJ in 1988, this detector was an improvement on Moravec's Corner Detector. It is based on the local auto correlation function of a signal, which measures the local changes of the signal with patches shifted by a small amount in different directions [20]. The corners image features are discrete, reliable and meaningful, therefore; they were involved in several computer vision application since a long time. The basic idea of this detector is the necessity of easily recognizing the point by looking at intensity values within a small window and by shifting the window in any direction, we should have a large change in appearance [29]. To determine the nature of a point (i.e, if point is considered as an interest point), Harris proposed computing the average change of intensity in the image and shifting a small local window in the image by a small amount (by one pixel) in any direction [20], point nature can be determined as follows:

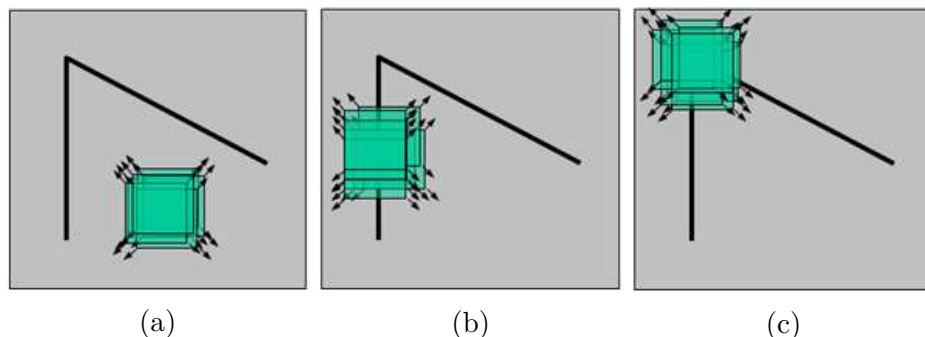


Figure II.2: Nature of a point with Harris algorithm; (a)Region,(b)Edge,(c)Corner [20].

In Harris detection algorithm, for a point to be considered as corner, all shifts (in at least one of the opposite directions) should produce a significant intensity change.

### II.3.2.1 Stages of Harris corner detector

#### ❶ Applying corner operator:

For each pixel in the image, the corner operator is applied to obtain a cornerness measure for this pixel. The cornerness measure is simply a number indicating the degree to which the corner operator believes this pixel is a corner. Interest point corner detection algorithms differ on how the corner operator makes this measurement, but all algorithms consider only pixels within a small window centred on the pixel a measurement is being made for. The output of this step is a cornerness map [30, 31]. Since for each pixel in the input image the corner operator is applied to obtain a cornerness measure, the cornerness map has the same dimensions as the image.

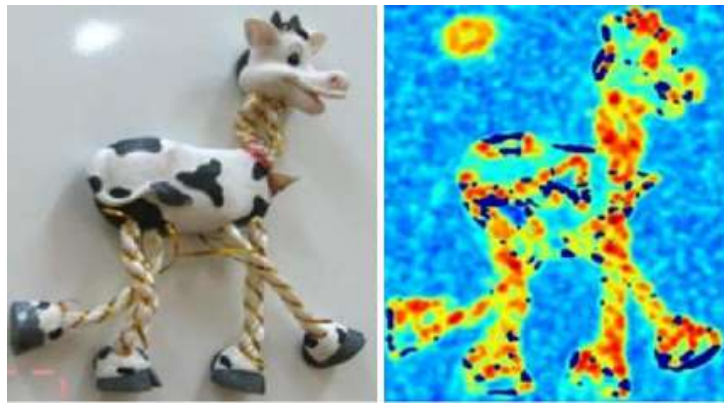


Figure II.3: Applying corner operator.

#### ❷ Threshold cornerness map:

Interest point corner detectors define corners as local maximum in the cornerness map. However, at this point the cornerness map will contain many local maxima that have a relatively small cornerness measure and are not true corners. To avoid reporting these points as corners, the cornerness map is typically threshold [32]. All values in the cornerness map below the threshold are set to zero. Choosing the threshold is application dependent and often requires trial and error experimentation.

The threshold must be set high enough to remove local maxima that are not true corners, but low enough to retain local maxima at true corners. In practice there is rarely a threshold value that will remove all false corners and retain all true corners so a trade-off must be made based on the requirements of the application.

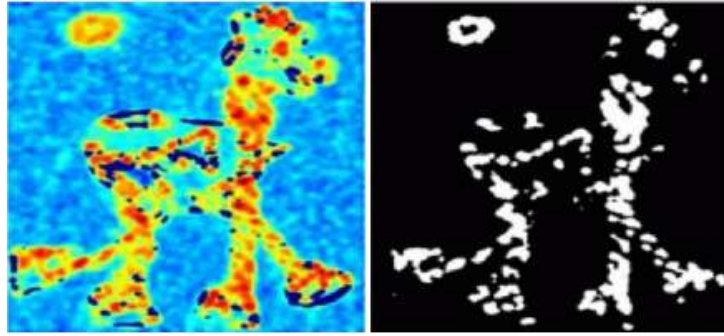


Figure II.4: Finding points with large corners response ( $R > \text{threshold}$ ).

### ③ Non-maximal Suppression:

The threshold cornerness map contains only non-zero values around the local maxima that need to be marked as corner points. To locate the local maxima, non-maximal suppression is applied. For each point in the threshold cornerness map, non-maximal suppression sets the cornerness measure for this point to zero if its cornerness measure is not larger than the cornerness measure of all points within a certain distance. After non-maximal suppression is applied, the corners are simply the non-zero points remaining in the cornerness map [31, 32].



Figure II.5: Taking only the points of local maxima of  $R$ .

#### *Properties and limitations:*

- \* Rotationally invariant.
- \* Partially invariant to affine intensity change.
- \* Sensitivity to noises.
- \* Non-invariant to large image scale.

## II.4 Modern detectors

### II.4.1 SIFT detector/descriptor

Scale Invariant Feature Transformation detector/descriptor (SIFT) was proposed by Lowe in 2004 [33]. Firstly, this detector uses a scale-space extrema to efficiently detect



the location of those stable key points in the scale and space. Then, an orientation histogram based on the gradient in different directions is formed around the key point and the dominant orientation is used to represent the key point orientations. Finally, a gradient histogram is created as a very distinctive descriptor of that key point. Thus, each key point is represented by the scale and orientation.

#### II.4.1.1 SIFT detector

The key point is selected based on the Difference of Gaussian by detecting locations that are invariant to scale change of the image, this can be accomplished by searching for stable features across all possible scales, using a continuous function of scale known as scale space. To detect the key points, scale octave is generated and the local extrema is detected by comparing the centre pixel with the neighbours in space. The DoG image can be computed from the difference of the two nearby scales separated by a constant factor  $k$  [21]:

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (\text{II.2})$$

Where  $*$  is the convolution operation in  $x$  and  $y$ , and  $L(x, y, \delta) = G(x, y, \delta) * I(x, y)$  defines the scale space representation of an image, with:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (\text{II.3})$$

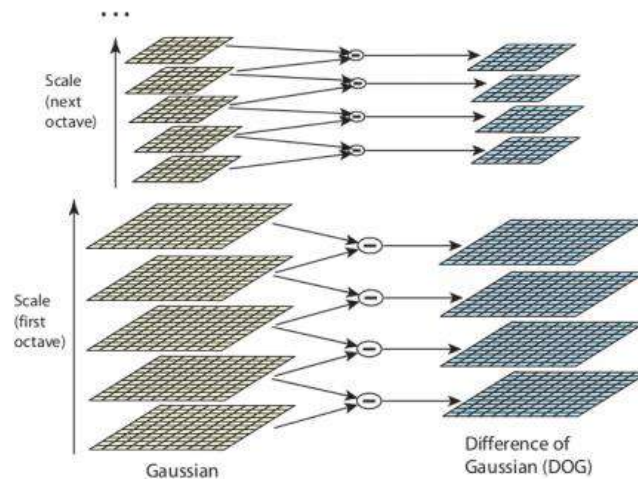


Figure II.6: SIFT detector generation.

#### II.4.1.2 SIFT descriptor

To describe the key points, SIFT makes use of the local gradient values and orientations of pixels around the key point. A key point descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the key-point location [34]. SIFT descriptor is a classic approach, also the "original" inspiration

for most of the descriptors proposed later. Up to date, it still outperforms most of the descriptors in the field. The drawback is that it is mathematically complicated and computationally heavy.

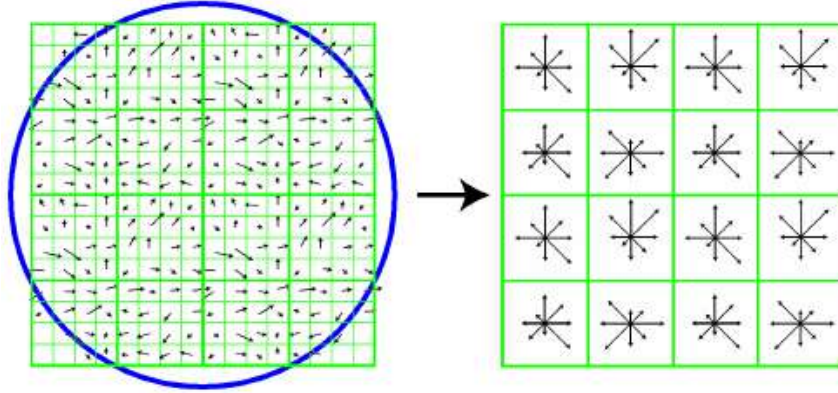


Figure II.7: SIFT descriptor generation.

### *Properties of SIFT [21]:*

- \* It detects suitable number of invariant and distinctive features.
- \* Extracted features face robustly with significant image changes (large image translation and rotation, scale change and photometric changes).

## II.4.2 SURF detector/descriptor:

SURF (Speed-Up Robust Features) is an image detector/descriptor which is widely used in the computer vision community, SURF was first developed by Herbert Bay in 2006 [23, 24]. It was designed to accelerate the detection of features that have good invariance properties. In an interesting way, the authors of SURF experimentally demonstrated that their new sensor exceeds the SIFT detector and many others in terms of speed and accuracy [21]. The approach for detecting points of interest using the SURF algorithm is based on the approximation of the Hessian matrix by using the image integral. This technique will reduce the computing time in a big way. The SURF algorithm consists of three main steps [36]:

1. Integral image generation.
2. Interest point localization.
3. Interest point description.

### II.4.2.1 Integral image generation

The integral image  $I_{\Sigma}(p)$  at a location  $p = (x, y)$  represents the sum of all pixels in the input image  $I$  of a formed rectangular region:

$$I_{\Sigma}(p) = \sum_{i=0}^x \sum_{j=0}^y I(x, y) \quad (\text{II.4})$$

Integral image  $I_{\Sigma}(p)$  in  $P = (x, y)$  represents the sum of all the pixels on the left and top of  $P$ . This integral is used in both the subsequent interest point detection and description to obtain higher efficiency. Once integral image is computed, it takes only 3 additions/subtractions to get the sum of the pixels intensities over an upright rectangular region ( $\Sigma = I_{\Sigma}(D) - I_{\Sigma}(C) - I_{\Sigma}(B) - I_{\Sigma}(A)$ ). Another benefit is that the calculation time is independent of the box size [35, 37].

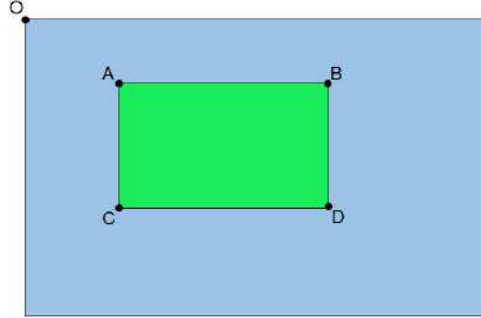


Figure II.8: Functionality of integral image.

Using integral images, it takes only four operations to calculate the area of a rectangular region of any size.

#### II.4.2.2 Interest point localization.

SURF detector locates features based on the Hessian matrix, which is defined as:

$$H(P, s) = \begin{pmatrix} L_{xx}(P, s) & L_{xy}(P, s) \\ L_{xy}(P, s) & L_{yy}(P, s) \end{pmatrix} \quad (\text{II.5})$$

Where  $L_{xx}(P, s)$  denotes the convolution of Gaussian second-order derivative in x direction with input image in point  $P$  at scale  $s$ , and similarly for  $L_{xy}(P, s)$  and  $L_{yy}(P, s)$ . Using the integral image simple box filters are used to approximate the second-order Gaussian partial derivation and yielding less computation cost (see Figure II.9).

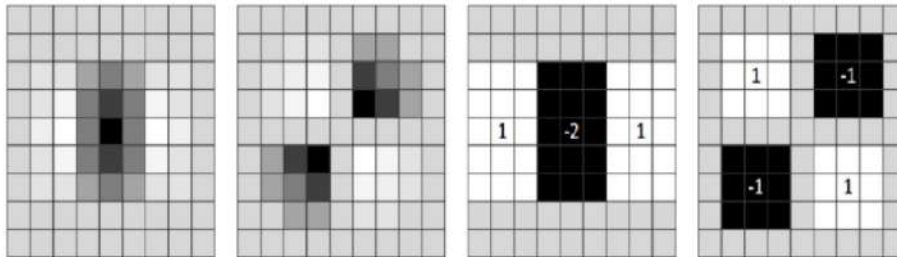


Figure II.9: Approximations of Gaussian 2<sup>nd</sup> order partial derivatives by box filters [27].

The left half shows Gaussian second order partial derivative in x-and xy-direction; the approximation for them - box filters, are presented in the right half, respectively. The grey regions are equal to 0.

The problem thus reduces from calculating Gaussian second-order derivative responses to the box filter responses. Denoting the blob responses by  $D_{xx}$ ,  $D_{yy}$  and  $D_{xy}$ , then the determinant of the original Hessian matrix in SURF is approximated as follows [27]:

$$\det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \quad (\text{II.6})$$

Where 0.9 is used to balance the Hessian determinant. In order to achieve scale invariance, SURF applies box filters of different sizes on the original image to search and compare interest points. Box filters of different sizes construct the scale space, which is divided into octaves. The local maxima of box filter responses larger than a pre-defined threshold in image and scale space are selected as interest point candidates. Non-maximum suppression in a 3 x 3 neighbourhood is applied to screen out “false” candidates with position correction elements above 0.5 and localize interest points [37].

### II.4.2.3 Interest point description.

SURF builds a descriptor around the neighbourhood of each interest point. First, a square region of 20s-by-20s centred on the interest point is constructed along the dominant direction. In order to keep it simple, the dominant directions of interest points are set to be upright. The region is then divided into 4 x4 smaller sub-regions with each window size 5s-by-5s (sampling step s). For each of these sub-regions, Haar wavelet responses (filter size 2s) are computed at 5 x 5 regularly distributed sample points. These responses are then weighted by a Gaussian function ( $\delta = 3.3s$ ) centred at the interest point [38, 39].  $d_x$  and  $d_y$  are used to denote weighted Haar wavelet response in horizontal direction and vertical direction. Each sub-region generates a 4-D vector  $V = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$ . All sub-regions are then concatenated into a vector, resulting in a 64-dimensional descriptor vector.

#### *Properties of SURF detector [23]*

- \* Invariant for the change of scale, orientation changes.
- \* Invariant for geometric and photometric transformations.
- \* A high repeatability.
- \* Faster compared to other algorithms (e.g. SIFT, Harris- Laplace, etc.).

