UNIVERSITY KASDI MERBAH OUARGLA
Faculty of New technologies of informatics and telecommunication
Department of electronics and telecommunication

Final Study Dissertation
In the aim of obtaining Master's degree - Academic
Domain : Sciences and technologies
Field : Automatics
Speciality : Automation and Systems
Presented by :

HAKMI Yacine ABASSI Maria

Theme:

Implementation of Modern Features Detection and Matching Algorithms

In front of the jury:

Mr : KAFI Redouane	President	UKM Ouargla
Mr : BENHELLAL Belkheir	Supervisor	UKM Ouargla
Mr : LATI Abdelhai	Co-supervisor	UKM Ouargla
Mr: BECHKA Larbi	Examiner	UKM Ouargla

Academic Year: 2019/2020

Dedication

This work is dedicated to all loved ones who have helped us during this incredible journey

I would like to thank my mother Djamila and my father Ahmed who I wanted him to be present this day with me to thank him for all his support .

Thanks to my friends Messaouda & Boutheyna for being alongside me.

To my dear parents, Mama & Mohamed who have always supported me and raised me to be a good person and to learn so I can have a bright future.

To my dear friends and all my colleagues at World Learning Algeria.

Maria Abbassi

Yacine Hakmi

Acknowledgement

During this journey of almost 18 years of learning, we would like to thank God for providing us the ability to think, write and for living a life where we had access to Education so we can learn and educate ourselves to secure successful futures.

We would like to sincerely thank our supervisor Mr. LATI Abdelhai for helping us to carry out this work, for his valuable guidance and for enlightening us with his knowledge. We thank the jury members as well for doing us the honor of evaluating our humble work.

We deeply thank all the teachers in the Electronic Department and all the people who contributed to the development of this work.

In closing, we thank our families, and our dearest parents for all the support we have received from them without forgetting our friends and our colleagues at the University of Ouargla – Kasdi Merbah.

Thank you all,

Table of Content

Table of Content

DEDICATION..... I

ACKNOLOWDGEMENT..... II

LIST OF ABBREVIATIONS..... III

TABLE OF CONTENT..... IIII

LIST OF FIGURES..... V

LIST OF TABLES..... IV

General Introduction..... 1

CHAPTER I: State of the Art about Image Features..... 3

I.1. Introduction..... 3

I.2. Image Processing..... 3

I.2.1. Segmentation..... 3

I.2.1.1 Segmentation overview..... 3

I.2.1.2 The Different Types of Image Segmentation..... 6

I.2.1.3 Summary of Image Segmentation Techniques..... 8

I.2.2. Binerisation..... 8

I.2.2.1 A Breakdown of Binarization..... 10

I.2.2.2 Steps involved in Image Binarization..... 11

I.2.3. Registration..... 12

I.3. Image Features..... 14

I.3.1. Types of Features..... 15

I.4. Applications of Image Features..... 15

I.4.1. Color Features (General features):..... 15

I.4.1.a. Texture Features:..... 16

I.4.1.b. Shape Features..... 17

I.4.2. Robotic Vision Domain Domain-specific features (Domain-specific features)..... 17

I.4.2.a. Regions (or Surfaces)....17

I.4.2.b. Lines (or Curves)....18

I.4.2.C. Points....19

I.5. Conclusion....4

CHAPTER II: Modern Features Detection and Matching Techniques 21

II.1. Introduction....21

II.2. Classical Features Detectors....21

II.2.1. Harris Detector....21

II.2.2. SUSAN Detector....23

II.2.3. FAST Detector....24

II.3. Modern Features Detectors....26

II.3. 1. SIFT Detector....26

II.3. 2. SURF Detector....30

II.3. 3. BRISK Detector....32

II.3.4. Local Binary Pattern (LBP)....36

II.3.5. Binary Robust Independent Elementary Features (BRIEF)....39

II.3.6. Other Feature Descriptors....40

II.4. Template Based Features Matching....42

II.4. 1. Similarities Measures....42

II.4. 2. Correlation Measures....43

II.5. Descriptors Based Features Matching....45

II.5.1. SIFT / SURF Approach....45

II.5. 2. FREAK/ BRISK Approach....46

II.6. Conclusion....46

CHAPTER III: Implementation Results 48

III.1. Introduction....48

III.2. Tools and Data Set....48

III.2.1 Development environment....48

III.2.2 Hardware environment....48

III.2.3 Software environment....48

III.3. Description of the Interface 49

III.4. Implementation of Features Detection.....50

II.4.1. Evaluation of Performance.....50

II.4. 2. Results Discussion.....54

II.4.2.1 Advantages of feature detections……..……54

II.4.2.2 Areas of improvement…….55

III.5. Implementation of Features Matching.....56

II.5.1. Evaluation of Performance.....56

II.5. 2. Results Discussion.....62

II.5.2.1 Advantages of Feature Matching…….62

II.5.2.2 Areas of improvement……..62

III.6. Conclusion 63

Conclusion Générale.....65

References.....67

Abstract.....73

List of Acronyms

List of Acronyms

Acronym	Expression
RGB	**Red, Green Bleu**
HSV	**Hue Saturation Value**
CCV	**Color Coherence Vector**
CM	**Color Moment**
SUSAN	**Smallest Univalue Segment Assimilating Nucleus**
FAST	**Features From Accelerated Segment Test**
USAN	**Univalue Segment Assimilating Nucleus**
DOG	**Difference Of Gaussians**
SIFT	**Scale Invariant Feature Transform**
SURF	**Speeded-Up Robust Features Descriptor**
BRISK	**Binary Robust Invariant Scalable Key Points**
LBP	**Local Binary Pattern**
LTP	**Local Ternary Pattern**
CS-LTP	**Center-Symmetric Local Ternary Pattern**
CS-LBP	**The Center-Symmetric Local Binary Pattern**
BRIEF	**Binary Robust Independent Elementary Features**
WLD	**Weber Local Descriptor**
LSS	**Local Self-Similarities**
FLSS	**Fast Local Self-Similarities**
HSOG	**Histograms Of The Second Order Gradients, Namely**
HOG	**Histogram Of Oriented Gradient**
MROGH	**Multisupport Region Order-Based Gradient Histogram**
MRRID	**Multisupport Region Rotation And Intensity Monotonic Invariant Descriptor**
SAD	**Sum Of Absolute Differences**
SSD	**Sum Of Squared Differences**
FREAK	**FAST RETINA KEYPOINT**

List of Figures

List of Figures

Figure	Page
I.1.Dog and Cat in same image	4
I.2.Localization of different objects	4
I.3.Object detection and instance segmentation	5
I.4.The shapes of all the cancerous cells	6
I.5.Locating different people in one image	6
I.6.Difference between semantic and instance segmentation	7
I.7.Plane image using Otsu's algorithm	9
I.8.Cup images transformed into binary images.	10
I.9.Identifying the threshold by using K-means method	11
I.10.Illustration of a histogram-based method	11
I.11.The RGB and HSV colors spaces.	15
I.12.Eye features extraction on color image	16
I.13.Typical iris recognition stages.	17
I.14.Localization using line and corner features	18
I.15.The trajectories of the two UAVs in the X and Y axes.	19
II.1.Classification of image points based on the eigenvalues of the autocorrelation matrix M	22
II.2.Feature detection in an image patch using FAST detector	24
II.3.Algorithm for the construction of difference of Gaussian	27
II.4.Technique to find extreme point from difference of Gaussian	28
II.5.Key-point descriptor for SIFT	29
II.6.Scale pyramid for computing the key point	30
II.7.Gaussian second order partial derivatives in y-direction and xy-direction	31
II.8.the SURF descriptor	32

II.9.Scale-space interest point detection in BRISK detector	**33**
II.10.The BRISK sampling patterns	**35**
II.11.Computing LBP descriptor for a pixel in a 3 × 3 neighborhood	**38**
II.12.LBP and CS-LBP features for a neighborhood of 8 pixels	**38**
II.13.Index pairs of the matched features.	**45**
III.1. Interface of our system	**49**
III.2. Uploading image 1	**50**
III.3. Display image 1	**50**
III.4. Uploading and displaying image 2	**51**
III.5. Selecting detector type	**51**
III.6. Selecting matching method	**52**
III.7. Features Detection using Harris features	**52**
III.8.Features Detection using BRISKF features	**53**
III.9. Features detection using FAST features	**53**
III.10.Features detection using SURFF features.	**54**
III.11. Detect SURF features and FREAK matching method.	**56**
III.12. Detect SURF features and SURF matching method	**56**
III.13. Detect Harris features and SSD matching method	**57**
III.14. Detect Harris features and SURF matching method	**57**
III.15. Example 2 of detect Harris features and SSD matching method	**58**
III.16. Example 2 of detect Harris features and SURF matching method	**58**
III.17. Example 3 of detect Harris features and SSD matching method	**59**
III.18. Detect SURF features and SURF matching method	**59**
III.19. Detect BRISK features and FREAK matching method.	**60**
III.20. Detect FAST features and SSD matching method	**60**
III.21. Detect SURF features and FREAK matching method	**61**
III.22. Example 2 of detect FAST features and SSD matching method	**61**

List of Tables

List of Tables

Table	Page
I.1. the different image segmentation algorithms	**8**
II.1. comparison table for different Detector/Descriptor	**36**
II.2. The most known correlation criteria	**44**

General Introduction

General Introduction

Our brain can interpret images much quicker than text, which is why images can communicate a product instantly. Additionally, images give depth and context to a description or story and provide a much more immersive experience than writing alone. The importance of digital image processing stems from two principal application areas: the first being the Improvement of pictorial information for human interpretation and the second being the processing of a scene data for an autonomous machine perception. Digital image processing has a broad range of applications such as remote sensing image, and data storage for transmission in business applications, medical imaging, acoustic imaging, Forensic sciences, and industrial automation. Images acquired by satellites are useful in tracking of earth resources, geographical mapping, and prediction of agricultural crops, urban population, weather forecasting, flood, and fire control. Space imaging applications include recognition and analyzation of objects contained in images obtained from deep space-probe missions. There are also medical applications such as processing of X-Rays, Ultrasonic scanning, Electron micrographs, Magnetic Resonance Imaging, Nuclear Magnetic Resonance. All this serves into one point which is that Image processing plays a big role in the development on many industries and different technologies that will help human conquer the world.

Feature detection is an important early vision problem. Previous work on feature detection include the use of grey level statistics and the detection of edges and corners. Methods based on detecting edges and corners are particularly useful in applications such as analysis of aerial images of urban scenes, airport facilities and image to map matching. Algorithms based on grey level statistics are applicable to a wider variety of images such as desert scenes and vegetation, which may or may not contain any man-made structures. Features, by definition, are locations in the image that are perceptually interesting. One can characterize an image feature detection algorithm by two attributes -- (a) Generality, and (b) Robustness. Given that the nature of salient features varies from application to application, it is desirable that a feature selection algorithm be as general as possible. In case of structured objects such features could be corners and locations with significant curvature changes. When analyzing human faces, features of interest could be the eyes, nose, mouth, etc. The generality criterion addresses the issue of whether a given feature detection algorithm can be used in a wide variety of applications such as image registration, Human Face Recognition and Motion Correspondence.

On the other hand, as the real time applications have to handle ever more data or to run on mobile devices with limited computational capabilities, there is a growing need for local descriptors that are fast to compute, fast to match, memory efficient, and yet exhibiting good accuracy. We also reviewed the basic concept of matching, as well as advances in template matching and applications such as invariant features or novel applications in medical image analysis. Additionally, deformable models and templates originating from classic template matching were discussed. These models have broad applications in image registration, and they are a fundamental aspect of novel machine vision or deep algorithms.

In this paper, we worked with our interface on both classical and modern features and we tested them to realize the pros and cons of each one. For features detection, we have proposed implementing some classical detectors such as Harris and FAST detectors, and other modern detectors as SIFT and SURF, the classical ones are easy and simple to program and provide good results in different situations for example Harris detector is more often used in stereo matching and image database retrieval since it provides good repeatability but it has lower accuracy that's why we can say that they are not completely efficient detectors. The modern detectors overcome some of problems that the classical ones have like the algorithm can be optimized and improved in the use of BRISK detector but the modern features are more expensive in terms of code complexity and calculation time. For features matching, correlation-based methods are better to be used for images with high overlapping ratio, and descriptors-based methods give better results for images with low overlapping ratio. and in matching SURF is much faster than SIFT.

During our experiment, we have talked about the basics of image processing which are segmentation, binarization and registration then we gave a review about the types and properties of image features in the first chapter. In the second chapter, we highlighted both classical and modern features detections and matching algorithms and talked about template-based feature matching and descriptor-based feature matching. Last and not least, we came to an experiment where we used our interface to try and test both classical and modern features and analyze each one’s advantages and disadvantages to come up with a conclusion of best practices of each one where this results can help future research to develop new techniques and give an overview on how to select the right method based on the obtained objectives.

Chapter I:

State of the Art about Image Features

I. State of the Art about Image Features

I.1. Introduction

The technology of Image processing encompasses by highly utilizing the computer proficiency to analyze the digital images i.e. the images generated using a computer. Image processing is used in numerous ways in many of the important technological-related fields like Oceanography, currency recognition, Medical imaging, remote image transmission, fake-note deduction, Satellite imaging etc. The Digitized image is analyzed and manipulated to improve the image's eminence. Separation of images at present is a most domineering phase in image processing which is popularly called as 'Image Segmentation'.

I.2. Image processing

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

I.2.1. Segmentation

I.2.1.1 Segmentation overview

Image segmentation is an important technology for image processing. There are many applications whether on synthesis of the objects or computer graphic images require precise segmentation. With the consideration of the characteristics of each, object composing images in MPEG4, object-based segmentation cannot be ignored. Nowadays, sports programs are among the most popular programs, and there is no doubt that viewers' interest is concentrated on the athletes. Therefore, demand for image segmentation of sport scenes is very high in terms of both visual compression and image handling using extracted athletes. In this project, we introduce a basic idea about color information and edge extraction to achieve the image segmentation. The, color information helps obtain the texture information of the target image while the edge extraction detects the boundary of the target image. By combining these, the

target image can be correctly segmented and represent. Besides, because color information and edge extraction can use basic image processing methods, they can not only demonstrate what textbook claims but also make us realize their function works. We expect that we can extract most part of the target [3].

Let us understand image segmentation using a simple example. Consider the below image:

There is only one object here – a dog. We can build a straightforward cat-dog classifier model and predict that there is a dog in the given image. But what if we have both a cat and a dog in a single image?

Figure. I. 1.Dog and Cat in same image

We can train a multi-label classifier, in that instance. Now, there is another caveat – we will not know the location of either animal/object in the image. That's where image localization comes into the picture (no pun intended!). It helps us to identify the location of a single object in the given image. In case we have multiple objects present, we then rely on the concept of object detection (OD). We can predict the location along with the class for each object using OD [2].

Figure I.2. Localization of different objects

Before detecting the objects and even before classifying the image, we need to understand what the image consists of. Enter – Image Segmentation. We can divide or partition

the image into various parts called segments. It's not a great idea to process the entire image at the same time as there will be regions in the image

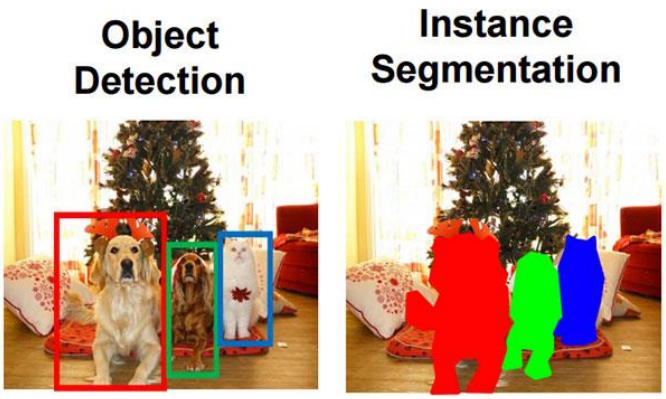


Figure I.3. Object detection and instance segmentation

which do not contain any information. By dividing the image into segments, we can make use of the important segments for processing the image. That, in a nutshell, is how image segmentation works. An image is a collection or set of different pixels. We group together the pixels that have similar attributes using image segmentation.

Object detection builds a bounding box corresponding to each class in the image. But it tells us nothing about the shape of the object. We only get the set of bounding box coordinates. We want to get more information – this is too vague for our purposes. Image segmentation creates a pixel-wise mask for each object in the image. This technique gives us a far more granular understanding of the object(s) in the image. Why do we need to go this deep? Can't all image processing tasks be solved using simple bounding box coordinates? Let us take a real-world example to answer this pertinent question [7].

Cancer has long been a deadly illness. Even in today's age of technological advancements, cancer can be fatal if we do not identify it at an early stage. Detecting cancerous cell(s) as quickly as possible can potentially save millions of lives. The shape of the cancerous cells plays a vital role in determining the severity of the cancer. You might have put the pieces

together – object detection will not be very useful here. We will only generate bounding boxes which will not help us in identifying the shape of the cells.

Image Segmentation techniques make a MASSIVE impact here. They help us approach this problem in a more granular manner and get more meaningful results. A win-win for everyone in the healthcare industry [8].

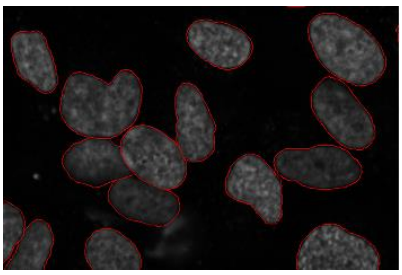


Figure I.4. The shapes of all the cancerous cells

Here, we can clearly see the shapes of all the cancerous cells. There are many other applications where Image segmentation is transforming industries:

- Traffic Control Systems
- Self-Driving Cars
- Locating objects in satellite images

There are even more applications where Image Segmentation is very useful. Feel free to share them with me in the comments section below this article – let's see if we can build something together.