## II.5 Binary Descriptors

Over the last years several new fast detectors and descriptors (BRIEF, ORB, FREAK) have been proposed and successfully applied to the robot navigation tasks and also in computer vision systems [39].

As clarified in the previous section, SIFT and SURF are both efficient, particularly the last mentioned, and give a great execution in detecting key point and extricating

descriptors for each one. In any case, they are based on gradient histograms where every single pixel need to be analysed, which costs computational time. In spite of the utilise of fundamentally images adopted by SURF, a few applications request speedier performances.

Binary descriptors provide the ability to encode all the information regarding the surrounding of a feature point as binary strings. Most recent binary descriptors consist of a sampling pattern, an orientation compensation method and a sampling pairs system. The pattern, which is distinctive for the chosen descriptor, is overlapped with the area around the detected key point and centred on it. Such sampling pattern is ideally a set of concentric circles. A number of pairs of points is chosen on the pattern, and the intensity value of each point in the pair is compared with its matched one. If the first result is larger then the second, the value “1” is written in the string, “0” otherwise. When all the pairs have been analysed, the information describing the area around the key point will be encoded in a string of "0" and "1". The orientation compensation is a mechanism where the orientation of the interesting area is calculated relatively to some intrinsic feature of the area itself. The chosen pairs are rotated to that same angle, before evaluating the intensity, to make sure that the binary descriptor will result rotation invariant. In the next sections two binary descriptors, which have been evaluated for this thesis purpose, are described to give a brief overview of their function. Despite being relatively new, BRISK and FREAK binary descriptors far surpass the industry standards as they perform much better than the predecessor (sift and surf) in this field [40].

### II.5.1 BRISK descriptor

Taking as input a set of key points, the BRISK descriptor creates a binary string by concatenating the results of intensity comparisons, as previously mentioned. In contrast with other binary descriptors, BRISK adopts a custom sampling pattern where points lie on scaled concentric rings (Figure II.10).

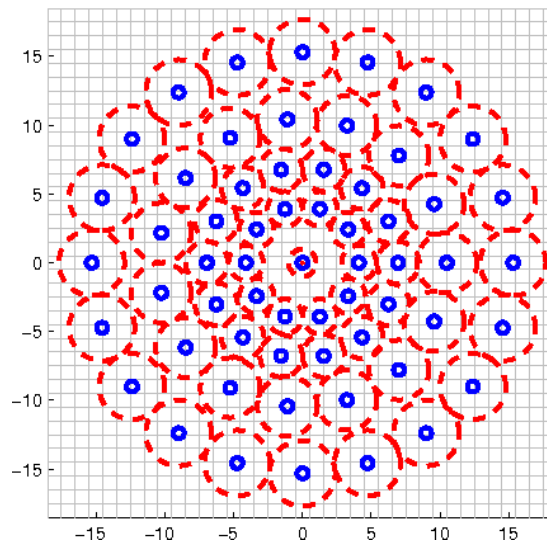


Figure II.10: BRISK sampling pattern[41].

When the points are analysed, a small area around them, which radius is equal to the red circles shown in the pattern above, is taken and smoothed with Gaussian. The BRISK algorithm defines the pairs in two subsets: short-distance-pairs, adopted to compute the intensity comparison necessary to assemble the descriptor, and long-distance-pairs, used to determine orientation (Figure II.10). Short-distance-pairs consist of sampling points which distance is below a certain threshold  $\Delta_{max}$ . The distance of long-distance-pairs sampling points is above a certain threshold  $\Delta_{min}$ , different from the previous one. The two thresholds are set as  $\Delta_{max} < \Delta_{min}$  such as no short-distance-pair is also a long-distance-pair.

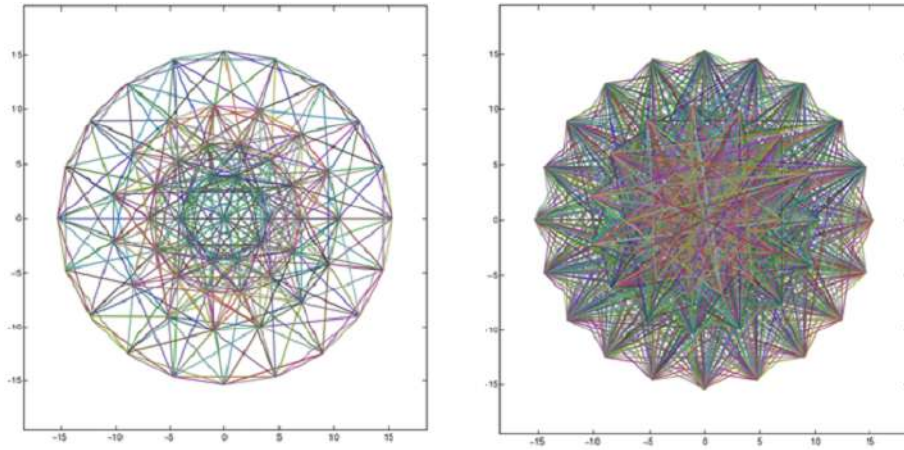


Figure II.11: Short distance pairs on the left and long distance pairs on the right.

The local gradients between long distance pairs are computed and the entire set is summed to estimate the feature orientation. Rotation invariance is obtained by rotating the short distance pairs at the same angle as the key point orientation [41]. For each pair the Gaussian smoothed intensity of one sampling point is compared with the other of the pair. As said before, if the intensity is larger in the first point "1" is written as result, "0" otherwise. All this comparing will finally result in the binary descriptor. Given a sample number of 60, the two subsets of distance pairing will count a total of 870 long and 512 short distance pairs. The descriptor will result 512 bits long.

The algorithm for the extraction of the BRISK descriptor is the following [40]:

1. A concentric sampling pattern is created.
2. A set of long range pairs and a set of short range pairs are created.
3. The global orientation is estimated using the set of long range pairs.
4. The sampling pattern is rotated in the direction of the computed orientation.
5. In order to avoid noise, the sampling pattern defines Gaussians centred at each point, so that the differences of the pairs are in fact differences of Gaussians.
6. The descriptor is constructed using a deterministic set of 512 short range pairs.

## II.5.2 FREAK descriptor

Similarly to BRISK; FREAK also makes use of a hand-crafted sampling pattern. In their work *Alahi et al* [42], suggest the use of a pattern similar to the human retinal sampling grid as shown in [Figure II.12](#) where the density of receptive areas is higher in the centre.

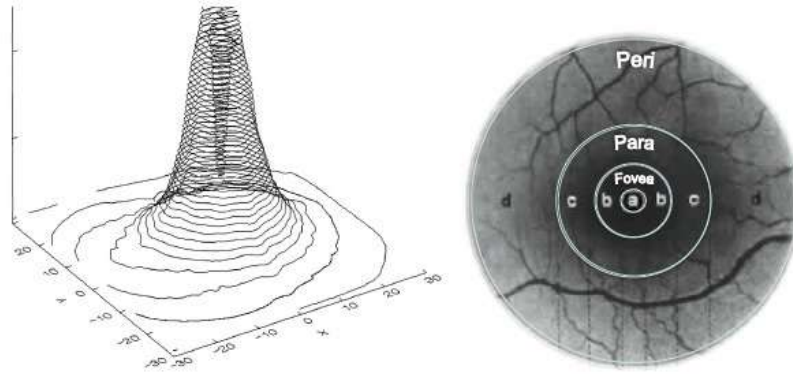


Figure II.12: Receptive fields distribution over the retina in a human eye[42].

To mimic this behaviour, a pattern where the points density drops exponentially with the distance from the centre has been designed. Differently from BRISK the circles - receptive fields - overlap while their size grows exponentially instead of gradually. Each sample point needs to be smoothed to make it less sensitive to noise. The rings shown in [Figure II.13](#) represent the standard deviation of the Gaussian kernel applied to the corresponding sampling point [42].

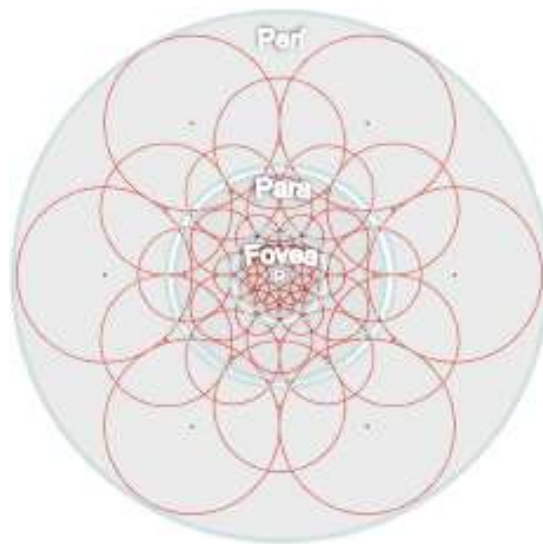


Figure II.13: FREAK sampling pattern [42].

The overlapping introduce redundancy which adds more discriminative power to the algorithm allowing to reduce the number of receptive fields. The same redundancy exists in the receptive field of the retina, according to *Tokutake et al* [43]. Computing the difference between pairs of key points areas and their respective Gaussian kernel, the

descriptor is built as a string of one-bit DoG. The matched pairs are selected differently from BRISK, which uses their spatial distance. In the used approach the best pairs are learned from training data, a coarse-to-fine [44, 45] (see Figure II.14).

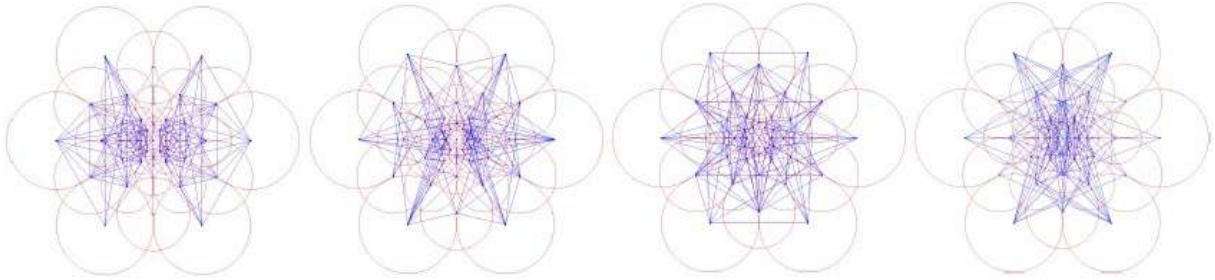


Figure II.14: FREAK's sampling pairs resulting from Coarse-to-fine analysis [42].

The first analysed pairs mostly compute feature points situated in the outer rings of the pattern. The inner ring is regularly left for the latest pairs. This is done similarly to the human eye, where the peripheral vision is used to estimate the location of an object of interest while the verification is achieved with the fovea area receptive fields.

The FREAK orientation method, which compensates rotation changes, is similar to the BRISK one. Only 45 pairs are selected, with symmetric receptive fields with respect to the centre. The higher number of receptive fields in the inner circle grants more errors handling than BRISK. This brings to a discretization of the space of orientation driving to a load of 5 times less memory[42].

## II.6 Conclusion

Different techniques were developed for detecting image features, for classical methods, features can be detected using simple algorithms, with no features description. For modern methods, features can be detected and described using complex algorithms, however, for binary methods, features of the previous methods may be described with easy operations.

The following chapter will use features detection as the starting step in two important image processing applications, which are image mosaicing and object tracking.



# *Image Mosaicing and Template Object Tracking*

---

III.1 Introduction . . . . .	23
III.2 Image mosaic issues . . . . .	24
III.3 Features extraction . . . . .	27
III.4 Features matching (Association) . . . . .	28
III.5 Geometric transformations . . . . .	34
III.6 Image projection "Warping" . . . . .	41
III.7 Template object tracking . . . . .	42
III.8 Conclusion . . . . .	43

---

## **III.1 Introduction**

In image processing, there are many applications which require high resolution single image with large view of a scene to achieve some required analysis. For example, in biological and medical applications, it is often necessary to analyse a complete scene section at high resolution which has large dimensions (a large number of pixels). However, in some cases the high resolution single image cannot be viewed even if using cameras with tens of millions of active pixels. The most common approach is to acquire several images of parts of the scene at high magnification and assemble them into a composite single image which preserves the high resolution. This process of assembling the composite image from a number of images is known as 'image mosaicing'. The basic idea of this technique is to find a suitable planar transformation which allows warping images into a single and common reference frame [46].

Image mosaicing is necessary in several applications such as:

- Construction of extended geographical maps.

- Tracking of moving objects.
- Creation of panoramas.
- Reconstruction of 3D scene by integrating images acquired from different sides.

## III.2 Image mosaic issues

### III.2.1 Image registration

#### III.2.1.1 Geometric Image Registration

Methods of geometric image registration fall into two broad categories [47]:

##### ❶ Direct methods

For large overlapping regions between images and small translations and rotations, direct methods or feature-less method can be useful. Direct methods compute the transformation between images by maximizing the photometric consistency over the whole overlapping image regions, these methods can be classified into [48, 49]:

##### \* Frequency domain:

Methods based on the frequency domain are based on phase-correlation in order to estimate the translations between an image pair. After that, log-polar coordinates were proposed for rotation and scale transformations models. Those methods are not preferred for use, because they are computationally expensive, as they require Fast Fourier Transform (FFT) to be computed over all the involved images.

##### \* Optical flow:

These methods are based on the estimation of the disparity of pixels between image pairs with assumption that, the photometric properties of image pixels (luminance and colour) remain constant according to Brightness Constancy Model (BCM).

##### ❷ Feature based methods

Contrarily to direct methods, feature based methods do not require a high percentage of overlapping between consecutive images to estimate the transformation model between them. This class of methods uses a sparse set of corresponding image features (points) to estimate the image to image mapping.

For given two different views of the same scene which are taken at different times, from different viewpoints, and/or by different sensors. Feature based method of geometric image registration is to find the accurate point to point correspondence between those images by finding for each image point in one view the image point in the second view which corresponds to the same actual point in the scene [36]. This process is a critical stage in various image processing applications; in which final information is obtained starting from combining various data sources [50].

### III.2.1.2 Photometric Image Registration

Photometric image registration refers to the procedure by which global photometric transformation between images is estimated and compensated. Examples of such transformations are:

- Global illumination changes across the scene.
- Intensity variations due to camera automatic gain control or automatic white balancing [11].

To solve these problems, In the literature [21], a suitable model of photometric transformation was proposed, and then; the parameters of this model was identified.

### III.2.2 Image Re-projection

After image registration, every point in every image can be transformed to a point in the global frame. In order to render an image mosaic from the set of all overlapped images and transformation models (homographies), it is necessary to map points of every image to points in the rendered image using one of the projections manifold [47, 36]:

#### III.2.2.1 Planar projection

Since the projective distortion of the back-projected images increases toward the periphery of the mosaic, the resulted image mosaic will have form similar to the classic "bowtie". The planar manifold is suitable for both general-scene/rotating- camera and planar scene/general-motion cases. But with image sequences which sweep a large angle ( $> 90$  degrees), the projective distortion means that the mosaic image becomes infinite in size, and the planar manifold cannot be useful any more.