I.2.1.2 The Different Types of Image Segmentation

We can broadly divide image segmentation techniques into two types. Consider the below images:

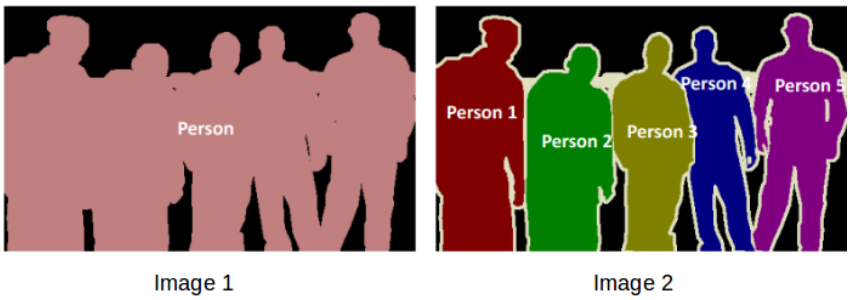


Figure I.5. Locating different people in one image

Can you identify the difference between these two? Both the images are using image segmentation to identify and locate the people present.

- In image 1, every pixel belongs to a particular class (either background or person). Also, all the pixels belonging to a particular class are represented by the same color (background as black and person as pink). This is an example of semantic segmentation.
- Image 2 has also assigned a particular class to each pixel of the image. However, different objects of the same class have different colors (Person 1 as red, Person 2 as green, background as black, etc.). This is an example of instance segmentation.

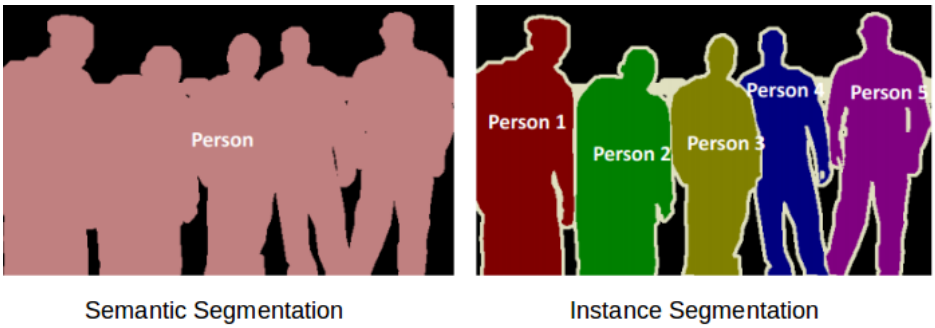


Figure I.6.Difference between semantic and instance segmentation

Let me quickly summarize what we have learned. If there are 5 people in an image, semantic segmentation will focus on classifying all the people as a single instance. Instance segmentation, on the other hand. will identify each of these people individually. So far, we have delved into the theoretical concepts of image processing and segmentation. Let's mix things up a bit – we'll combine learning concepts with implementing them in Python. I strongly believe that is the best way to learn and remember any topic [10].

I.2.1.3 Summary of Image Segmentation Techniques

I have summarized the different image segmentation algorithms in the below table. I suggest keeping this handy next time you are working on an image segmentation challenge or problem!

Algorithm	Description	Advantages	Limitations
Region-Based Segmentation	Separates the objects into different regions based on some threshold value(s).	a. Simple calculations b. Fast operation speed c. When the object and background have high contrast, this method performs well	When there is no significant grayscale difference or an overlap of the grayscale pixel values, it becomes very difficult to get accurate segments.
Edge Detection Segmentation	Makes use of discontinuous local features of an image to detect edges and hence define a boundary of the object.	It is good for images having better contrast between objects.	Not suitable when there are too many edges in the image and if there is less contrast between objects.
Segmentation based on Clustering	Divides the pixels of the image into homogeneous clusters.	Works well on small datasets and generates excellent clusters.	a. Computation time is too large and expensive. b. k-means is a distance-based algorithm. It is not suitable for clustering non-convex clusters.
Mask R-CNN	Gives three outputs for each object in the image : it is class, bounding box coordinates, and object mask	a. Simple, flexible, and general approach b. It is also the current state-of-the-art for image segmentation	High training time

Table.I.1. The different image segmentation algorithmes

I.2.2.Binerization

Auto encoders are not able to recognize the images because of the noise in the images, otherwise referred to as “image processing.” For avoiding the background noise generated in images we will use a Binarization technique commonly employed with artificial intelligence.

Image binarization is the process of taking a grayscale image and converting it to black-and-white, essentially reducing the information contained within the image from 256 shades of gray to 2: black and white, a binary image. This is sometimes known as image thresholding, although thresholding may produce images with more than 2 levels of gray. It is a form or segmentation, whereby an image is divided into constituent objects. This is a task commonly performed when trying to extract an object from an image. However, like many image processing operations, it is not trivial, and is solely dependent on the content within the image. The trick is images that may *seem* easy to convert to B&W are many times not [6].

The process of binarization works by finding a threshold value in the histogram – a value that effectively divides the histogram into two parts, each representing one of two objects (or the object and the background). In this context it is known as global thresholding (we will talk about local thresholding later). Here is an example of a plane spotters' card from WW2 (left), thresholder using Otsu's algorithm (right).

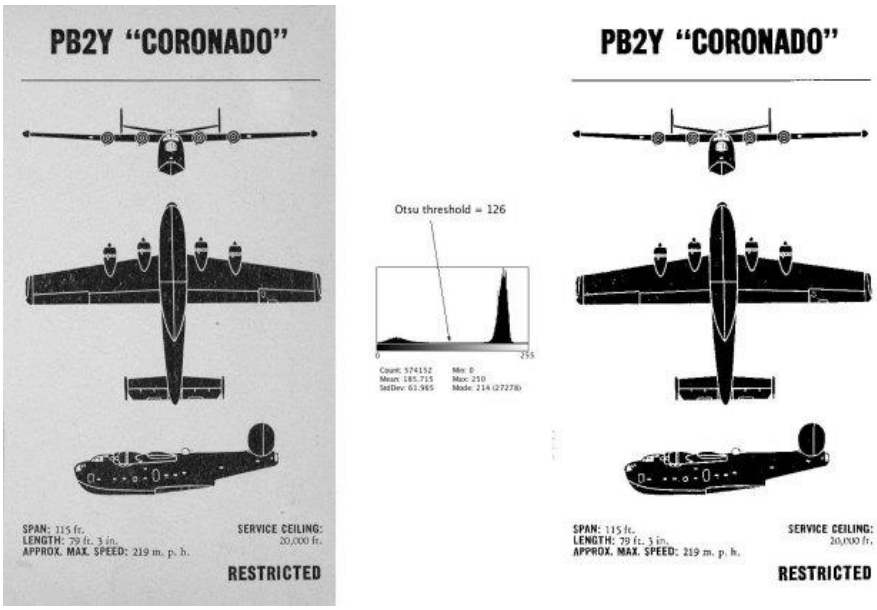


Figure I.7. Plane image using Otsu's algorithm

Most thresholding algorithms work by using some type of information to decide about where the threshold is. Sometimes the information is statistical and uses the mean, median, entropy, other times information is in the form of shape characteristics of the histogram. Otsu's algorithm is one of the classical thresholding algorithms introduced by Nobuyuki Otsu in 1979 . The algorithm works by exhaustively searching for the threshold that minimizes the weighted

within-class variance or put another way maximizes the between-class variance. (I will discuss this classic algorithm in a future post) [9].

The threshold calculated is 126, shown in combination with the histogram. To binarize the image, pixels less than 126 are set to 0, whilst pixels >= 126 are set to 1 (or 255 if you want to view it). Notice that the object is often shown as black, and the background as white. One might consider this counter-intuitive, however objects often appear as dark entities on a white background, so it is not unrealistic. Regardless of the algorithm used, the quality of the result ultimately depends on the complexity of the image. Images with simple objects are more likely to be successfully segmented than those with many varied objects[9].

There are literally hundreds of thresholding algorithms in the literature, but none to date work in a generic manner, i.e. can be applied to any image with a satisfactory result.

I.2.2.1.A Breakdown of Binarization:

A color image consists of 3 channels (Red, Green and Blue) with values ranging from 0 to 255. One of the key features of binarization is converting grey scale images into black and white (0 and 1). What is more, binarization provides sharper and clearer contours of various objects present in the image. This feature extraction improves the learning of AI models.

In the process of image binarization a threshold value is chosen, and all pixels with values above this threshold are classified as white, and all other pixels as black. The problem then is how to select the correct threshold (otherwise referred to as a thresholding method) [11].

Figure I.8. Cup images transformed into binary images

One can see that binarization takes an image with foreground/background and returns the binary image. It discards the background noise and gives the contour of the image in the foreground.

I.2.2.2.Steps involved in Image Binarization

The ‘imager ‘package uses the K-means method to automatically identify the threshold for an image and this method is equivalent to globally optimal version of popular Otsu’s method. It is very important to know that an incorrect threshold value can result in distorted binary images, where parts of the object could be missing [15].