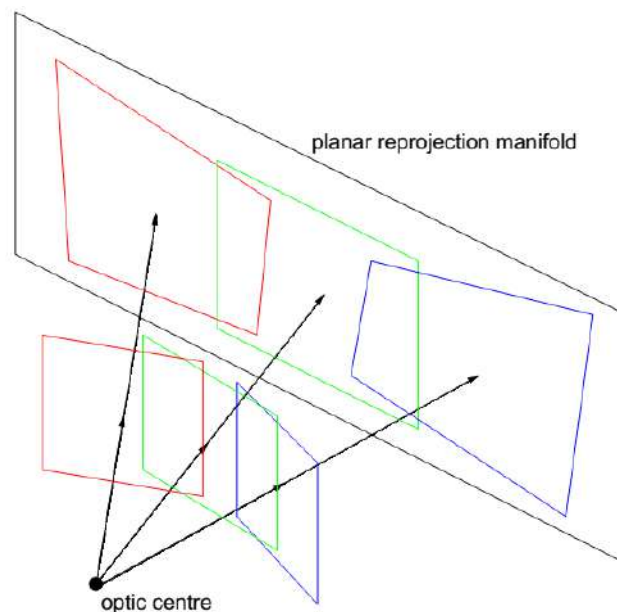


Figure III.1: Mosaic rendering by re-projection onto a planar manifold [36].

### III.2.2.2 Cylindrical projection

In the case where image sequence is obtained from camera rotates about a single axis, a cylindrical manifold is best suited, in which for a very large angle (possibly a full 360-degree sweep); the mosaiced image does not suffer from the same projective distortion seen in the planar manifold projections.

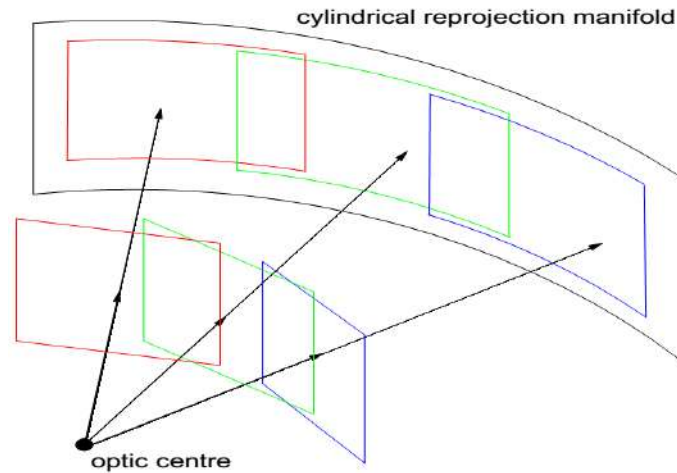


Figure III.2: Mosaic obtained by re-projection onto a calibrated cylindrical manifold [36].

### III.2.3 Image Blending

Since the used images that are used will slightly not have perfectly matching pixels at all regions where they overlap, the image blending calculations are designed to average and more properly meld all the images together [51]. In addition, this calculation is aimed at eliminating the boundary line from one image to another; it is often the case that significant global photometric differences can occur between images in a sequence. If not corrected, this can give rise to unsightly seams in rendered mosaics.

Common methods of image blending include simple averaging of intensity values, feathering and temporal median filtering. Better results may be obtained if photometric registration and correction is performed prior to rendering the mosaic [52].

#### III.2.3.1 Weighted Image Blending

The first idea of image blending was to simply take the average between intensities of the two images in the overlapped region [53]. This can give a good performance, but if the images contain mobile objects, or if the parameters of the transform function (homography matrix) are not very precise, then, it is more likely to have an undesirable effect of blurring. The following image in Figure III.3 illustrates this effect:

Therefore, to eliminate this effect, as shown in Figure III.4 we should take into account the distance between pixels and the edges of the two images.

A continuous blending of intensities can be obtained by taking the weights of the pixels in the region  $C$  for images 1 and 2 inversely proportional to the distances of the pixels to



Figure III.3: Blurring effect after blending [53].

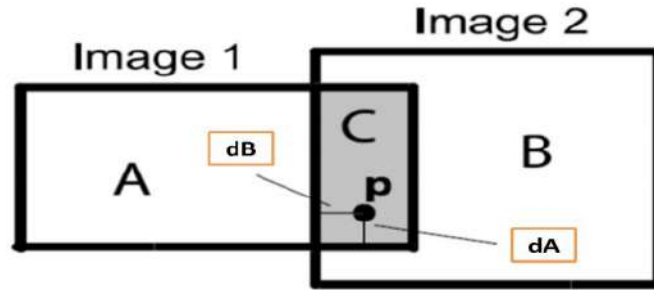


Figure III.4: Blending based on distance.

nearest boundaries of the two images [53, 54]. The intensity of  $p$  will be equal to:

$$I_p = \frac{I_p^A d_A + I_p^B d_B}{d_A + d_B} \quad (\text{III.1})$$

Where:

$I_p^A$ : Intensity of image 1 at point  $p$ .

$I_p^B$ : Intensity of image 2 at point  $p$ .

$d_B$ : The distance of pixel  $p$  to the nearest pixel to A.

$d_A$ : The smallest distance of pixel  $p$  to B.

### III.3 Features extraction

In both of discussed applications in this report (mosaicing and object tracking), feature extraction is the first stage to achieve the mosaic or the tracking. Here, those features must be points. So, we can extract them through the corners detectors which we mentioned in the previews chapter.

We chose the classical based detectors (FAST/Harris) to obtain the mosaiced image or to tracking the template object. Because those classical detectors are easier to implement on FPGA than the modern detectors.

### III.3.1 Properties of good features

In an image, the detected features should satisfy some characteristics in order to be good features [29]:

— **Repeatability:** Given two images of the same object or scene, taken under different viewing conditions, a high percentage of the features detected on the scene part visible in both images should be found in both images.

— **Accuracy:** The detected features should be accurately located, both in image location, as with respect to scale and possibly shape.

— **Locality:** Features should be local, so as to reduce the probability of occlusion and to allow simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions (e.g., based on a local planarity assumption).

— **Quantity:** The number of detected features should be sufficiently large, such that a reasonable number of features are detected even on small objects. However, the optimal number of features depends on the application. Ideally, the number of detected features should be adaptable over a large range by a simple and intuitive threshold. The density of features should reflect the information content of the image to provide a compact image representation.

— **Efficiency:** Preferably, the detection of features in a new image should allow for time-critical applications.

— **Invariant to geometric transformations:** Which means scaling, rotation and translation transformations.

— **Invariant to photometric transformations:** If the image is viewed in different light conditions, the same features must be detected.

For image mosaicing, repeatability is an important condition in the matching process (association), because the matching should be performed between the repeated features in the two overlapped images, and if one feature is not found in one of the two images, a false association error may occur; which will lead to deformed image transformation model, thus a deformed image mosaic can be obtained.

### III.4 Features matching (Association)

The matching is to find for each point of an image, its correspondent in the other image knowing that the image points are projections of the real 3D points of the same scene. Several matching methods were proposed in the literatures [36]:

- Methods based on correlation comparison criteria.
- Methods based on a comparison between the features descriptors.
- Methods based on tracking points of interest.

### III.4.1 Correlation based features matching

Once the feature points of the two images are detected separately by any type of feature detectors, correlation-based matching algorithm is certainly easier to implement and debug as compared to feature-based matching algorithms. This matching algorithm requires a measure of similarity (Table III.1) in order to find the point correspondences between the two overlapped images. For each pixel key-point in one image, there are a lot of possible candidates in the other image to be examined in order to determine the best correspondence pixel key-point [55]. The problem associated with these window-based matching algorithms is that the size of the correlation windows must be carefully chosen. If the correlation windows are too small, the intensity variation in the windows will not be distinctive enough, and many false matches may result.

Table III.1: The most known correlation criteria [31].

Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j) \in W}  I_1(i,j) - I_2(x+i, y+j) $
Zero-mean Sum of Absolute Differences (ZSAD)	$\sum_{(i,j) \in W}  I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i, y+j) + \bar{I}_2(x+i, y+j) $
Locally scaled Sum of Absolute Differences (LSAD)	$\sum_{(i,j) \in W} \left  I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i, y+j)} I_2(x+i, y+j) \right $
Sum of Squared Differences (SSD)	$\sum_{(i,j) \in W} (I_1(i,j) - I_2(x+i, y+j))^2$
Zero-mean Sum of Squared Differences (ZSSD)	$\sum_{(i,j) \in W} (I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i, y+j) + \bar{I}_2(x+i, y+j))^2$
Locally scaled Sum of Squared Differences (LSSD)	$\sum_{(i,j) \in W} \left( I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i, y+j)} I_2(x+i, y+j) \right)^2$
Normalized Cross Correlation (NCC)	$\sum_{(i,j) \in W} \left( \frac{\sum_{(i,j) \in W} I_1(i,j) \cdot I_2(x+i, y+j)}{\sum_{(i,j) \in W} I_1^2(i,j) \cdot \sum_{(i,j) \in W} I_2^2(x+i, y+j)} \right)$
Zero-mean Normalized Cross Correlation (ZNCC)	$\frac{\sum_{(i,j) \in W} (I_1(i,j) - \bar{I}_1(i,j)) \cdot (I_2(x+i, y+j) - \bar{I}_2(x+i, y+j))}{\sqrt{\sum_{(i,j) \in W} (I_1(i,j) - \bar{I}_1(i,j))^2 \cdot \sum_{(i,j) \in W} (I_2(x+i, y+j) - \bar{I}_2(x+i, y+j))^2}}$

Sum of Absolute Differences (SAD) is one of the simplest of the measures which is calculated by subtracting pixels within a square neighbourhood between the reference image  $I_1$  and the target image  $I_2$  followed by the aggregation of absolute differences within the square window; If the left and right images exactly match, the resultant will be zero [30]. In Sum of Squared Differences (SSD), the differences are squared and aggregated within a square window. This measure has a higher computational complexity compared to SAD algorithms as it involves numerous multiplication operations.

Cross Correlation is even more complex to both SAD and SSD algorithms as it involves numerous multiplication, division and square root operations. In which for a feature point in the first image, cross correlation can be built with each feature point of the second image, and choosing the corresponding features as the ones with the highest correlation values in the interval  $[-1; +1]$  with a value of  $+1$  for identical features in both overlapped images. In practice, a value greater than 0.8 is considered to be a good match [29, 30].

Correlation matching is easier to implement compared to other matching techniques, but a common problem with this matching approach is that false matches can occur. In practice, a number of rules are applied before a match is accepted:

- All pairs having a correlation score above some defined threshold value can be considered as pairs of corresponding points. But a feature point could be matched with several others. Imposing unicity means that for each feature point in one image, only its strongest match in the other image is considered [56].
- Imposing symmetry condition to keep only pairs in which each point is the other's strongest match [57]. This increases the chances that the two points in the matched pairs correspond to projections of the same physical scene point.

### III.4.2 SIFT/SURF descriptors based features matching

In this method, the best candidate match for each key-point is found by identifying its nearest neighbour in the database of key-points from training images. The nearest neighbours are defined as the key-points with minimum Euclidean distance from the given descriptor vector. The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbour to the distance of the second closest [21]. Lowe rejected all matches in which the distance ratio is greater than 0.8, which eliminates 90% of the false matches while discarding fewer than 5 % of the correct matches.

With the Nearest-neighbour algorithm, the similarity score between two feature vectors is the magnitude of the difference of their descriptors, so a lower score indicates a closer match. For each feature  $p$  in image 1, we compute the difference between  $p$  and every feature  $p'$  in image 2, keeping track of the best and second-best matches. We accept a match between  $p$  and  $p'$  if the difference between  $p$  and  $p'$  is less than  $t$  times the difference between  $p$  and its second-best match from image 2. Additionally, to prevent points in image 2 from being matched to more than one feature in image 1, we output only the best match for each feature in image 2.



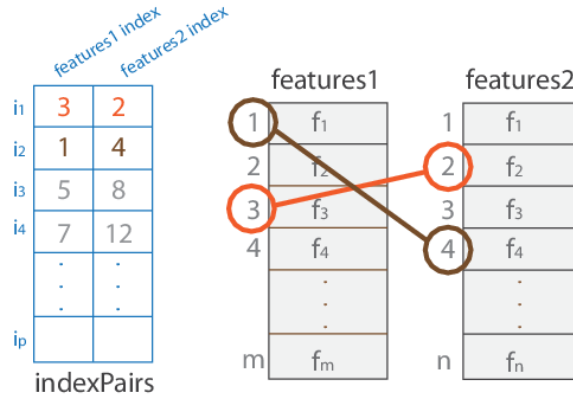


Figure III.5: Index pairs of the matched features.

This matching algorithm is characterized by:

- \* Suitable when good features can be extracted from the scene.
- \* Faster than correlation-based methods .
- \* Good for applications like visual navigation.
- \* Relatively insensitive to illumination changes.

### III.4.3 LBP Descriptors Based features matching

Local Binary Patterns (LBP) algorithm is a binary system description which expresses the relationship of size of a gray image pixel point and its neighbour-hood pixels points; it was originally used to describe image texture information. Nowadays, research workers put forward a lot of improved LBP algorithm which has been applied in features matching; face recognition, etc. because of its simple computation complexity and partial scale, rotation, and illumination invariance [58].

From the description of Local Binary Descriptors (LBDs), it is clear that they involve only simple arithmetical operations. Furthermore, the distance between two LBDs is measured using the Hamming distance, which is a simple bitwise exclusive or (XOR) instruction. Hence, computation and matching of LBDs can be implemented efficiently, sometimes even using hardware instructions (XOR), allowing their use on mobile platforms where computational power and electric consumption are strong limiting constraints. Since they also provide good matching performances, LBDs are getting more and more popular over SIFT and SURF: combined with FAST or Harris for the key-point detection, they provide a fast and efficient feature extraction and matching.

#### III.4.3.1 LBP Feature Descriptor

The original LBP operator labels the pixels of an image with decimal numbers, called Local Binary Patterns or LBP codes, which encode the local structure around each pixel. So, to describe pixel points, it should be compared with its N neighbours. The following algorithm shows how to create LBP Feature descriptor for N=8 [58, 59]:

1. The gray values of 8 neighbourhood pixel points are compared with the gray value of the central pixel point. According to the comparison sign value, binarization is done to those 8 neighbourhood pixel points, i.e. If a pixel point's gray value is greater than the central pixel point's, the gray value will be set to 1, and if a pixel point's gray value is less than the central pixel point's, the gray value will be set to 0.
2. After the binarization, the obtained binarized gray values of the eight neighbourhood pixel points should be multiplied by weight matrix as shown in Figure III.6.
3. The decimal numeral after adding the eight values up is  $LBP=1+2+4+16=23$ . And the binary vector is  $LBP = (10110010)$ .

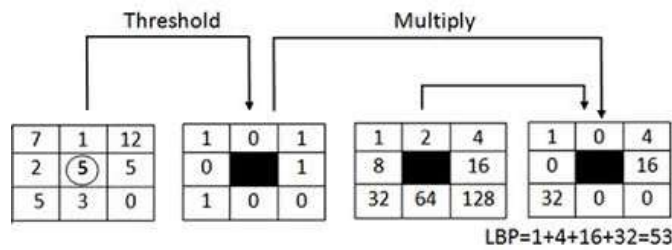


Figure III.6: The steps to create LBP Feature Descriptor ( $P=8, R=1$ ).

Calculation of LBP feature descriptor is no longer limited to  $3 \times 3$  neighbourhood, because of the basic LBP operator ( $3 \times 3$  neighbourhoods) cannot capture dominant features with large scale structures. To deal with the texture at different scales, the operator was later generalized to use neighbourhoods of different sizes [59, 60, 61].

A local neighbourhood is defined as a set of sampling points evenly spaced on a circle which is centred at the pixel to be labelled, and the sampling points that do not fall within the pixels are interpolated using bilinear interpolation, thus allowing for any radius and any number of sampling points in the neighbourhood.

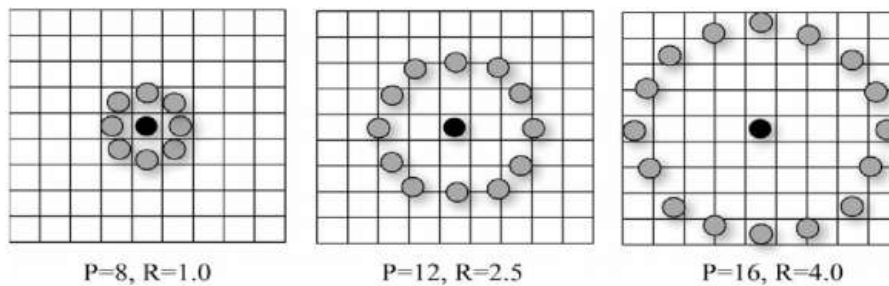


Figure III.7: LBP descriptors for different values of points ( $P$ ) and radius ( $R$ ).

For any radius and any number of sampling points in the neighbourhood, the local binary patterns code for a pixel located at the coordinate  $(x_c, y_c)$  can be defined as :

$$LBP_{N,R}(x,y) = \sum_{p=0}^{N-1} s(g_p - g_c)2^p \quad (\text{III.2})$$

### III.4.3.2 Efficient distance matching for LBP Descriptors

Given binary vectors for all features in the overlapped images, we can efficiently compare them. Unlike standard feature descriptors which undergo a large variety in descriptive power.

For two feature points,  $p_{ij}$  and  $p_{i'j'}$  from images  $i$  and  $i'$  respectively, we can compute the matching distance as [59]:

$$d_s(p_{ij}, p_{i'j'}) = d_{ham}(\hat{h}_{ij}, \hat{h}_{i'j'}) \quad (\text{III.3})$$

Where  $d_{ham}(a, b)$  is the Hamming distance between the two binary vectors  $a$  and  $b$ .

If two features are compared, small distance value context between them is a sign of good match ability. Although in the ideal case the binary vectors of the matched features should be completely coinciding, some relaxation should be made to avoid mismatching due the noise in the binary vectors. For that, we define features to have a potential to be matched if their matching distance is smaller than a threshold  $t_{ham}$ .

Instead of matching every feature point of the first image to every feature in the second image, some approaches were proposed for facilitating both the speed and the quality of the matching process. It takes much less time to perform the search of the binary vector that is close to a given one than to perform full searching the feature descriptor space. Moreover, using binary descriptors guarantees that the matched points describe the objects of the same classes, reducing the number of potential outliers.

### III.4.4 BRISK, and FREAK Based features matching

As binary visual descriptors, BRISK, and later FREAK were meant for fast matching, allowing tracking while the object was moving in front of the camera. Clearly they suit events where the object is still, and the camera is changing its position. As said before, binary descriptors computation requires less resources in terms of calculation power, and memory to store the resulting feature points. The matching phase provides another speed up if done using the Hamming distance.

The Hamming distance calculated between two binary string having the same length is the number of differing bits. The matching between two BRISK obtained descriptions can be achieved with a single instruction, the sum of the XOR operation between the two binary strings [40].

### III.4.5 Features tracking based matching

This approach of features matching using tracking is more widely used for image registration and video tracking applications, it is based on finding features in a set of overlapped images, then matching those features by finding a set of likely feature locations in every two successive images where the expected amount of motion and appearance deformation between them is expected to be small.

### III.4.5.1 KLT Standard Tracking Algorithm

KLT (Kanade-Lucas-Tomasi) algorithm was proposed by Kanade, Lucas in 1981 [62] and it was developed by Tomasi and Kanade [63, 64], this technique is largely used in computer vision for tracking features in a sequence of images. The KLT is based on extracting the features in the first image, then tracking these features in the sequence of images. With assumption that the camera moves slowly so that the change between images is small, so that the neighbour of a point  $f(x, y)^T$  tracked in the image  $I1$  is found in image  $I2$  by a simple translation  $d$ :

$$I_2 = I_1(x - dx, y - dy) + n(x, y) \quad (\text{III.4})$$

Where

$d(dx, dy)^T$  is the vector of translation between the image  $I1$  and the image  $I2$ .

$n(x, y)$  is the noise in the position  $(x, y)$ .

The estimation of the distance  $d$  is done while minimizing the quadratic error  $\varepsilon$  computed in neighbour window  $W$  according to an optimization criteria given by the following relation (III.5).

$$\varepsilon = \sum [I_1(f - d) - I_2(f)]^2 \omega(f) \quad (\text{III.5})$$

Where  $\omega(f)$  is a weighting function, generally  $\omega(f)$ , but it can also take a Gaussian form if we want to give more importance to the centre of the window. The development of Taylor series to the first order of the intensity function  $I1$  is given as follows (III.6):

$$I_1(p - d) = I_1(p) - JJ^T d \quad (\text{III.6})$$

Where  $J$  is the Jacobian matrix of  $I1$ ; with  $J = \left[ \frac{\partial I_1(p)}{\partial x} \quad \frac{\partial I_1(p)}{\partial y} \right]^T$ .

The development gives a solution of the form  $Ad = b$  with :

$$\begin{cases} A = \sum JJ^T W(p) \\ b = \sum_{P \in Q} [I_1(P) - I_2(p)] J(p) W(p) \end{cases} \quad (\text{III.7})$$

Having these equations, a least squares approach can be used to estimate  $d$ .

## III.5 Geometric transformations

### III.5.1 Geometric transformations Models

Given a set of point correspondences  $p_k \Rightarrow p_{k'}$ , for  $k = 0, 1, \dots, N$ , between two images,  $I(x, y)$  and  $I'(x, y)$ , a geometric transformation  $T$  can relate those images such that (III.8):

$$I'(x, y) = I(T(x, y)) \quad (\text{III.8})$$

The function  $T$  can be very complex; the image formation process likely yields images that vary non-linearly in their geometry (radial lens distortion, for example, varies non-linearly as a function of the radius from the centre of the image). We limit ourselves to the simpler case where  $T$  is modelled as a linear coordinate transformation.

### III.5.1.1 Isometric transformation

An isometric is a simple geometric transformation that preserves Euclidian distance. This means that after applying this transformation; the distance between two points in one image will be the same as the distance between their corresponding points in the mapped image. The same goes for the angles between lines and areas. The matrix of isometric transformation is composed only of 2D rotations and 2D translations and therefore has only 3 degrees of freedom [65].

✓ An isometric transformation can be written as follows (III.9):

$$x' = \begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix} * x \quad (\text{III.9})$$

Where:

$R$ : is a 2x2 rotation matrix.

$t$ : is a translation 2-vector.

$0^T$ : is a row of 2 zeros.

### III.5.1.2 Similarity transformation

The only difference between a similarity transform and an isometric transform is that the latter contains a factor called isotropic scaling which is invariant with respect to direction. This scale adds an additional degree of freedom so a similarity transform contains 4 degrees of freedom overall. Like with isometrics, angles are not affected by this transformation. The distance between points are no longer invariant, but the ratio of distances is preserved under similarity transformations since any scale change cancels out [66].

✓ A similarity transform can be written as (III.10):

$$x' = \begin{pmatrix} s * R & t \\ 0^T & 1 \end{pmatrix} * x \quad (\text{III.10})$$

Where:

$s$ : is a scalar and represents the isotropic scaling.

### III.5.1.3 Affine transformation

An affine transformation is similar to a similarity transform, but it is composed of two rotations angles and two non-isotropic scaling factors. Thus, affine contains two more

degrees of freedom than the similarity transformation; one for the angle specifying the scaling direction and one for the ratio of the scaling parameters. An affine transformation does not preserve the distance ratios or the angles between lines. However, after applying this geometric transformation to map two images, parallel lines in one image remain parallel in the mapped image, and the ratios of lengths of parallel line segments and areas are also preserved [21, 65, 66].

✓ An affine transformation can be written (III.11):

$$x' = \begin{pmatrix} A & t \\ 0^T & 1 \end{pmatrix} * x \quad (\text{III.11})$$

Where:

$A$  : is a 2x2 non-singular matrix.

$A$  can be decomposed as:

$$A = R(\theta) * R(\varphi)DR(\varphi) \quad (\text{III.12})$$

Where:

$R(\theta)$  and  $R(\varphi)$  are rotation matrices for  $\theta$  and  $\varphi$  respectively .

$D$  is a diagonal matrix.

$$D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \quad (\text{III.13})$$

Where:

$\lambda_1$  and  $\lambda_2$  can be considered as two scaling values.

The matrix  $A$  is thus a concatenation of a rotation by  $\varphi$ , a scaling by  $\lambda_1$  in the  $x$  direction, a scaling by  $\lambda_2$  in the  $y$  direction, a rotation back by  $-\varphi$  and then another rotation by  $\theta$ .

### III.5.1.4 Projective transformations

Projective transformations contain two more degrees of freedom than affine transformations, thus, the matrix contains nine elements. The form of the projective transformation matrix  $H$  determines the type of geometric transformation represented. With a rotation angle  $\theta$ , and making use of the "1" in the homogeneous coordinate, we can add translation [66].

$$H = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \quad (\text{III.14})$$

Multiply the rotation matrix by  $s$  to obtain scaling:

$$H = \begin{bmatrix} s * \cos(\theta) & -s * \sin(\theta) & t_x \\ s * \sin(\theta) & s * \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s * R & t \\ 0^T & 1 \end{bmatrix} \quad (\text{III.15})$$

Skewing can be introduced by multiplying the  $a$  and  $b$  parameters:

$$H = \begin{bmatrix} s * a * \cos(\theta) & -s * b * \sin(\theta) & t_x \\ s * a * \sin(\theta) & s * b * \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s * R & t \\ 0^T & 1 \end{bmatrix} \quad (\text{III.16})$$

While perspective is adjusted in the final row:

$$H = \begin{bmatrix} s * a * \cos(\theta) & -s * b * \sin(\theta) & t_x \\ s * a * \sin(\theta) & s * b * \cos(\theta) & t_y \\ p_1 & p_2 & 1 \end{bmatrix} \quad (\text{III.17})$$

In total, there are 8 parameters encoded in the H-matrix. Its elements are shown in equation (III.18):

$$H = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \quad (\text{III.18})$$

Since  $H$  operates on homogeneous coordinates, it is homogeneous itself (we can always divide  $H$  by a constant without changing its function). Linear transformation matrices can be combined. For example, say we want to rotate an image around its centre at  $(x_c; y_c)$ . We can express that operation as shifting the image upward and to the left, until its centre lies on the origin, rotating the image and then translating it back to its original position. A projective transformation can be written as:

$$x' = \begin{pmatrix} A & t \\ V^T & v \end{pmatrix} \quad (\text{III.19})$$

$$\text{Where: } V = (v_1 \ v_2)^T ; A = \begin{pmatrix} h_{00} & h_{01} \\ h_{10} & h_{11} \end{pmatrix} \text{ and } t = \begin{pmatrix} H_{02} \\ H_{12} \end{pmatrix}.$$

The main differences between affine and projective transformations are summarized in the [Table III.2](#) as follows [66]:

There are many situations in computer vision where estimating a one of the mentioned transformations may be required. In our case, for image mosaicing; we need a transformation model to project two overlapped images on each other to create an image mosaic,






Table III.2: Comparison between affine and projective transformations.

Affine Transformation	Projective Transformation
$V = 0$	$V = (v_1 \ v_2)^T$
Scaling factors $\lambda_1$ and $\lambda_1$ are the same everywhere in the plane.	Scaling factors $\lambda_1$ and $\lambda_1$ vary with the position in the image.
The orientation of a transformed line depends only on the orientation of the original line.	The position of the original line on the plane also effects the transformed line's orientation.
Minimum of 3 pairs of corresponding points are needed to estimate the parameters.	Minimum of 4 pairs of corresponding points are needed to estimate the parameters.

therefore; the projective transformation (homography) is the most suitable model for our purpose.

Table III.3 illustrates examples of the most important types of geometric transformation, those transformations are used in many computer vision applications.

Table III.3: Illustration of the projective linear group and its three subgroups [21].

Name	Symbolic Matrix	Example Matrix	Example Image
Original	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
Isometric	$\begin{pmatrix} \cos(\theta) & \sin(\theta) & t_x \\ -\sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0.707 & 0.707 & 0 \\ -0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
Similarity	$\begin{pmatrix} s * \cos(\theta) & s * \sin(\theta) & t_x \\ -s * \sin(\theta) & s * \cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1.5 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
Affine	$\begin{pmatrix} \alpha_{00} & \alpha_{01} & t_x \\ \alpha_{10} & \alpha_{11} & t_y \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1.5 & 0 & 0 \\ 1 & 1.5 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
Projective	$\begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0.005 & 1 \end{pmatrix}$	



### III.5.2 Homography estimation

The estimation of the homography between two views is a key step in many applications involving multiple view geometry. The homography exists between two views between projections of points on a 3D plane. A homography exists also between projections of all points if the cameras have purely rotational motion.

A number of algorithms have been proposed for the estimation of the homography relation between two images of a planar scene. They use features or primitives ranging from simple points to complex ones like non-parametric curves. Different algorithms make different assumptions on the imaging setup and what is known about them. After the matching problem is completed, a homography transformation then is needed to be determined in order to map one image to the other image by establishing point-by-point correspondence between the two images. Homography transformation is a mathematical concept used in projective geometry to describe a relationship between two planes, such that any point on one plane corresponds to one point in the other plane [67].

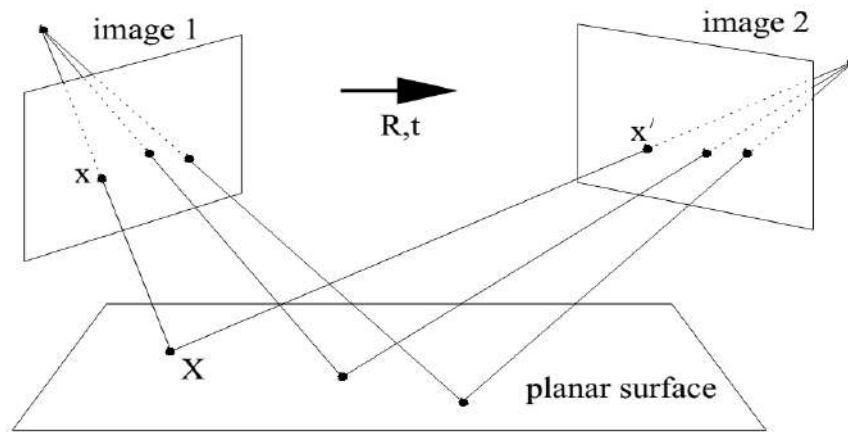


Figure III.8: Two images of a planar scene connected by a rotation and translation.

This transformation is valid in the following three cases[68]:

- Images of a plane viewed under arbitrary camera motion.
- Almost-planar image : A common situation when the UAV flies at high altitude.
- Images obtained by a rotation around the optical centre of the camera.

Homography can be defined as an invertible application of the projective space  $P^2$  into  $P^2$  that applies lines into lines. Some basic properties of the homography are the following [21]:

1. Any homography can be represented as a linear and invertible transformation in homogeneous coordinates:

$$\begin{bmatrix} kx' \\ ky' \\ k \end{bmatrix} = \underbrace{\begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix}}_H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (\text{III.20})$$

2. A homography between two planes is a linear transformation between three-dimensional homogeneous vectors  $y$ , represented by the  $3 \times 3$   $H$  matrix such as  $y = Hx$
3. Given the homogeneous skill of the homography  $H$ , it can be multiplied by an arbitrary constant  $k \neq 0$  and represent the same transformation. This means that the matrix  $H$  is constrained by eight independent parameters and a scale factor.