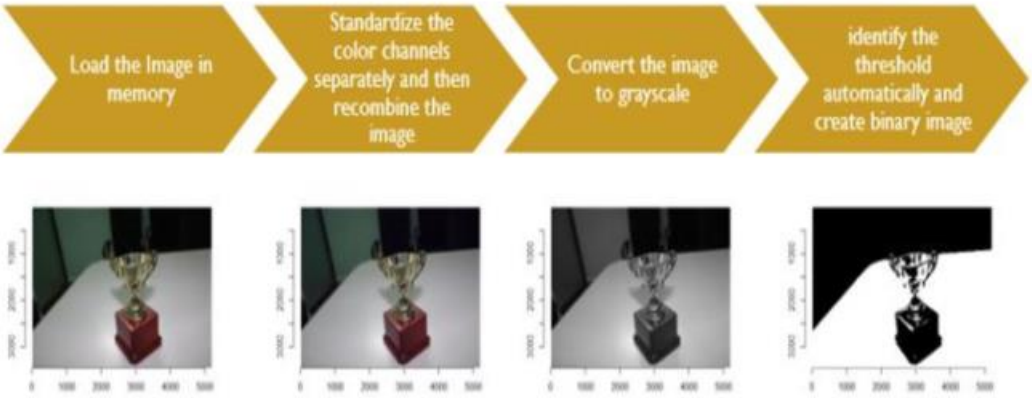


Figure I.9. Identifying the threshold by using K-means method

The image below illustrates a histogram-based method which uses pixel values.

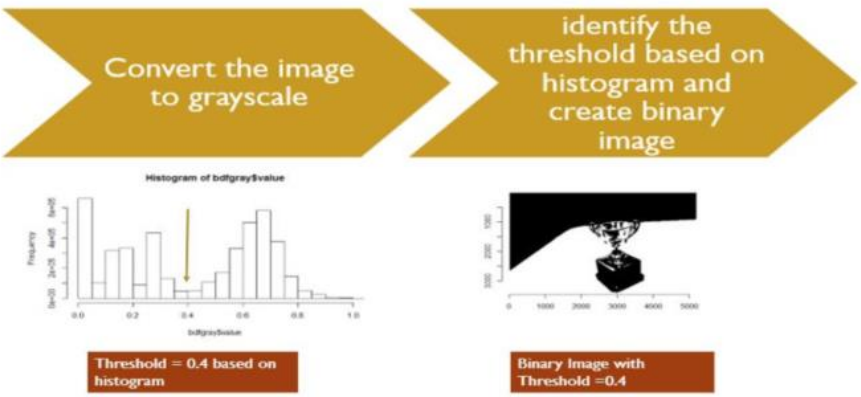


Figure I.10. Illustration of a histogram-based method

I.2.3. Registration:

Image registration is the technique which tries to find the correspondence between two or more images of the same scene capture at different times, from various view angle, and/or by various sensors and spatially align them to the coordinates of the target image.

Before exploring the various methods developed by researchers, here we are introducing some basic terms which are often used in image registration literature.

●**Reference image:** the image which fixed as the target. The expected output should be in the coordinates of this image.

●**Sensed image:** the image which is spatially mapped to be aligned with the reference image.

●**Transformation**: the mapping function used to map the sensed image to the reference image.

Because of the different types of images (taken at different time, angle, equipment) that need to be registered and the various degradation types that the image has undergone, it is not possible to define an generalized method which can be used in all registration assignment, because we have to consider not only the geometric deformation but also the presence of noise in the images which will be unique for each and every cases.

Nevertheless, we can summarize the main steps of image registration as

❶. **Feature detection** (extracting key-points and describing them using the neighborhood)

❷. **Feature matching** (finding feature correspondence between 2 images)

❸. **Transformation estimation** (find out the transformation function)

❹. **Resampling** (use the transformation function to change the coordinates of the detected image to that of the targeted image) [4].

The implementation of each step-in image registration has its basic problems, and we must pick up the right feature for our task. The features must be distinguishable objects under

all imaging conditions. The detected and the reference image must have enough common feature elements even if the sensed image undergone some geometric deformations.

The detection techniques must have good detection accuracy and less sensitive to the possible image degradation and noises. In an ideal case, the detection technique should have the ability to detect the same features every projection in the scene irrespective of the deformation that the image has undergone. Incorrect detection of features, image degradation and presence of noise, can arise as problem in the feature matching step. Sometimes features corresponds to the same key-points in 2 different images can be different due to the different imaging situations like poor lighting and/or sensor spectral sensitivity [13].

The feature descriptor should be designed by taking these considerations into account. That means the descriptor should be invariant to the assumed degradation, rotation, scaling. It must be able to different features for different key-points. The feature matching algorithm must be good enough to match the same features in both images without error even though many similar features are present in the sensed image [14].

The kind of transform function must select in accordance with the previous knowledge about the image acquisition method and the possible image degradations. If prior information is unavailable, the mapping function must be flexible enough to manage all possible deformations which we can expect. The quality of the feature detection algorithm, the robustness of feature descriptor, the reliability of the feature matching algorithm, and the tolerable approximation error need to be considered too [17].

Finally, the choice for the suitable technique for resampling depends on the compromise between the accuracy expected for the interpolation method as well as the computational easiness (time required also). The nearest-neighbor techniques or bilinear interpolation techniques are will give best results in most of the applications, but some applications require more advanced methods like spline interpolation etc [18].

I.3. Image features:

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) [54].

I.3.1. Types 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 color, shape, and texture features. Yet in robotic vision the types are divided into regions, lines, and points. In the classify the various features currently employed as follows:

❶ **General features** : Application independent features such as color, texture, and shape. According to the abstraction level, they can be further divided into:

- Pixel-level features : Features calculated at each pixel, e.g. color, 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 subarea 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. Application of image features:

I.4.1. Color Features (General features):

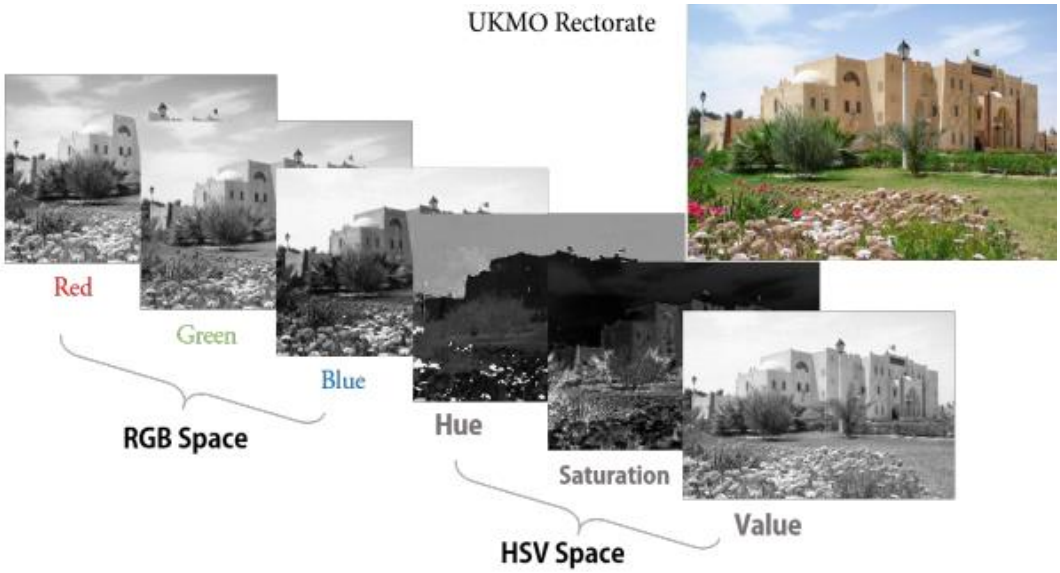


Figure I.11: The RGB and HSV colors spaces.

Color is very important feature in color images. Color features represent subject to a particular color space or model, there are many color spaces used in color imaging such as red, green, blue (RGB), hue, saturation, value (HSV) and luminance and chrominance (Y, Cb, Cr). When the color space is specified; color features can be extracted from images or regions. The extraction of color features could be done by using many techniques (color descriptors), including color histogram, color coherence vector (CCV) and color moment (CM) [4].

In Z. Zheng et al developed a robust and accurate algorithm to extract eye features from color images, he could detect the center of the pupil in H channel of HSV color space as shown in Figure I.12. 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

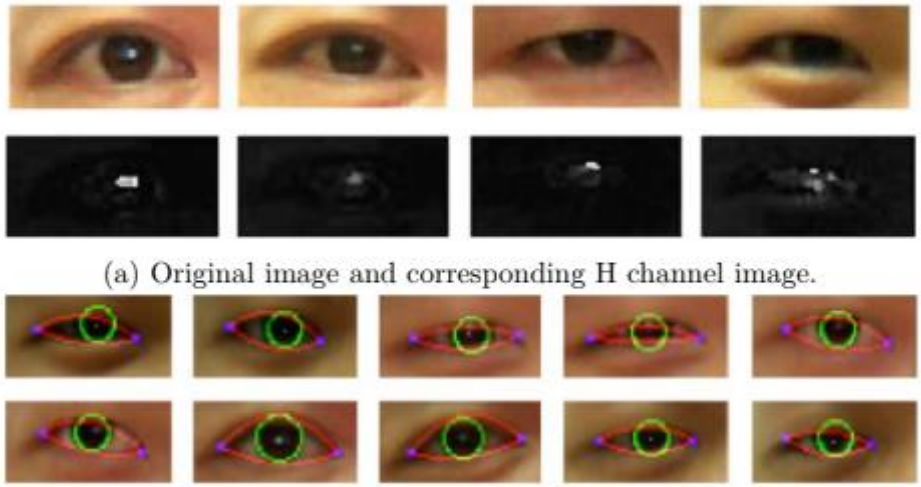


Figure I.12: Eye features extraction on color image

I.4.1.a. 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 color 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 Gabor functions analysis was used 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.13. [18].