The homography that relates two given images is computed from sets of matches between point features. Depending on the scene characteristics, the homography computation from these matches could become a hard problem. A careful analysis points out two factors that may significantly increase the complexity of the computation, mainly when the UAV flies at altitudes of the same order of other elements on the ground (buildings, trees...etc.):

- In 3D scenes, the parallax effect will increase, and the planarity assumption may not hold.
- Depending on the frame-rate and the vehicle motion, the overlap between images in the sequence is sometimes small. This generates a non-uniform distribution of the features along the images.

### III.5.2.1 Basic DLT algorithm

Estimating homography from pairs of images has been studied quite extensively in the literature. The Direct Linear Transform (DLT) algorithm is a simple algorithm used to solve for the homography matrix  $H$  given a sufficient set of point correspondences[69]. Letting a point and its correspondent that can be written respectively as  $(x, y, 1)$ ;  $(u, v, 1)$ , we can get:

$$-h_4x - h_5y - h_6 + (h_7x + h_8y + h_9)v = 0 \quad (\text{III.21})$$

$$-h_1x - h_2y - h_3 + (h_7x + h_8y + h_9)u = 0 \quad (\text{III.22})$$

These two equations; (III.21) and (III.22); can be written in matrix form as:

$$Ah = 0 \quad (\text{III.23})$$

where:

$$A = \begin{pmatrix} -x & -y & -1 & 0 & 0 & 0 & ux & uy & u \\ 0 & 0 & 0 & -x & -y & -1 & vx & vy & v \end{pmatrix}$$

and

$$h = (h_1 \ h_2 \ h_3 \ h_4 \ h_5 \ h_6 \ h_7 \ h_8 \ h_9)^T$$

Since  $N$  pairs of correspondences provide  $2N$  equations, 4 pairs are sufficient to solve for the 8 degrees of freedom of  $H$ . For  $N > 4$  pairs of points, this equation will not have an exact solution. In this case, a solution which minimizes the algebraic residuals,  $r = AH$ , in a least-squares sense may be obtained, by taking the singular vector corresponding to the

smallest singular value [64]. The Estimation of homography matrix using the minimum number of correspondences is useful for many applications [66, 70].

After estimating homography matrix from the set of found matched key-points using the mentioned techniques above, this matrix may be incorrect and would introduce excessive error; because of false associations problem, one most commonly used method to correct homography is random sample consensus (RANSAC), a general technique to take data, fit an initial model made of a random sample of the provided data, test for consensus among the rest of the data (i.e., find data that is 'close' to the model), produce a new model using the data in consensus, and if the stopping conditions are not met, start over [71].

### III.5.2.2 RANSAC

Homography estimation using RANSAC (Random Sample Consensus) is a key step in feature matching as it improves the stability of image registration. It can estimate the parameters of homography matrix with a high degree of accuracy. In this method, for a number of iterations, a random sample of 4 correspondences is selected and a homography  $H$  is computed from those four correspondences by the direct method.

Each other correspondence is then classified as an inlier or outlier depending on its concurrence with  $H$ . After all of the iterations are done, the iteration that contained the largest number of inliers is selected.  $H$  can then be recomputed from all of the correspondences that were considered as inliers in that iteration [72].

One important issue when applying the RANSAC algorithm described above is to decide how to classify correspondences as inliers or outliers. Statistically speaking, the goal is to assign a distance threshold,  $t$ , (between  $x'$  and  $Hx$  for example), such that with a probability the point is an inlier [65].

## III.6 Image projection "Warping"

When the homography is computed, we can transform an image to the coordinate frame of the other one. And that can be done in two different ways:

### III.6.1 Forward warping

Here, we find the coordinates of pixels of the second image in the frame of the first image using the forward application of the homography.

$$x' = H * x \tag{III.24}$$

### III.6.2 Backward warping

Here, we find the coordinates of pixels of the first image in the frame of the second image with the backward application of the homography.

$$x = H^{-1} * x' \tag{III.25}$$

### III.7 Template object tracking

Template matching is conceptually a simple process. We need to match a template to an image, where the template is a sub-image that contains the shape we are trying to find. Accordingly, we centre the template on an image point and count up how many points in the template matched those in the image. The procedure is repeated for the entire image and the point which led to the best match, the maximum count, is deemed to be the point where the shape (given by the template) lies within the image. Consider that we want to find the template of Figure III.9a in the image of Figure III.9b. The template is first positioned at the origin and then matched with the image to give a count which reflects how well the template matched that part of the image at that position. The count of matching pixels is increased by one for each point where the brightness of the template matches the brightness of the image. The points in the image are matched with those in the template, and the sum is of the number of matching points as opposed to the weighted sum of image data. The best match is when the template is placed at the position where the rectangle is matched to itself. Obviously, this process can be generalized to find, for

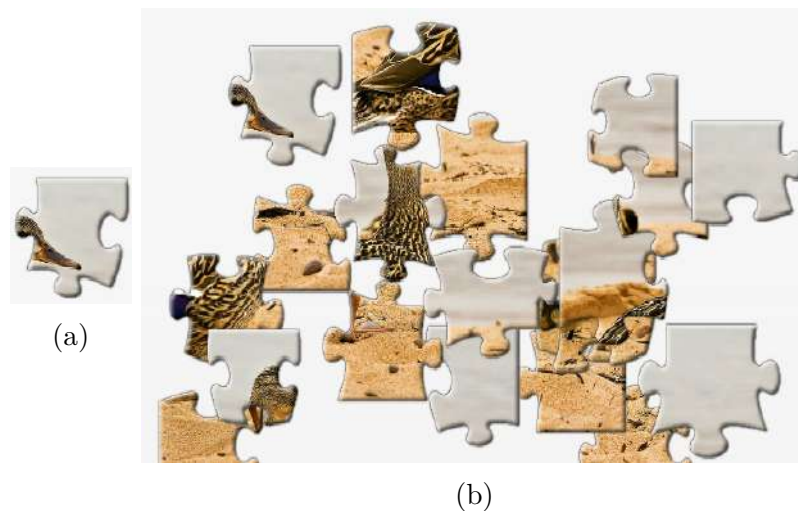


Figure III.9: Illustrating template matching.

example, templates of different size or orientation. In these cases, we have to try all the templates (at expected rotation and size) to determine the best match. Formally, template matching can be defined as a method of parameter estimation. The parameters define the position (and pose) of the template. We can define a template as a discrete function  $T_{x,y}$ . This function takes values in a window. That is, the coordinates of the points  $(x,y) \in W$ . For example, for a 2x2 template, we have the set of points  $W = (0,0), (0,1), (1,0), (1,1)$  [6].

One strategy to determine the moving object, by background removal, and to then track points in the moving object. Another strategy is to determine interest points, such as corners, and to then track the appearance of these points in successive frames (and there is natural debate on which features to track ).

Indeed, Several methods are proposed among which extracting and using the key points of the image is the one mostly applied. Here, the classical detectors (Harris & FAST) is applied and the binary descriptors and indexes of each key point found thereof are introduced by FREAK, inspired by human eye, applied in comparing and recognizing the objects in each frame. Based on this the accuracy and speed of the recognition increase with less memory space needed for implementation [73].

### III.8 Conclusion

In this chapter, we presented a very wide application of image registration which is image mosaicing. We stated the various necessary steps to create an image mosaic in which different approaches were discussed. The image mosaic is obtained by fusing the overlapped split images of a large split, but the most important task before mosaicing is to find out the overlap region or correspondence points between the split images. The overlap region is found by matching similar features present in the given split images. Even if there are many approaches for achieving image mosaicing, we can hardly tell which is the easy way or the best way to achieve it.

In the next chapter we will present our strategy to approach an image mosaiced, then we will visualise the simulation results of this strategy.

# *Results and Evaluation*

---

IV.1 Introduction . . . . .	44
IV.2 Work environment . . . . .	44
IV.3 Evaluation and results . . . . .	45
IV.4 Conclusion . . . . .	56

---

## **IV.1 Introduction**

In this chapter we are going to present the results obtained by our application (in MATLAB platform and LabVIEW platform), with a quantitative evaluation of these performances, associated by a comparison between deference methods.

## **IV.2 Work environment**

The algorithm proposed here has been implemented in Matlab R2014b and has been executed in system with configuration an Intel®Core™ i5 CPU powered PC equipped with 8GB of RAM.

In fact, we choose as platform of development the tool Matlab® as the implementation of the various stages, and this due to the ease of implementation of the operations of image processing, generally, because Matlab software is suitable for the development of complex image processing algorithms such as image mosaicing algorithm.

Furthermore, we convert to the LabVIEW platform which constitutes a graphical programming environment that allows one to design (Front Panel (graphical user interface)) and analyse a DSP system in a shorter time as compared to text-based programming environments. LabVIEW graphical programs are called virtual instruments (VIs). VIs run based on the concept of data flow programming [74]. Expecting to implement the image mosaicing algorithm on the FPGA, because with the NI LabVIEW FPGA Module, we can take advantage of the parallel processing capabilities of FPGAs without being an expert in VHDL or Verilog.

## IV.3 Evaluation and results

### IV.3.1 Image Mosaicing

The proposed mosaicing algorithm is based on Harris detector/ FREAK descriptor for a robust matching followed by estimating the homography for geometric registration. Figure IV.1 shows the flow chart for the essential steps of the mosaicing algorithm applied.

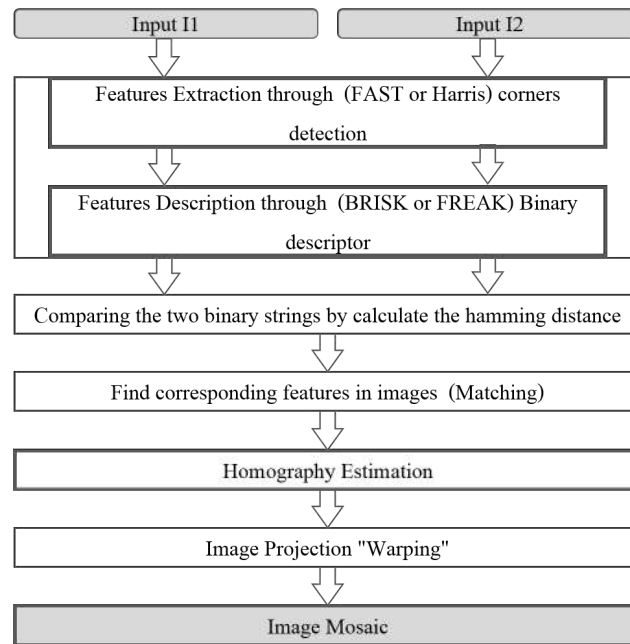


Figure IV.1: Flowchart for the proposed image mosaicing.

#### IV.3.1.1 Matlab results

To visualize(display) the results of each stages of our method, we are going to work on a typical sample, which represents two digital images of the laboratory Figure IV.2, we notice these two image pairs, the view of interest has significantly changed but some of the local details are even though there are some of the details are significantly different also.

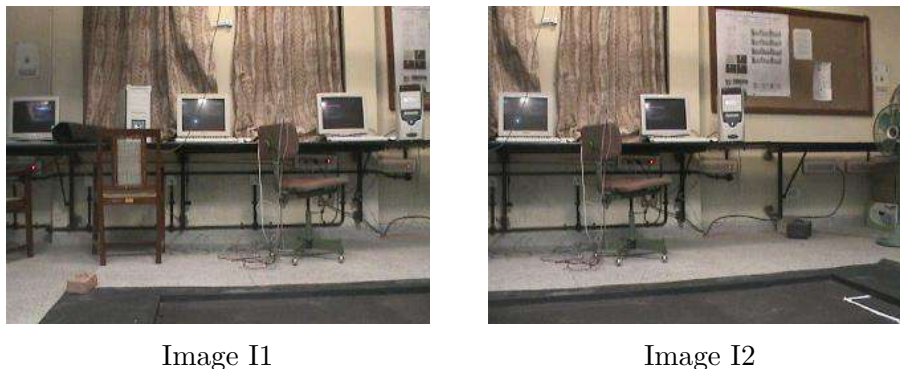


Figure IV.2: The inputs images.

### ① Features extraction:

In [Figure IV.3](#) we can observe that a lot of points features are detected, such as the chair there, and some features at the computer screen there...etc. We can see similar features and actually now we can notice that it found features repeated in both images. In general, this Harris detector output we are looking for which we need to achieve the next stage, which is matching I1 features (corners) with correspondence features (corners) detected in I2.

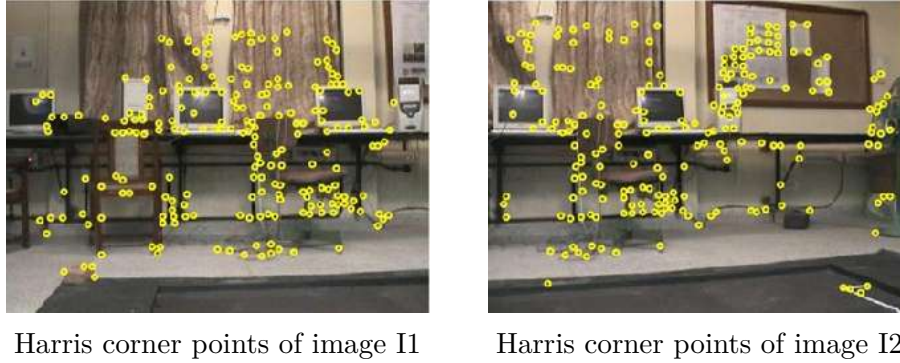


Figure IV.3: Features detection using Harris detector.

The main effective parameter on Harris Corner Detector algorithm is the threshold value, [Table IV.1](#) illustrates the effect of threshold value on the performance of the detector.

Table IV.1: The effect of threshold value on the performance of the detector.

Threshold Value	Features detected in I1	Time (s)	Features detected in I2	Time (s)
1000	260	0.1021	304	0.0471
2000	167	0.0983	183	0.0468
3000	114	0.1364	134	0.0631

From the above results, we can notice that :

- \* Harris corner algorithm fully utilizes the characteristics of the corner.
- \* Harris algorithm gives efficient number of features in a short time.
- \* Harris detector provides distinctive corners features with high repeatability.

### ② Features matching:

#### i. Correlation-based matching:

For correlation-based matching, we measure the sum of absolute difference (SAD) correlation to select the best corner matches, [Figure IV.4](#) shows the output of this operations.



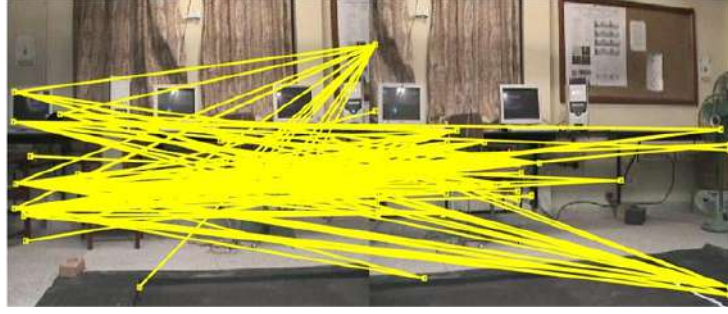


Figure IV.4: Features matching using correlation.

We can observe from the correlation-based matching, even though it simple to calculated but there are a lot of false associations occur in the matching, which effect badly in next stage (homography estimation). The correlation window is the effective parameter in this algorithm (see [Table IV.2](#) bellow).

Table IV.2: The effect of the correlation window.

Correlation window	Matched features	Correct matches	Time (s)
3x3	260	8	0.6028
7x7	167	18	0.9019
9x9	114	27	1.0188

- \* We note that the larger window size has increased the number of Correct matching points. On the other hand, the process become a slower.
- \* To eliminate the false association, we mast to use the RANSAC algorithm, or we should use descriptors-based features matching.

### ii.Descriptor-based matching:

[Figure IV.5](#) shows the output of binary descriptor matching to associate the corners which are detected in previous stage by using Harris corner detection.

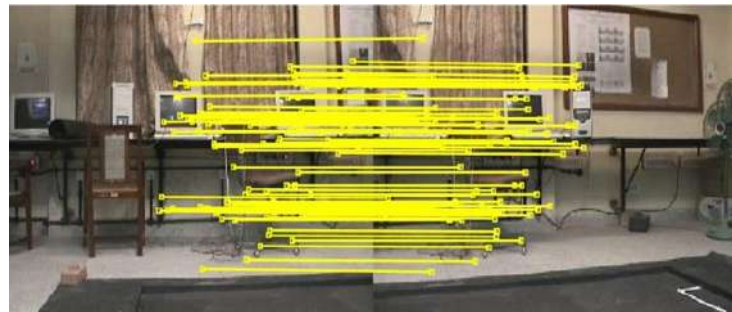


Figure IV.5: Features matching using BRISK descriptor.

When we compare the output results in [Figure IV.4](#) with the results of [Figure IV.5](#) , we can easily judge that classical Harris features detector associated with BRISK descriptors give better matching result than correlation.

- \* This method has a significant resistance against changes in scene, brightness, rotations and low scale change.

### ③ Homography estimation:

By using the previous images and by assuming that the geometrical transformation which connects them is projective, the homography matrix will not have an exact solution. Because the number pairs of point  $N$  is greater than four ( $N > 4$ ). Thus, we solve this by using Singular Value Decomposition (SVD); which obtain the most appropriate matrix. this estimated homography matrix allowed us to transform image  $I_2$  to new image  $I_2'$ :

$$H = \begin{bmatrix} 0.9094 & 0.0010 & 121.4578 \\ -0.0377 & 0.9663 & 3.5759 \\ -0.0003 & -0.0000 & 1.0000 \end{bmatrix}$$

### ④ Image warping:

The projection of the two images on each other according to a mapping between source image  $I_1$  and destination image  $I_2$  give us the final image mosaic :



Figure IV.6: Image mosaic by backward warping.

- \* Our algorithm provides seamless images mosaic with high resolution and extended field of view.

#### IV.3.1.2 LabVIEW results

To build a single image with a wide view from two overlapped images, we design a LabVIEW VI programme which allowed us to obtain a mosaiced image. To being able to test our system, and verify some results given in this document, we design an execution interface on LabVIEW. this allowed the user to upload two overlapped images furthermore, we give the user the ability to select the type of corners detector (FAST or Harris) and its threshold value. also, the user will be able to choose between FREAK or BRISK descriptor for the matching. as a result, the mosaiced image, the detected corners in both images, and their matching is shown to the user, with some essential information:

- Number of features detected in both images.
- Number of matched points.
- Elements of the transformation matrix (Homography matrix).

The following figures are some screen shots of our LabVIEW interface (Front Panel):

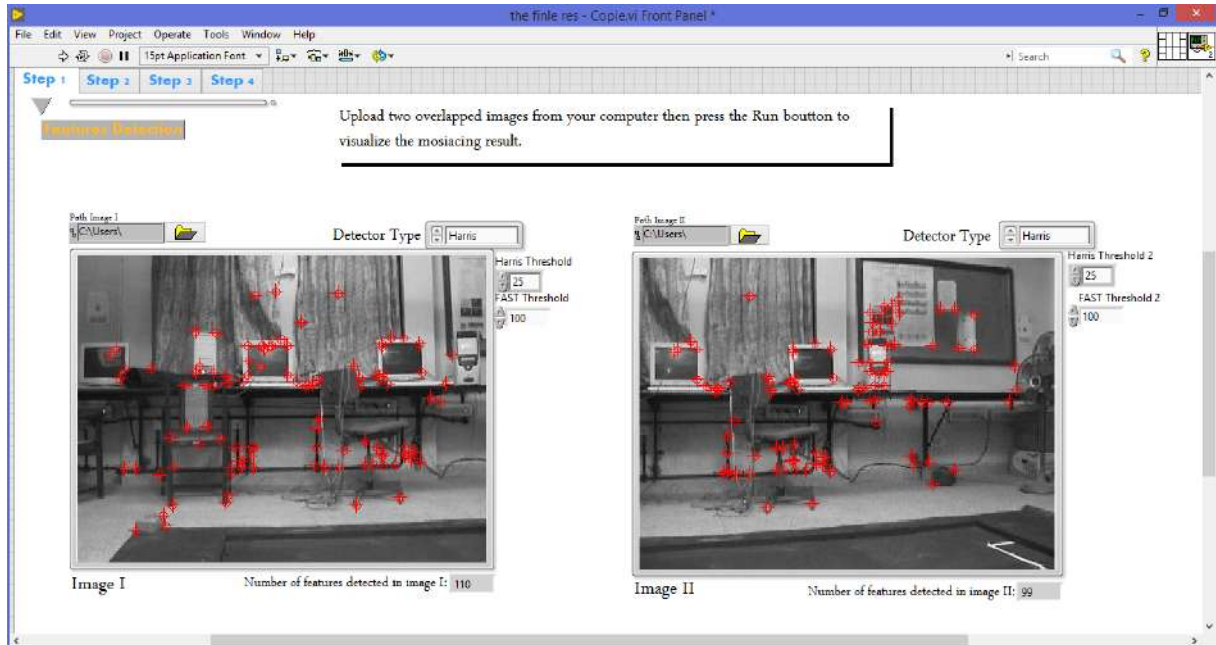


Figure IV.7: Page 1: displaying Features detect in both images.

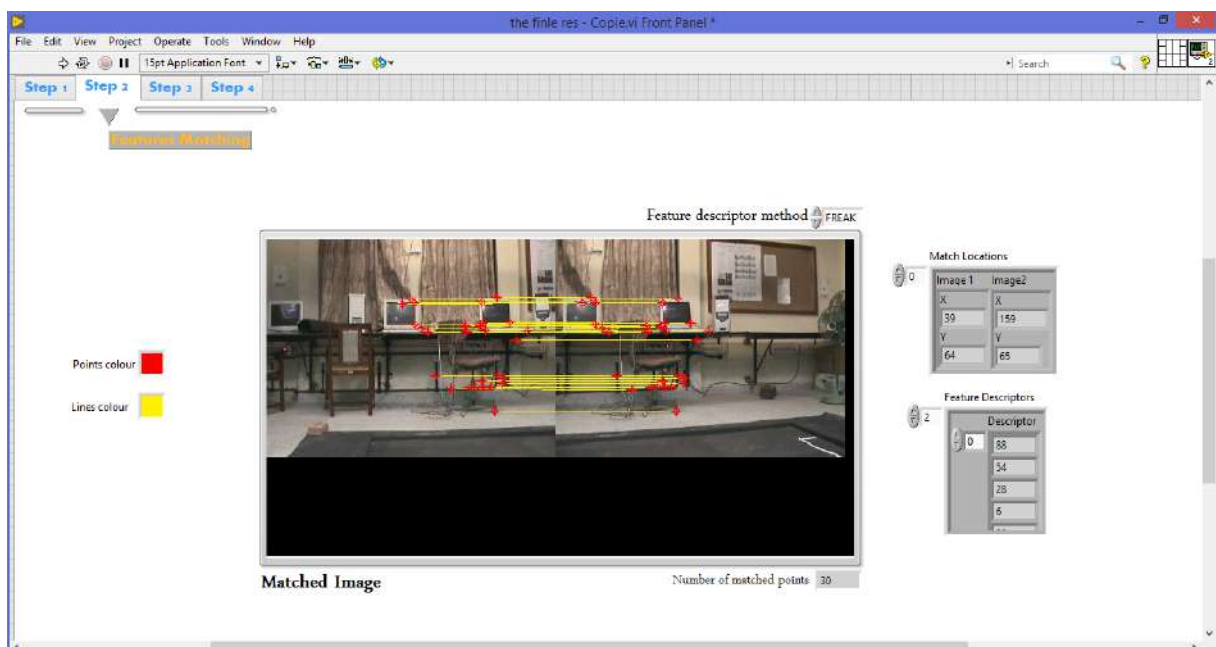


Figure IV.8: Page 2: displaying the features matched between images.

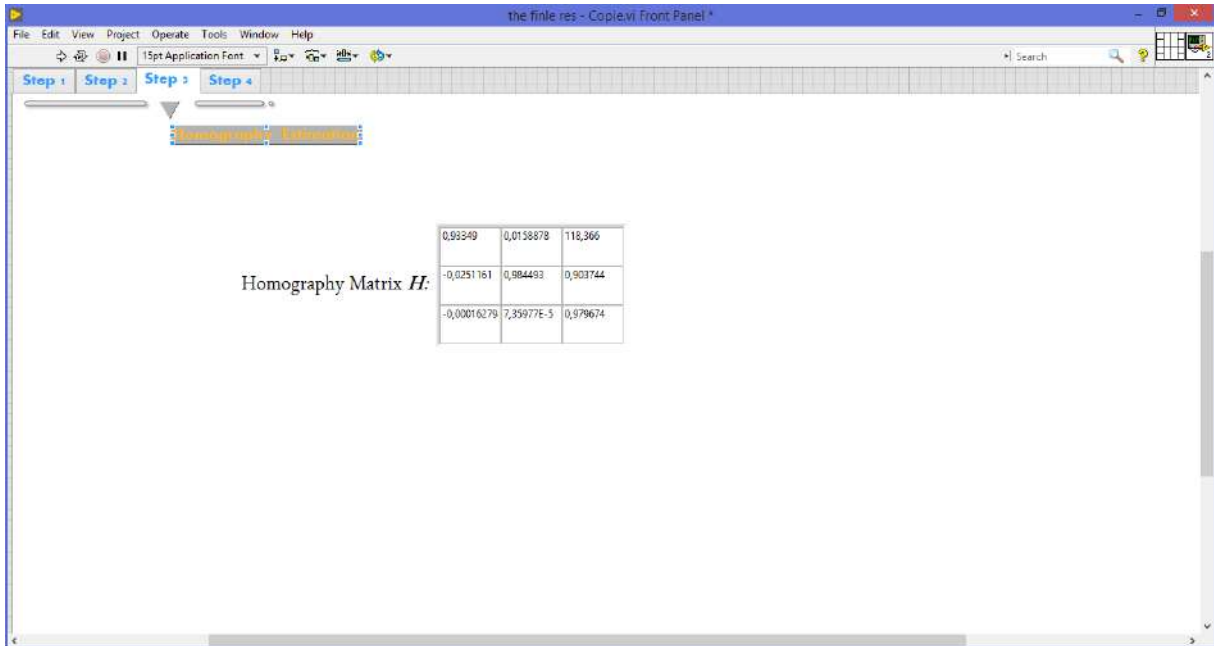


Figure IV.9: Page 3: displaying the estimated matrix

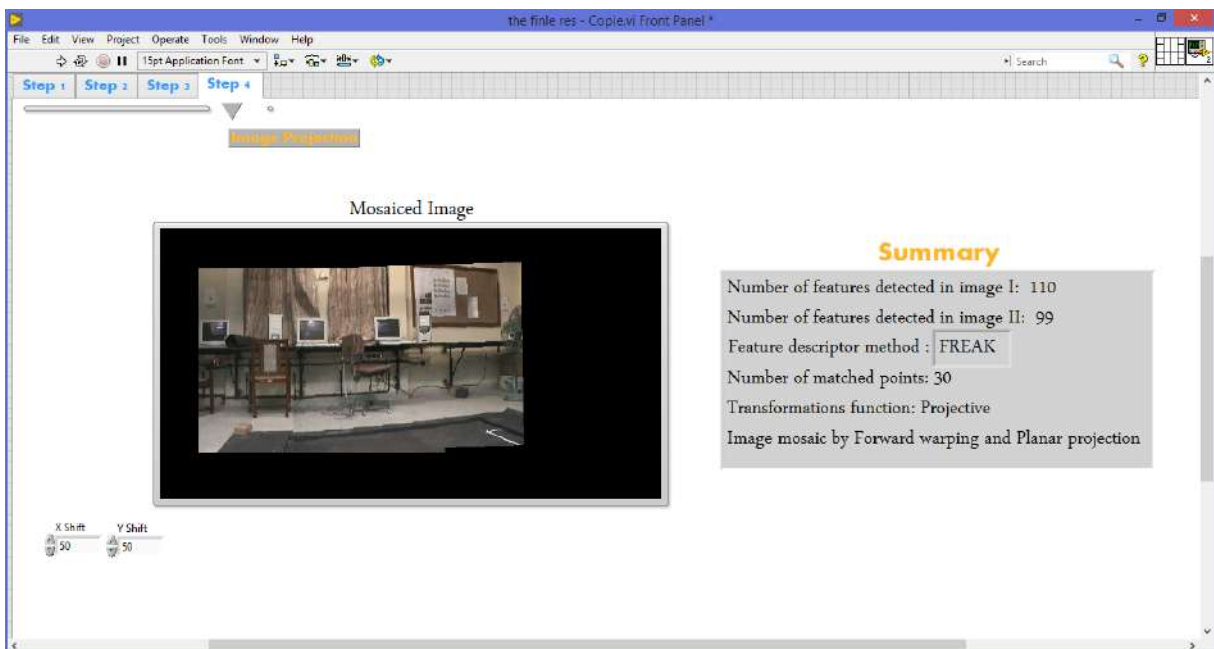


Figure IV.10: Page 4: displaying the mosaiced image and the summary.

After all, for testing the results of our LabVIEW application, we are going to present in the flowing tables different categories of images sources outputs: satellite, aerial images (from UAV) and medical images.