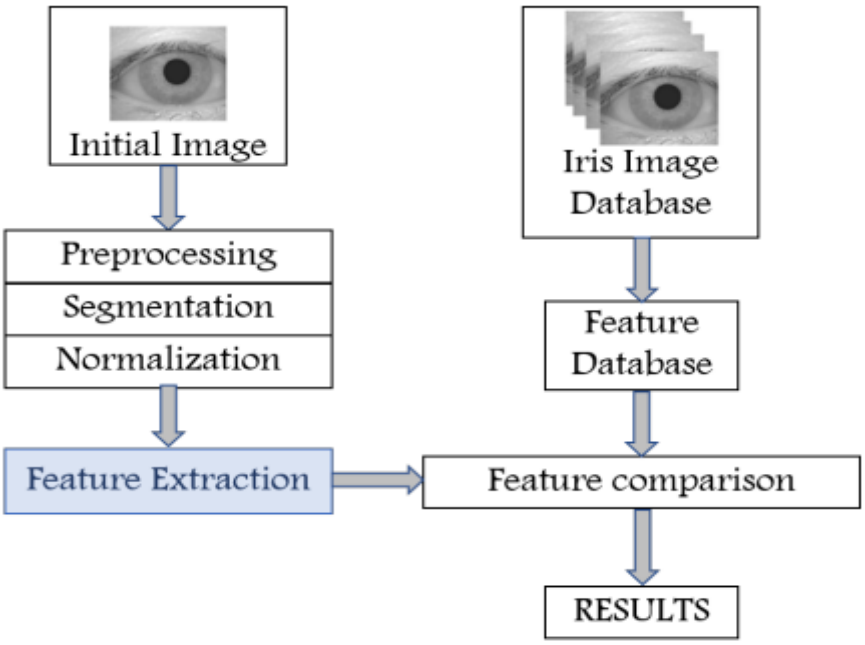


Figure I.13: Typical iris recognition stages.

I.4.1.b. 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. These types of features will be discussed in more details in the next section [19].

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

I.4.2.a. Regions (or Surfaces)

They can be projections of closed areas, water tanks, lakes, buildings, or shades. They are often represented by their gravity centers, 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; 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 .

I.4.2.b. 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 Laplacian of Gaussian, are used. The lines are often represented by pairs of points of extremities, or by their points of medium, 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 [16].

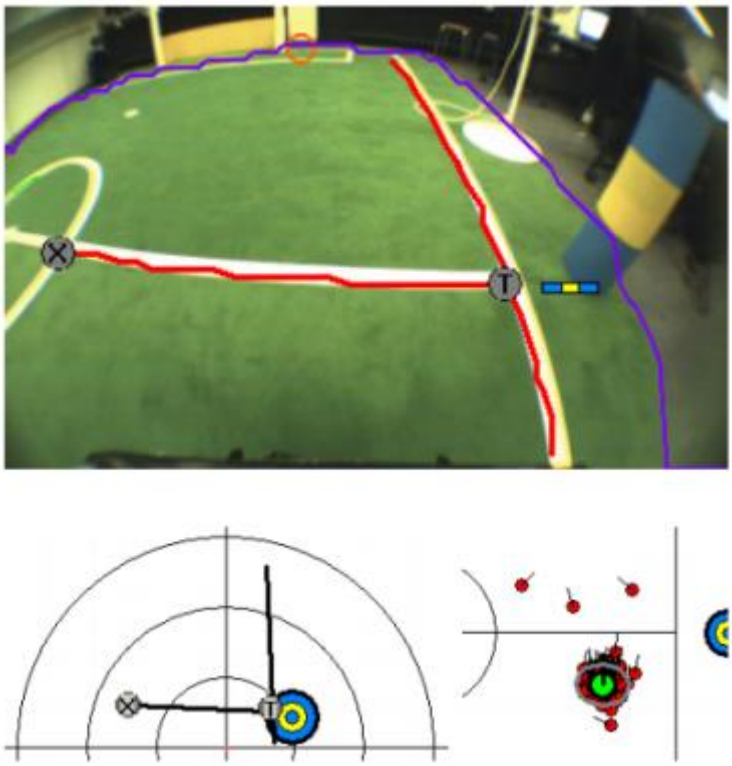


Figure I.14: Localization using line and corner features .

I.4.2.C. Points

Points are ideal for image registration because their coordinates can be used directly to determine the parameters of the transformation function, and 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 [17] [19].

The proposed technique in was for airborne enabling unmanned aerial vehicles to construct a reliable map of an unknown environment and localize 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.

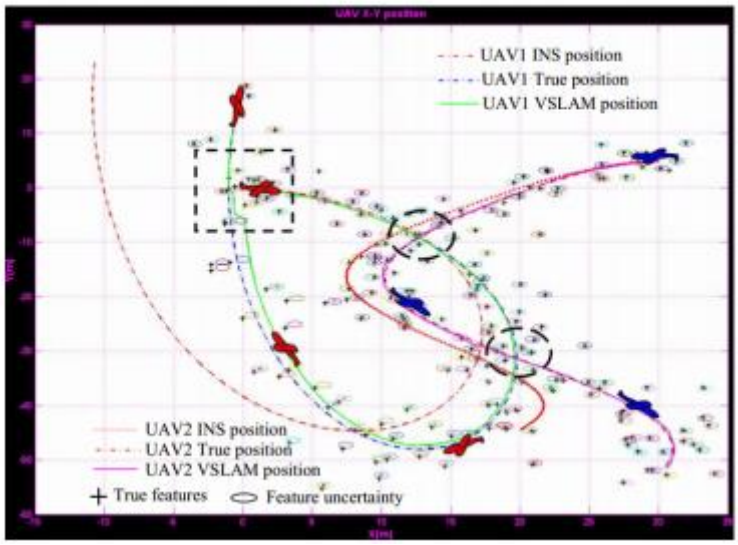


Figure I.15: The trajectories of the two UAVs in the X and Y axes.

I.5 Conclusion:

From this chapter we can conclude that digital images are employed in many domains (surveillance, traffic, military, biometry, and robotics, etc.) from multiple imaging sources, that makes image processing rich topic and reward for study. The digital images contain frequently of information, such as textures, colors, 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 following chapter, we will identify many techniques and algorithms which allowed us to extract the corners point features, and we are going to observe the propriety of those extraction techniques.

Chapter II:

Modern features detection and matching techniques

II. Modern features detection and matching techniques

II.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 analyzing.

• Pattern recognition and Biometrics (way to recognize people).

• Robotic and machine learning.

II. Classical features detectors

II.2.1. Harris Detector

Harris and Stephens have developed a combined corner and edge detector to address the limitations of Moravec's detector. By obtaining the variation of the auto- correlation (i.e., intensity variation) over all different orientations, this results in a more desirable detector in terms of detection and repeatability rate. [18]

The resulting detector based on the auto-correlation matrix is the most widely used technique. The 2 × 2 symmetric auto-correlation matrix used for detecting image features and describing their local structures can be represented as

$$\mathbf{M\ (x,\ y)} = \sum_{\mathbf{u,v}}(\mathbf{u,v}) * \begin{bmatrix} \mathbf{I_x^2(x,y)} & \mathbf{I_xI_y(x,y)} \\ \mathbf{I_xI_y(x,y)} & \mathbf{I_y^2(x,y)} \end{bmatrix} \qquad \textbf{(II-1)}$$

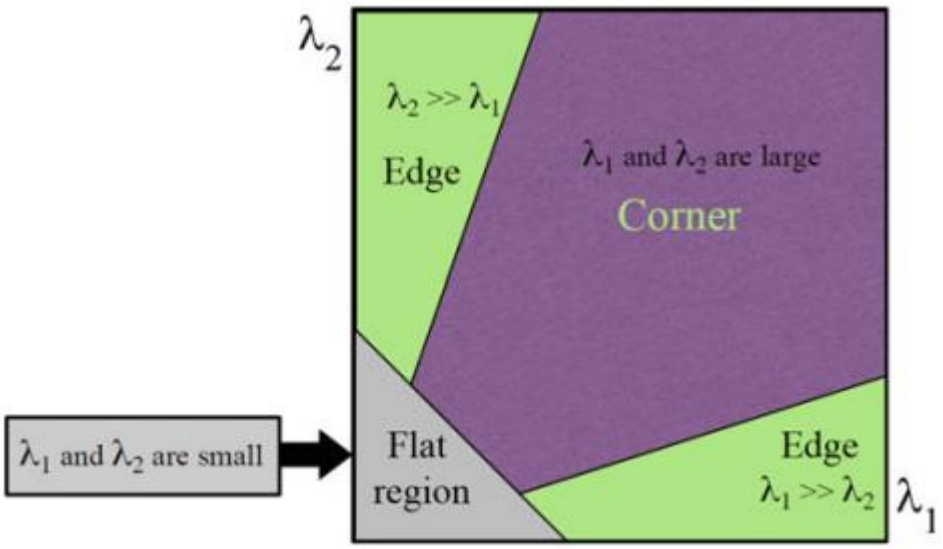


Fig.II.1.Classification of image points based on the eigenvalues of the autocorrelation matrix M

where Ix and Iy are local image derivatives in the x and y directions respectively, and w (u, v) denotes a weighting window over the area (u, v). If a circular window such as a Gaussian is used, then the response will be isotropic, and the values will be weighted more heavily near the center. For finding interest points, the eigenvalues of the matrix M are computed for each pixel. If both eigenvalues are large, this indicates existence of the corner at that location. An illustrating diagram for classification of the detected points is shown in Fig. 1. Constructing the response map can be done by calculating the cornerness measure C(x, y) for each pixel (x, y) using [18]

$$C(x, y) = \det(M) - K(\text{trace}(M))^2 \qquad \text{(II-2)}$$

Where

$$\det(M) = \lambda 1 * \lambda 2, \text{ and } \text{trace}(M) = \lambda 1 + \lambda 2 \qquad \text{(II-3)}$$

The K is an adjusting parameter and λ1, λ2 are the eigenvalues of the auto-correlation matrix. The exact computation of the eigenvalues is computationally expensive since it requires the computation of a square root. Therefore, Harris suggested using this cornerness measure that combines the two eigenvalues in a single measure. The non- maximum suppression should be done to find local maxima and all non-zero points remaining in the cornerness map are the searched corners [25].

II.2.2. Susan Detector:

Instead of using image derivatives to compute corners, Smith and Brady introduced a generic low-level image processing technique called SUSAN (Smallest Univalue Segment Assimilating Nucleus). In addition to being a corner detector, it has been used for edge detection and image noise reduction. A corner is detected by placing a circular mask of fixed radius to every pixel in the image. The center pixel is referred to as the nucleus, where pixels in the area under the mask are compared with the nucleus to check if they have similar or different intensity values. Pixels having almost the same brightness as the nucleus are grouped together and the resulting area is termed USAN (Univalue Segment Assimilating Nucleus). [19] A corner is found at locations where the number of pixels in the SUSAN reaches a local minimum and below a specific threshold value T. For detecting corners, the similar comparison function C(r,r0) between each pixel within the mask and mask's nucleus is given by

$$C(r,r0) = \begin{cases} 1, \text{if } |I(r) - I(r0)| \leq T, \\ 0, \text{otherwise}, \end{cases} \qquad \text{(II-4)}$$

and the size of USAN region is

$$n(r0) = \sum_{r \in c(r0)} C(r, r0) \qquad \text{(II-5)}$$

where r0 and r are nucleus's coordinates and the coordinates of other points within the mask, respectively. The performance of SUSAN corner detector mainly depends on the similar comparison function C(r,r0) , which is not immune to certain factors impacting imaging (e.g., strong luminance fluctuation and noises) [25].

SUSAN detector has some advantages such as: (i) no derivatives are used, thus, no noise reductions or any expensive computations are needed; (ii) High repeatability for detecting features; and (iii) invariant to translation and rotation changes. Unfortunately, it is not invariant to scaling and other transformations, and a fixed global threshold is not suitable for general situation. The corner detector needs an adaptive threshold, and the shape of mask should be modified.

II.2.3. Fast Detector:

FAST (Features from Accelerated Segment Test) is a corner detector originally developed by Rosten and Drummondn [20][21] . In this detection scheme, candidate points are detected by applying a segment test to every image pixel by considering a circle of 16 pixels around the corner candidate pixel as a base of computation. If a set of contiguous pixels in the Bresenham circle with a radius r are all brighter than the intensity of candidate pixel (denoted by Ip) plus a threshold value t , $I_P + t$, or all darker than the intensity of candidate pixel minus the threshold value $I_P - t$, then p is classified as a corner. A high-speed test can be used to exclude a very large number of non-corner points; the test examines only the four pixels 1, 5, 9 and 13. A corner

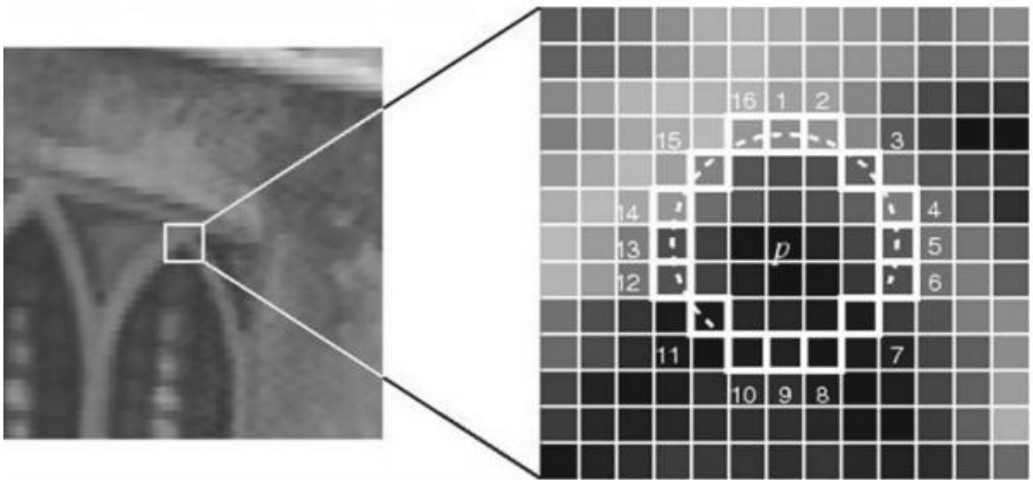


Fig.II.2. Feature detection in an image patch using FAST detector

can only exist if three of these test pixels are brighter than $I_P + t$ or darker than $I_P - t$ and the rest of pixels are then examined for conclusion. Figure 2 illustrates the process, where the highlighted squares are the pixels used in the corner detection [28].

The pixel at p is the center of a candidate corner. The arc is indicated by the dashed line passes through 12 contiguous pixels which are brighter than p by a threshold. The best results are achieved using a circle with r = 3 and n = 9.

Although the high-speed test yields high performance, it suffers from several µ limitations and weakness [21].

An improvement for addressing these limitations and weakness points is achieved using a machine learning approach. The ordering of questions used to classify a pixel is learned by

using the well-known decision tree algorithm (ID3), which speeds this step up significantly. As the first test produces many adjacent responses around the interest point, an additional criterion is applied to perform a non-maximum suppression. This allows for precise feature localization. The used cornerness measure at this step is

$$C(x, y) = \max\left(\sum_{j \in Sbright} \left|I_{p \to j} - I_p\right| - t, \sum_{j \in Sbright} \left|I_p - I_{p \to j}\right| - t\right) \quad \text{(II-6)}$$

where $I_{P \to j}$ denotes the pixels laying on the Bresenham circle. In this way, the processing time remains short because the second test is performed only on a fraction of image points that passed the first test.

In other words, the process operates in two stages. First, corner detection with a segment test of a given n and a convenient threshold is performed on a set of images? (preferably from the target application domain).

Each pixel of the 16 locations on the circle is classified as darker, similar, or brighter.

Second, employing the ID3 algorithm on the 16 locations to select the one that yields the maximum information gain.

The non-maximum suppression is applied on the sum of the absolute difference between the pixels in the contiguous arc and the center pixel. Notice that the corners detected using the ID3 algorithm may be slightly different from the results obtained with segment test detector due to the fact that decision tree model depends on the training data, which could not cover all possible corners [26].

Compared to many existing detectors, the FAST corner detector is very suitable for real-time video processing applications because of its high-speed performance. However, it is not invariant to scale changes and not robust to noise, as well as it depends on a threshold, where selecting an adequate threshold is not a trivial task.

II.3. Modern features detectors

Once a set of interest points has been detected from an image at a location p(x, y), scale s, and orientation θ, their content or image structure in a neighborhood of p needs to be encoded in a suitable descriptor for discriminative matching and insensitive to local image deformations. The descriptor should be aligned with θ and proportional to the scale s. There are many image feature descriptors in the literature; the most frequently used descriptors are discussed in the following sections.

II.3.1 Scale Invariant Feature Transform (SIFT)

SIFT is an efficient scale and rotation invariant detector cum descriptor which is commonly used for feature matching and registration. This descriptor is comparatively less sensitive about scaling transformations in the image domain and robust to locally occurring deformation and light intensity variation. SIFT consists of four different steps: scale-space key point detection, key point localization, orientation assignment and feature description [22].