Table IV.3: Testing on Outdoor/Indoor phone images



Result 1: tow images of the entrance of NTIC faculty mosaig



Result 2: panoramic mosaig of three images

Table IV.4: Testing on Aerial and satellite images



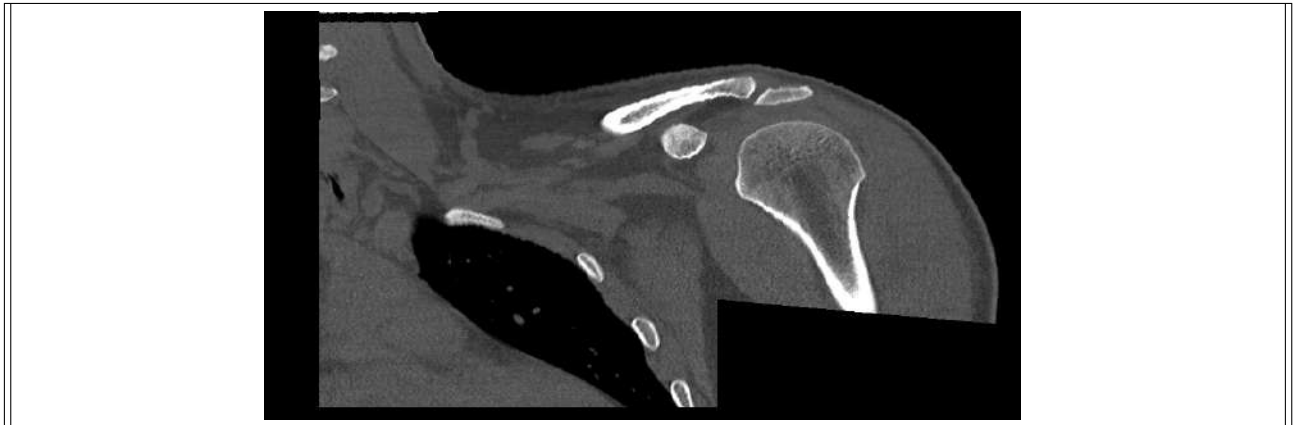
Result 3: mosaicing of aerial images <sup>1</sup> (from UAV)


Result 4: mosaicing of three satellite images <sup>2</sup>

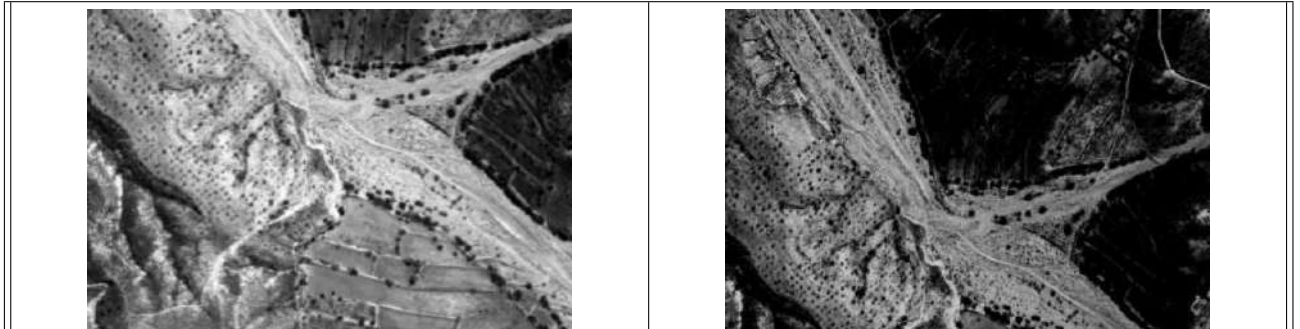
<sup>1</sup>sequence of video images filmed by drone [https://www.youtube.com/watch?v=L\\_dPBK2tAbY](https://www.youtube.com/watch?v=L_dPBK2tAbY)

<sup>2</sup>The images are extracted from *hamza 2012* [69]

Table IV.5: Testing on remote sensing and medical imaging



Result 5: image mosaicing in medical imaging<sup>3</sup>(CT scan)



Result 6: image mosaicing in Remote Sensing imaging<sup>2</sup>

<sup>3</sup>Computed Tomography CT scan imaging download from:<http://okradiologygroup.com/ct/>

<sup>2</sup>The images are extracted from *hamza 2012* [69]

\* The performance and efficiency of our proposed algorithm were validated using different data set, these data set are collected using different camera types in different conditions.

\* On these tests, our application gives generally very satisfying results.

### IV.3.2 Template object tracking

Tracking of an object pre-selected by the user or recognizing the specific object of interest and its tracking can be applied in public transportation, traffic, military and rescue systems etc. In recent decades several studies are conducted on image processing for object recognizing and tracking through different scientific and experimental methods.

Our strategy to achieve this algorithm firstly, we pass through the same three stages (feature extraction and feature matching then the homography estimation) ,then we multiple the four corners coordinates of the template image with the homography matrix , then we obtain new four coordinates which represent the corners of the rectangular target tracked. [Figure IV.11](#) shows the flow chart for the essential steps of the object tracking algorithm applied.

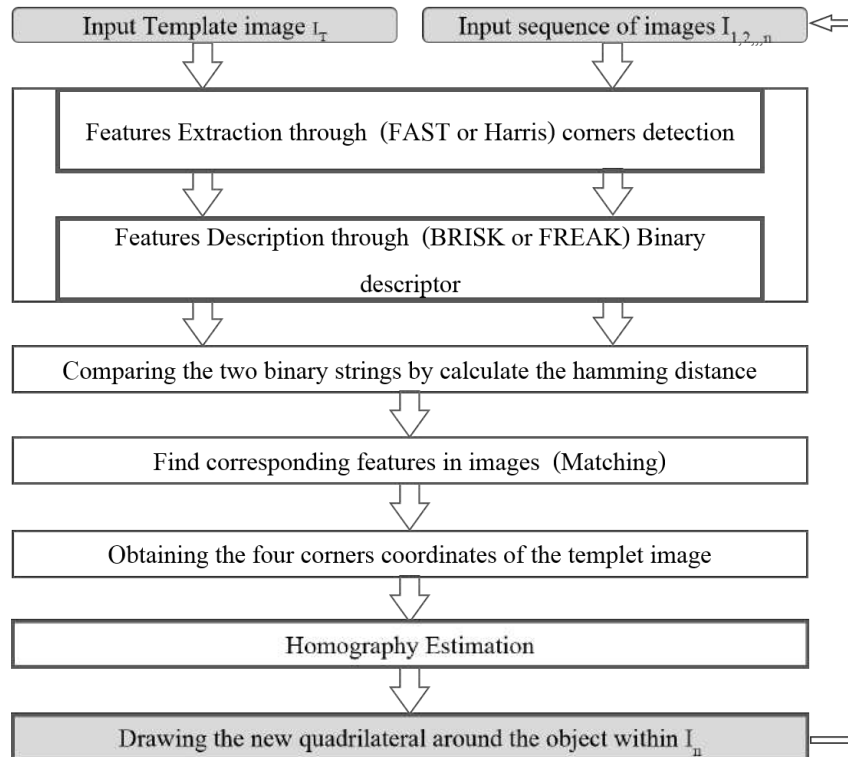
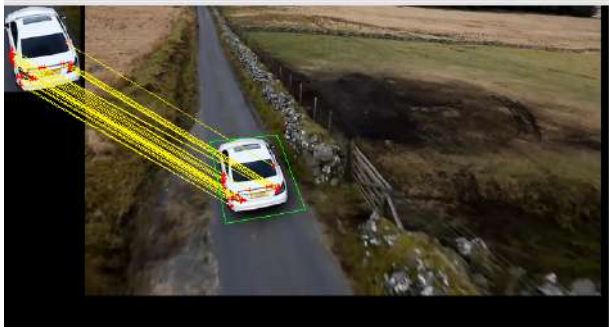
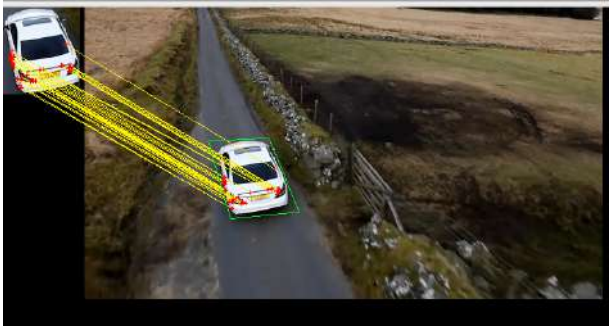
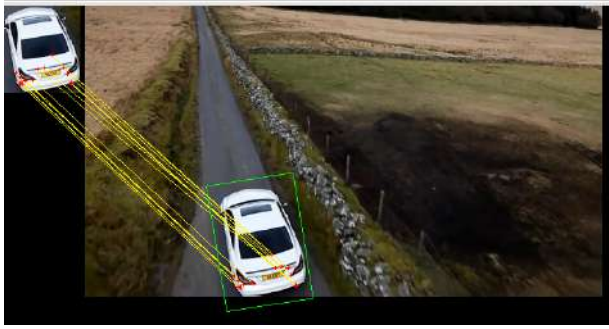
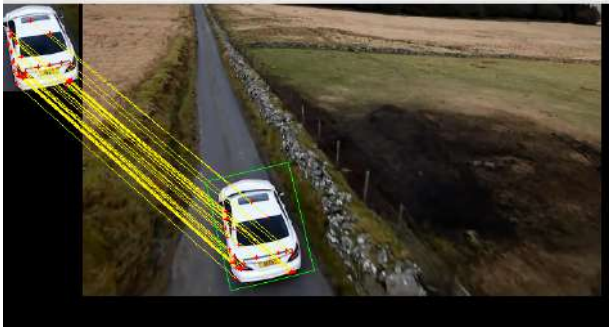
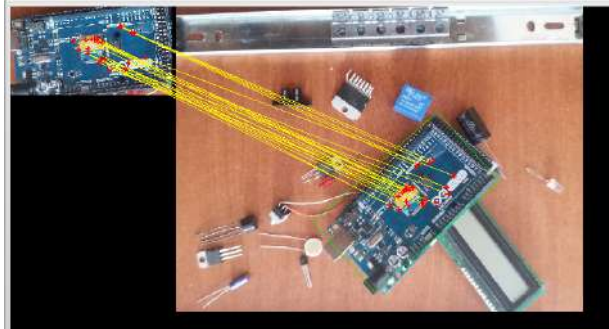
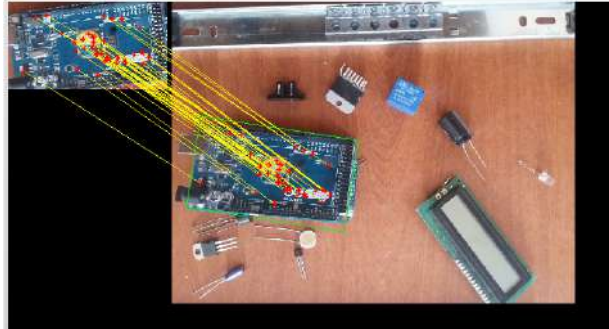
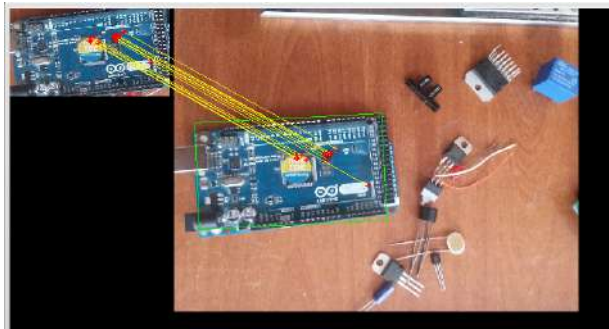
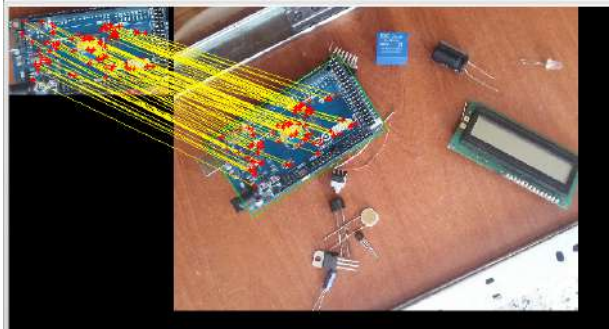


Figure IV.11: The method proposed for object tracking flowchart.

Some results of this algorithm are shown next in [Table IV.6](#)



Table IV.6: Template object tracking results

	
	
Result 7 <sup>4</sup> : Car tracking (detected object within green trapezium).	
	
	
Result 8:object (Arduino) detection	

Key points-based tracking is illustrated in which [Table IV.6](#) shows the result of tracking a car moving in a sequence of video (filming by UAV) images <sup>4</sup> and shows also the results of object (Arduino) detection and tracking. In both results, of which four are

<sup>4</sup>sequence of images captured from video, link: <https://www.youtube.com/watch?v=X792Jf2HGFE>

shown here. A green trapezium has been drawn around the template target in each frame and the parameters of this trapezium are delivered by the tracking algorithm. As the object or camera moves, the position of the rectangle is centered on the car and the width and angles of the rectangle changes as the car's motion. From this, we can then analyse the target and their motion, we can conclude this algorithms could be useful in public transportation, traffic, military and robotic domains.

## IV.4 Conclusion

In this chapter, we displayed results of our execution LabVIEW interface mosaicing algorithm based on Harris detector/ FREAK descriptor which gives better robust matching than the correlation method, applied to various categories of images. The visual tests were very satisfactory.

The performance of an image mosaicing algorithm depends mainly on the performance of the first and the second stages. Those stages are features detection and matching, because if features are extracted using a robust detector, and matching guarantees that the points which represent the same element of the scene in both images are correctly matched, the work which remains to create the mosaiced image is only finding an image transformation model (Homography matrix), by which overlapped images can be warped on each other.

The template object tracking algorithms described in this chapter can detect and track any kind of object, at various distances even if the object is moved, rotated or flipped. And it can be useful in many domains such as surveillance, traffic, military, biometry and robotics etc.

Both of our VI executable for image mosaicing and object tracking are based on points features, Those VIs give good outputs results with different images from different sources in many domains.

# *General Conclusion*

The work presented in this memory deals with the extraction of geometry information from a sequence of images. The problem is decomposed into a number of tasks; each task is being performed with existing techniques and combined to create an image mosaic or to follow object in a scene. Within our project, we went through literature review about the various classifications of features and approaches developed during the last years for their detection, and also; we gave an outline on some performed works.

For most of computer vision applications, points based features are the ideal image primitives, because their coordinates can be used directly to estimate certain transformation between overlapped images, therefore; we discussed in the second section of our report; the famous types of classical and modern key points detector, in which we saw the advantages and drawbacks of each detector.

Following some procedures which are illustrated in the third chapter, we can successfully do images mosaicing or track moving object with two images. In our results, due to its high repeatability, the Harris Corners Detector was used to detect massive sparse points of corner features in overlapped images, then; sum of absolute difference (SAD) correlation measure was used to select the best corner matches. But so far these correspondences are redundant with errors; therefore, imposing some constraints for eliminating the wrong matches were required, that is why we have proposed to use other matching technique; which is based on creating binary descriptors around the detected corners in both images, and calculating distance measure between these descriptors to choose the best matches. The second matching approach produced better result compared to the first one, thus, it was used to estimate the homography matrix from a set of produced inliers. To create an image mosaic; backward image warping technique was used to project images in a simple planar manifold, and for object tracking; four corners around the tracked object were selected and linked at each new overlapped image; with a condition that the new view includes corners of the tracked trick.

It was proposed to implement our algorithms on embedded system (FPGA kit), we started working to implement our algorithms with VHDL code on Xilinx kit, but we faced problems of image acquisition because of the limited sources of the available kits. We also

worked on implementing our algorithms on FPGA through LabVIEW tool that is why; we implemented all stages of our application on LabVIEW; but unfortunately; we missed hardware interface to allow charging our codes from LabVIEW to Xilinx kit.

Our applications were tested in Matlab platform, and then they were implemented on LabVIEW tool, the efficiency of the obtained results is verified by using different types of overlapped images. Finally, an execution interface was constructed on LabVIEW, this allowed the user to upload overlapped images and get the result with one order. The recommendations that can be made for the future investigation, is to use other kinds of features detectors, such as SIFT or SURF, and also other techniques for features matching. We also recommend the generalization of our algorithm to create a video from a sequence of overlapped images and to track certain object in that video.