The initial stage of key point detection is to find out the position and scales of the key-points that can be repeatably assigned under various viewing conditions of the same target. Detection of scale invariant points of the image can be achieved by checking for stable key-features throughout all possible each scale, using a special function of scale called as scale space. It has been experimentally proved under some reasonable assumptions that Gaussian function is the only possibility for a scale space kernal. Therefore, the scale space function of an image L(x, y, σ), can be defined as the convolution between the variable-scale Gaussian kernal, G(x, y, σ), and the input image, I(x, y)

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \qquad \text{(II-7)}$$

where '$*$' is a mathematical operator which denotes convolution.

$$G(x, y, \sigma)=\frac{1}{2\sigma\pi^2} e^{-(x^2+y^2)2\sigma^2} \qquad \text{(II-8)}$$

Now the next step is to find out the scale space extrema point. To achieve this goal, we used the result of convolution between DoG function and input image. D(x, y, σ) can be obtained from the difference between the two adjacent scales differed by a constant scaling factor k. By this method we can accurately detect the stable candidate point locations and their scale in the scale space [22].

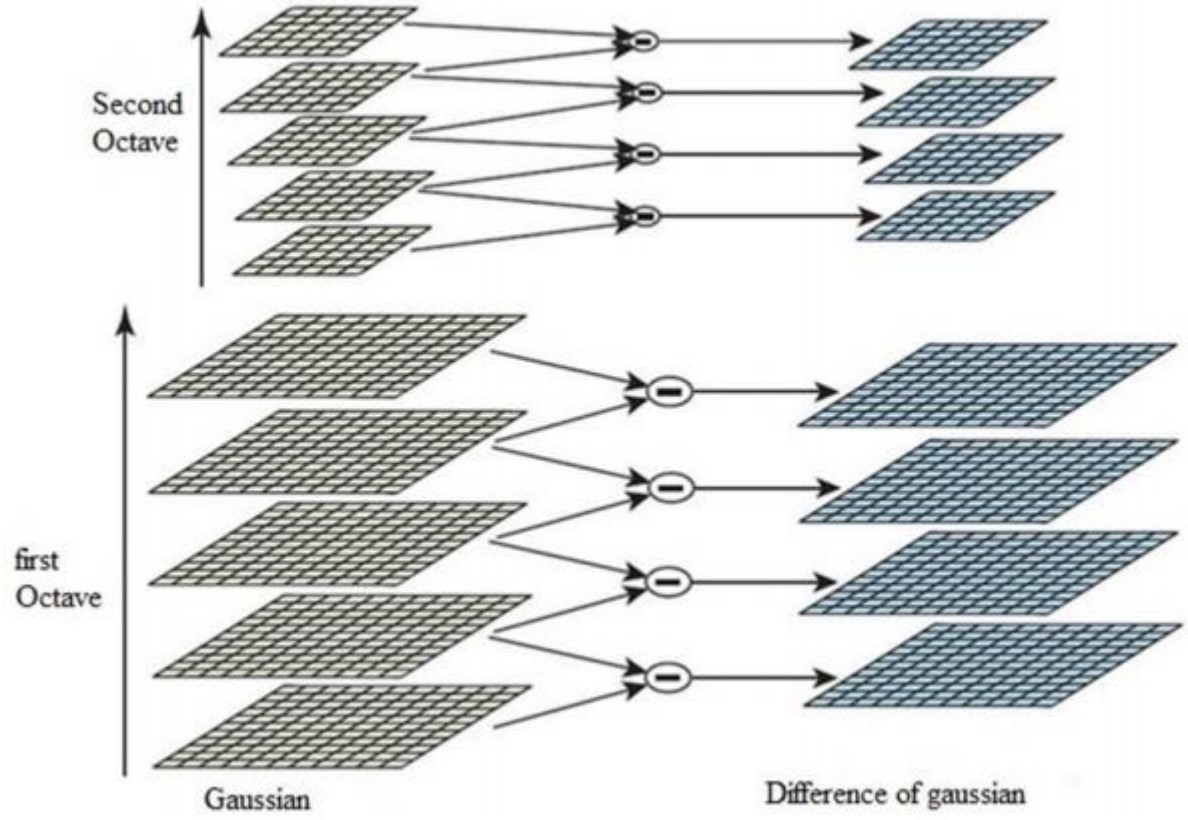


Figure.II.3. Algorithm for the construction of difference of Gaussian

$$D(x, y, \sigma) = (G(x, y, k_\sigma) - G(x, y, \sigma)) * I(x, y) \quad = L(x, y, k_\sigma) - L(x, y, \sigma) \qquad \text{(ll-9)}$$

A robust algorithm for the implementation of D(x, y, σ) is shown in Figure 4. The original image is convolved with Gaussian kernel with increasing scale to produce images differed by a constant scaling element k in scale space, illustrated in the figure above. Each octave in the scale space is obtained by scaling the previous octave with a factor 2. Typical value of the number of images required in each octave are fixed as s + 3. The difference between neighboring image scales in the same octave are calculated to find out the DoG. After a total octave has been formed, we up sample the Gaussian image with a scaling factor 2_σ by taking every 2^{nd} point in every row and column. The precision of sampling operation depend to σ is unchanged compared with the initial stage of the previous octave, while computational easiness is comparably increasing [23].

❶ Local extrema detection

For detecting the maximum or minimum point of D(x, y, σ), we have to compare each key-point to its all the neighbors in the present frame (eight) and all the neighbors in the scale top and bottom (nine) (see Figure 5). The point will consider it as the interest point if and only if it is larger or lesser than all of these neighbors. The expense for this search (computational time) is considerably less because many of the candidate points will be expelled in the first few search [23].

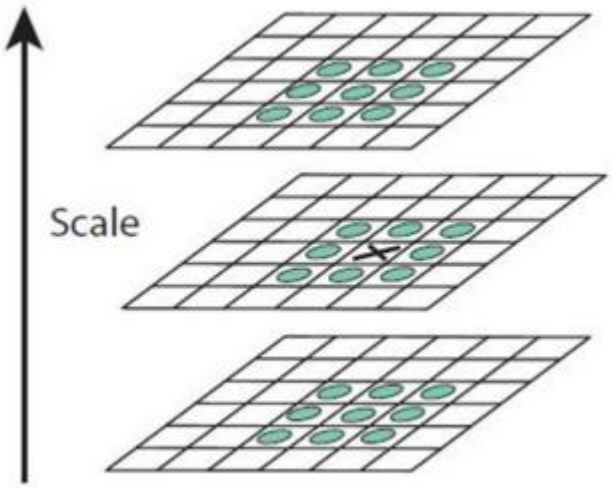


Figure.II.4. Technique to find extreme point from difference of Gaussian

❷ Orientation assignment

By including the exact orientation and gradient magnitude to each interest point based on the neighborhood of the interest point, the interest point descriptor become rotation invariant. The scale of the detected candidate point is utilized in the selection of the Gaussian kernel to smooth the image, L, with the nearest scale, so that all operations can be done in a scale independent manner. For a particular image sample, L(x, y), at its scale, the gradient magnitude, is denoted as m(x, y), and orientation is denoted as θ(x, y), can be computed using the difference between adjacent pixels.

$$m(x, y) = \sqrt{(L(x\ 1, y)\ L(x\ 1, y))^2 + (L(x, y\ 1)\ L(x, y\ 1))^2}$$

$$(x, y) = \tan^{-1}\ ((L(x, y\ 1)\ L(x, y\ 1)\ /\ (L(x\ 1, y)\ L(x\ 1, y))) \qquad \text{(II-10)}$$

The orientation histogram of this descriptor is made from the orientations of the key-point inside a local area around the interest point. The orientation histogram consists of total 36 bins around the key-point.

❸ The local image descriptor

The next step is to calculate a key-point descriptor by considering the local neighborhood of the interest point that is highly distinguishable and unique yet is as invariant to scale changes and rotation. This descriptor is invariant to variations, such as change in lighting or other degradation.

●Descriptor representation

The candidate point descriptor is formed by first calculating both the gradient magnitude and orientation of the neighborhood area around the key point location, as illustrated on the figure. All these vectors are then concatenated into orientation histograms by including the contents of all 4x4 sub regions. Here the length of each arrow represents the total sum of the gradient magnitudes closer to that direction in the vicinity of the region. Here we use an orientation histogram having 8 bins around the interest point. Therefore, we will get a feature descriptor vector of dimension 4X4X8=128 [29].