# Bibliography

- [1] Michael S Mahoney. The history of computing in the history of technology. *Annals of the History of Computing*, 10(2):113–125, 1988.
- [2] Mahesh K Prasanna and Shantharama C Rai. Image processing algorithms-a comprehensive study. *International Journal of Advanced Computer Research*, 4(2):532, 2014.
- [3] Marcin Gabryel and Robertas Damaševičius. The image classification with different types of image features. In *International Conference on Artificial Intelligence and Soft Computing*, pages 497–506. Springer, 2017.
- [4] Hemlata Joshi and Mr Khomlal Sinha. A survey on image mosaicing techniques. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 2(2):pp–365, 2013.
- [5] Heung-Yeung Shum and Richard Szeliski. Panoramic image mosaics. Technical report, Citeseer, 1997.
- [6] Mark S. Nixon and Alberto S. Aguado. *Feature Extraction & Image Processing for Computer Vision*. 2012. ISBN 9780123965493. doi: 10.1016/B978-0-12-396549-3.00001-X. URL <http://www.sciencedirect.com/science/article/pii/B978012396549300001X>.
- [7] Guillaume Gales. Detection et mise en correspondance de points d interet. 2007.
- [8] Gaurav Kumar and Pradeep Kumar Bhatia. A detailed review of feature extraction in image processing systems. *International Conference on Advanced Computing and Communication Technologies, ACCT*, (February):5–12, 2014. ISSN 23270659. doi: 10.1109/ACCT.2014.74.
- [9] J. Pi, Y., Liao, W., Liu, M., and Lu. *Pattern Recognition Techniques, Technology and Applications*. 2008. ISBN 978-953-7619-24-4. doi: 10.5772/90. URL [http://www.intechopen.com/books/pattern\\_recognition\\_techniques\\_technology\\_and\\_applications](http://www.intechopen.com/books/pattern_recognition_techniques_technology_and_applications).

- [10] R.S. Choras. Image feature extraction techniques and their applications for CBIR and biometrics systems. *International Journal of Biology and Biomedical Engineering*, 1(1):6–16, 2007. URL <http://www.naun.org/journals/bio/bio-2.pdf>.
- [11] Dong Ping Tian. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 8(4):385–395, 2013. ISSN 19750080. doi: 10.1109/HIS.2012.6421310.
- [12] M Kunaver and J F Tasic. Image feature extraction-an overview. *The International Conference on Computer as a Tool 2005 EUROCON 2005*, (February):183–186, 2005. doi: 10.1109/EURCON.2005.1629889. URL [http://ieeexplore.ieee.org/xpls/abs/\\_all.jsp?arnumber=1629889](http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=1629889).
- [13] Zhonglong Zheng, Jie Yang, and Limin Yang. A robust method for eye features extraction on color image. *Pattern Recognition Letters*, 26(14):2252–2261, 2005. ISSN 01678655. doi: 10.1016/j.patrec.2005.03.033.
- [14] Nikhil R. Pal and Sankar K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993. ISSN 00313203. doi: 10.1016/0031-3203(93)90135-J.
- [15] Haikel Salem Alhichri and Mohamed Kamel. Virtual circles: a new set of features for fast image registration. *Pattern Recognition Letters*, 24(9-10):1181–1190, 2003.
- [16] John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. ISSN 01628828. doi: 10.1109/TPAMI.1986.4767851.
- [17] D. Marr and E. Hildreth. Theory of Edge Detection. *Proceedings of the Royal Society B: Biological Sciences*, 207(1167):187–217, 1980. ISSN 0962-8452. doi: 10.1098/rspb.1980.0020. URL <http://rspb.royalsocietypublishing.org/cgi/doi/10.1098/rspb.1980.0020>.
- [18] Hannes Schulz, Weichao Liu, Jörg Stückler, and Sven Behnke. Line Structure-based Localization for Soccer Robots. *4th Workshop on Humanoid Soccer Robots at International Conference on Humanoid Robots (Humanoids)*, (Humanoids):73–78, 2009. URL [http://www.ais.uni-bonn.de/papers/HSR09/\\_Schulz\\_/\\_Localization.pdf](http://www.ais.uni-bonn.de/papers/HSR09/_Schulz_/_Localization.pdf).
- [19] A. Ardeshir Goshtasby. *2-D and 3-D Image Registration*, volume 26. 2004. ISBN 9780471724278. doi: 10.1002/0471724270. URL <http://linkinghub.elsevier.com/retrieve/pii/S016786550500019X{%}0Ahttp://doi.wiley.com/10.1002/0471724270>.

- [20] Adina Raluca Stoica. Delaunay Diagram Representations For Use in Image Near-Duplicate Detection Senior Project submitted to The Division of Science , Mathematics & Computing By. (May), 2011.
- [21] A Nemra. Robust airborne 3D visual simultaneous localisation and mapping. *PHD Thesis. Cranfield University*, 55(4-5):345–376, 2011. URL <http://dspace.lib.cranfield.ac.uk/handle/1826/6157>.
- [22] Pietro Azzari, Luigi Di Stefano, and Stefano Mattoccia. An evaluation methodology for image mosaicing algorithms. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 89–100. Springer, 2008.
- [23] Vimal Singh Bind, Priya Ranjan Muduli, and Umesh Chandra Pati. A robust technique for feature-based image mosaicing using image fusion. 2013.
- [24] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.
- [25] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision. Graz, Austria.*, pages 430–443, 2006.
- [26] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2010.
- [27] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515. IEEE, 2005.
- [28] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proceedings of the European conference on computer vision*, pages 430–443. Springer, 2006.
- [29] Tinne Tuytelaars, Krystian Mikolajczyk, et al. Local invariant feature detectors: a survey. *Foundations and trends® in computer graphics and vision*, 3(3):177–280, 2008.
- [30] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.
- [31] Takeo Kanade, Atsushi Yoshida, Kazuo Oda, Hiroshi Kano, and Masaya Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*, pages 196–202. IEEE, 1996.

- [32] Sylvie Chambon and Alain Crouzil. Évaluation et comparaison de mesures de corrélation robustes aux occultations. *Rapport de recherche*, 2002.
- [33] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [34] Dilipsinh Bheda, Mahasweta Joshi, and Vikram Agrawal. A study on features extraction techniques for image mosaicing. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(3):3432–3437, 2014.
- [35] Hemlata Joshi and Mr Khomlal Sinha. A survey on image mosaicing techniques. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 2(2):pp–365, 2013.
- [36] David Capel and Andrew Zisserman. Automated mosaicing with super-resolution zoom. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 885–891. IEEE, 1998.
- [37] Jin Zhao, Sichao Zhu, and Xinming Huang. Real-time traffic sign detection using surf features on fpga. In *High Performance Extreme Computing Conference (HPEC), 2013 IEEE*, pages 1–6. IEEE, 2013.
- [38] Bo Yu, Li Wang, and Zheng Niu. A novel algorithm in buildings/shadow detection based on harris detector. *Optik-International Journal for Light and Electron Optics*, 125(2):741–744, 2014.
- [39] Adam Schmidt, Marek Kraft, Micheal Fularz, and Zuzanna Domagała. Comparative assessment of point feature detectors in the context of robot navigation. *Journal of Automation Mobile Robotics and Intelligent Systems*, 7, 2013.
- [40] Antonio Ezio Frascarelli. Object detection. master degree thesis . malardalen university sweden.
- [41] J Gleason. Brisk (presented by josh gleason). In *International Conference on Computer Vision*, 2011.
- [42] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pages 510–517. Ieee, 2012.
- [43] Yoichiro Tokutake and Michael A Freed. Retinal ganglion cells–spatial organization of the receptive field reduces temporal redundancy. *European Journal of Neuroscience*, 28(5):914–923, 2008.



- [44] Andrew Witkin. Scale-space filtering: A new approach to multi-scale description. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.*, volume 9, pages 150–153. IEEE, 1984.
- [45] Stephen Gould, Joakim Arfvidsson, Adrian Kaehler, Benjamin Sapp, Marius Messner, Gary R Bradski, Paul Baumstarck, Sukwon Chung, Andrew Y Ng, et al. Peripheral-foveal vision for real-time object recognition and tracking in video. In *IJCAI*, volume 7, pages 2115–2121, 2007.
- [46] Matthew Brown, Richard I Hartley, and David Nistér. Minimal solutions for panoramic stitching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.
- [47] Zhengyou Zhang, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial intelligence*, 78(1-2):87–119, 1995.
- [48] Heung-Yeung Shum and Richard Szeliski. Construction and refinement of panoramic mosaics with global and local alignment. In *Computer Vision, 1998. Sixth International Conference on*, pages 953–956. IEEE, 1998.
- [49] Yuri Rzhanov, Lloyd Huff, and George Randy Cutter. Seafloor video mapping: modeling, algorithms, apparatus. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–I. IEEE, 2002.
- [50] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003.
- [51] Jason Schlessman, Mark Lodato, Burak Ozer, and Wayne Wolf. Heterogeneous mp soc architectures for embedded computer vision. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1870–1873. IEEE, 2007.
- [52] Brandyn A White. Using fpgas to perform embedded image registration. *BSc. Major Thesis, Computer Engineering, University of Central Florida*, 2009.
- [53] Richard Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2006.
- [54] Vladan Rankov, Rosalind J Locke, Richard J Edens, Paul R Barber, and Borivoj Vojnovic. An algorithm for image stitching and blending. In *Three-Dimensional and Multidimensional Microscopy: Image Acquisition and Processing XII*, volume 5701, pages 190–200. International Society for Optics and Photonics, 2005.
- [55] A.Allouache. Construction 3d à partir de vues multiples.magister, laboratoire robotique et productique, ecole militaire polytechnique (emp). algérie. 2014.

- [56] Etienne Vincent and Robert Laganiere. Matching feature points in stereo pairs: A comparative study of some matching strategies. *Machine Graphics and Vision*, 10(3):237–260, 2001.
- [57] Pascal Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *International joint conference on artificial intelligence (IJCAI)*, number CVLAB-CONF-1991-001, 1991.
- [58] Ning Sun, Zhenhai Ji, Cairong Zou, et al. Gender classification based on local binary pattern. *Journal of Huazhong University of Science and Technology: Nature Science Edition*, 35:177–181, 2007.
- [59] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.
- [60] Hongliang Jin, Qingshan Liu, Hanqing Lu, and Xiaofeng Tong. Face detection using improved lbp under bayesian framework. In *Image and Graphics (ICIG'04), Third International Conference on*, pages 306–309. IEEE, 2004.
- [61] Adel Hafiane, Guna Seetharaman, and Bertrand Zavidovique. Median binary pattern for textures classification. In *International Conference Image Analysis and Recognition*, pages 387–398. Springer, 2007.
- [62] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.
- [63] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. 1991.
- [64] Jianbo Shi et al. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE, 1994.
- [65] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [66] Joni-Kristian Kämäräinen and Pekka Paalanen. Experimental study on fast 2d homography estimation from a few point correspondences. *Research report 111, Department of Information Technology, Lappeenranta University of Technology*, 2009.
- [67] Tiago Coito, JR Caldas Pinto, and José Azinheira. Building and evaluation of a mosaic of images using aerial photographs. In *International Conference Image Analysis and Recognition*, pages 798–805. Springer, 2013.

- [68] H. Sayah A. Khellal. Realisation d'un systeme de stabilisation pour une camera embarquee sur un vehicule aerien (mosaiquage et localisation).pfe. ecole militaire polytechnique (emp), algérie. 2012.
- [69] DJEBLI HAMZA. La mosaïque d'images.thème de magister, l'ecole nationale supérieure d'informatique . algerie . 2012.
- [70] Elan Dubrofsky. Homography estimation. *Diplomová práce. Vancouver: Univerzita Britské Kolumbie*, 2009.
- [71] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in computer vision*, pages 726–740. Elsevier, 1987.
- [72] Joong jae Lee and Gyeyoung Kim. Robust estimation of camera homography using fuzzy ransac. In *International Conference on Computational Science and Its Applications*, pages 992–1002. Springer, 2007.
- [73] S Khachikian and M Emadi. Applying fast & freak algorithms in selected object tracking.
- [74] Nasser Kehtarnavaz and Sidharth Mahotra. *Digital Signal Processing Laboratory: LabVIEW-Based FPGA Implementation*. Universal-Publishers, 2010.

## **Abstract –**

The work presented in this memory, consists of features based image mosaicing and object tracking algorithms from multiple overlapped views. An image mosaic is a combination of a sequence of overlapped views, and it is a powerful mean of obtaining a larger view of a scene than the available within single views. Object tracking is the process of locating a moving object (or multiple objects) over time using a camera; these techniques are generally used on several images; like remote sensed images, bio-medical images, satellite images or other digital images. Our developed algorithms are based mainly on key-points detection and matching in order to find geometric transformation to register overlapped images on each other. Correlation and descriptors based features matching techniques were implemented and compared in Matlab platform, then, we implemented our algorithms on LabVIEW platform, in which we have created an interface to facilitate the things for users. The efficiency of our implementation was verified from the quality of the obtained results using real images.

**Keywords –** *Key points detection, Key points matching, Image mosaicing, Objects tracking.*

## **Résumé –**

Le travail présenté dans cette mémoire consiste en des algorithmes de mosaïquage d'images et de suivi d'objets à partir de multiples vues chevauchées. Ces techniques peuvent être utilisées sur plusieurs types d'images comme des images de télédétection, des images de biomédicales, des images satellitaires ou d'autres images numériques. Une image mosaïque est une combinaison d'une séquence de vues chevauchées et c'est un puissant moyen d'obtenir une plus grande vue d'une scène que celle disponible dans une seule vue. Le suivi d'objet est le processus de localisation d'un objet en mouvement (ou de plusieurs objets) au fil du temps à l'aide d'une caméra. Nos algorithmes développés sont basés principalement sur la détection et l'association de points d'intérêt afin de trouver la transformation géométrique pour le recalage des images chevauchées les unes sur les autres. Des techniques d'association basées sur la corrélation et les descripteurs ont été implémentées et comparées dans la plateforme Matlab, puis nous avons implémenté nos algorithmes sur la plateforme LabVIEW, dans laquelle nous avons créé une interface pour faciliter les choses pour les utilisateurs. Le travail est validé par les résultats obtenus à partir des images réelles.

**Mots-clés –** *Détection de points d'intérêt, Association de points d'intérêt Mosaïquage d'image, Suivi d'objets.*

## **ملخص –**

يتكون العمل المقدم في هذه المذكرة من خوارزميات إنشاء فسيفساء الصورة وتتبع الأشياء عن طريق إستخدام عدة مشاهد متداخلة. فسيفساء الصورة هي واحدة من تقنيات معالجة الصور المفيدة لمزج الصور الرقمية، تتبع الأشياء هو عملية تحديد موقع شيء متحرك (أو أشياء متعددة) مع مرور الوقت باستخدام كاميرا، يمكن إستخدام هذه التقنيات على العديد من الصور، مثل صور الاستشعار عن بعد والصور الطبية الحيوية وصور الأقمار الصناعية أو غيرها من الصور الرقمية. تعتمد الخوارزميات المطورة بشكل أساسي على التقاط النقاط المفتاحية ومطابقتها من أجل إيجاد تحويل هندسي لتسجيل الصور المتداخلة على بعضها البعض. تم تنفيذ أساليب مطابقة قائمة على الارتباط والواصف ومقارنتها في برنامج **Matlab**، ثم قمنا بتنفيذ خوارزمياتنا على برنامج **LabVIEW**، حيث أنشأنا واجهة لتسهيل الأمور للمستخدمين. تم التحقق من فعالية زرع الخوارزميات عن طريق النتائج المتحصل عليها من خلال معالجة صور حقيقية.

**كلمات مفتاحية -** ، *التقاط النقاط الدالة، مطابقة النقاط الدالة، فسيفساء الصورة، تتبع الأشياء.*