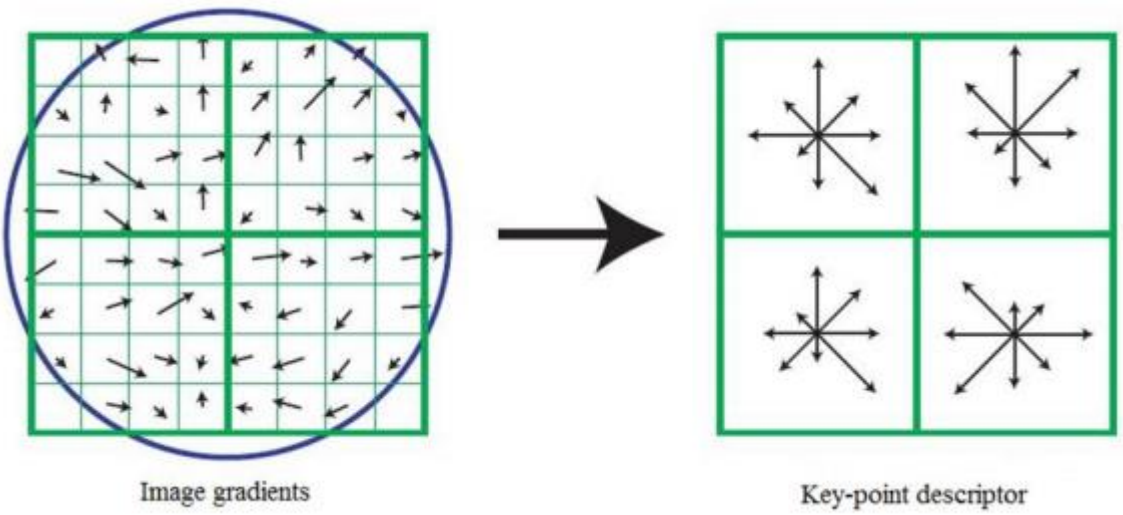


Figure.II.5. Key-point descriptor for SIFT

II.3.2. Speeded-Up Robust Features Descriptor (SURF):

SURF is basically derived from SIFT. Like the name indicates for feature extraction, SURF is faster than SIFT which is the main requirement of the today's real time applications. SURF detector used the help of approximated Hessian Matrix to modify the SIFT detector .

SURF descriptor uses response of the haar-wavelets within the neighborhood of the interest point for feature description. . Both the detector and descriptor consume less time because the detector is an approximated version of SIFT and the descriptor is of lower dimension compared with SIFT. So that SURF is better than previously used schemes (SIFT) with respect computational time. Here the increase in speed is achieved at the expense of losing accuracy[22].

SURF forms an image pyramid without performing 2:1 down sampling for upper levels in the pyramid, so that the resolution of all images will be the same. With the help of box filter approximation of second-order partial derivatives of Gaussian function SURF descriptor filters the stack. This is an advantage with the integral images, and it allows to calculate the rectangular box filters in considerably very less time. For interest point matching operation, the nearest neighbor can be defined as the key points whose descriptors are at minimum distance.

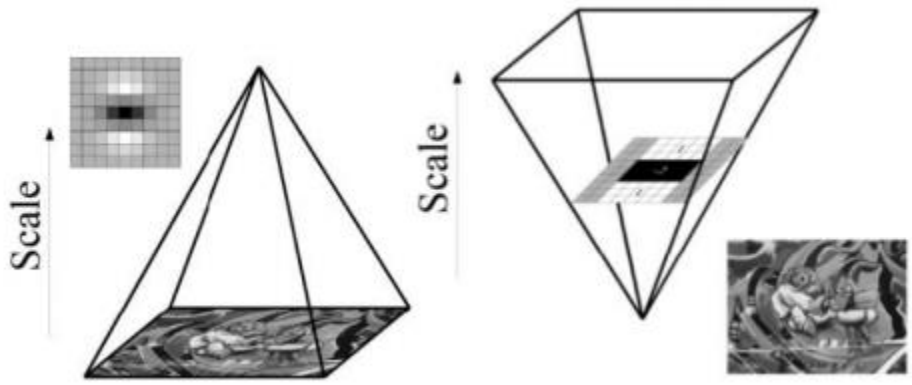


Figure.II.6.Scale pyramid for computing the key point

3.2.1 Fast Hessian Detector

Surf detector uses the help of Hessian metrics to approximate the 2nd order approximation of the Gaussian function. This gives good robustness and ability to find good matches with pair image. Consider the point X = (x, y) is the given image I, then the Hessian metrics H(x, σ) for the point X with the Scale σ, is described as the equation below

$$\mathbf{H(x,\sigma)} = \begin{bmatrix} \mathbf{L_{xx}(x,\sigma)} & \mathbf{L_{xx}(x,\sigma)} \\ \mathbf{L_{xx}(x,\sigma)} & \mathbf{L_{xx}(x,\sigma)} \end{bmatrix} \qquad \textbf{(II-11)}$$

Scale spaces are generally constructed as image pyramids. Here the images are convolved many times with the Gaussian kernal and then sub-sampled in order to make a top level of the pyramid. In SIFT algorithm we used second order derivative of Gaussian function, but we use the 9 × 9 box filters which are the approximations for second order derivatives of

Gaussian function. D_{xx} , D_{xy}, D_{yy} denotes the approximations of 2nd order derivative of Gaussian function [23].

The weights put into the rectangular areas are kept simple to improve computational easiness and simplicity. Here in the next step we need to normalize the relative weight with the Hessian's determinant.

$\frac{|L_{xy}(1.2)|_F \ |D_{xx}(9)|_F}{|L_{xx}(1.2)|_F \ |D_{xy}(9)|_F} = 0.912\ldots \approx 0.9$ where $|x|_F$ is the Frobenius norm. This gives

$\det(H_{approx}) = D_{xx}\,D_{yy} - (0.9D_{xy})^2$ approx. Because of the use of box filters and integral images, it is not

required to use the same filter to the output of a previously filtered layer, but we can use filters of different sizes with good speed directly on the base image.

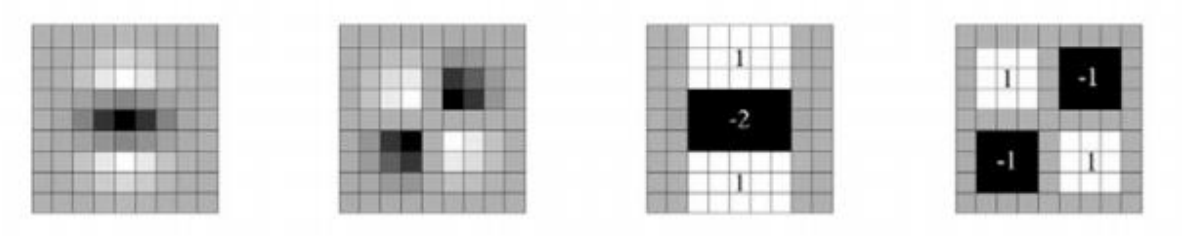


Figure.II.7. Gaussian second order partial derivatives in y-direction and xy-direction

3.2.2 Orientation Assignment

In order to make the descriptor rotation invariant it is required to compute the orientation for the key points. To achieve this goal, hear wavelet response in both x and y direction must calculate for all neighboring pixels inside a circle of radius 6s around the key point. Once the wavelet response is calculated then the next step is to find out the dominant orientation of the candidate point. The dominant orientation is computed by calculating the sum of all responses (haar wavelet response in both x and y direction) inside a rotating window covering an angle of $\pi/3$. The responses from both horizontal and vertical direction inside the window should be added. The sum of these responses then gives a new vector. The lengthiest such vector gives its orientation to the key point [30].

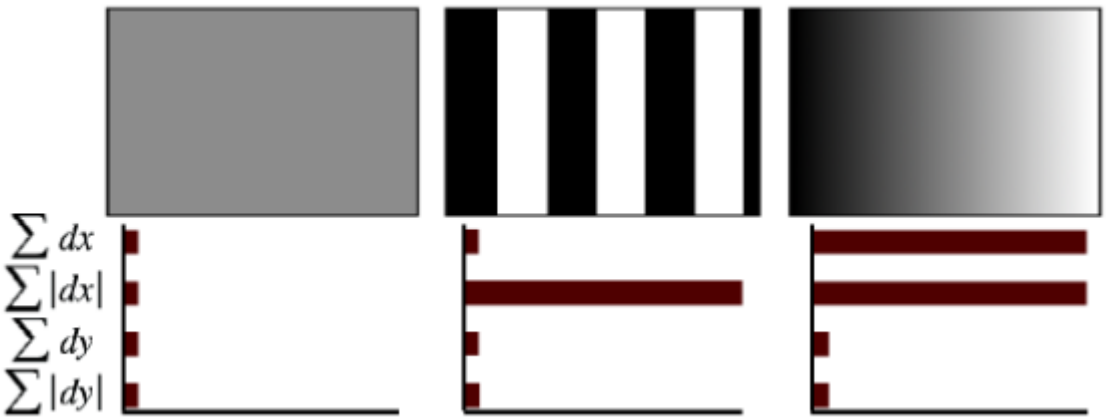


Figure.II.8. the SURF descriptor

Descriptor components

Constructing a square region of size 20s is the next step in extracting the feature descriptor vector. Next, we will split this region to 4x4 square sub region. For each sub region we need to calculate the response of haar wavelets along both horizontal dx and vertical d_x direction. Then, the wavelet responses dx and d_y should be added up over each sub area and form a first set of elements of the feature descriptor vector. In order to incorporate the information about the gradient of the intensity variations, it is required to calculate the sum of the modules of the responses, $|d_x|$ and $|d_y|$. Like this, each sub-area has a descriptor vector v with dimension 4, $\mathbf{v = (\sum dx, \sum d_x, \sum |dx|, \sum |d_y|)}$. Similarly, all the 4X4 areas also have four-dimensional vector finally results in SURF descriptor with dimension 64D [23].

II.3.3. Binary Robust Invariant Scalable Key points (BRISK)

❶Scale Space Key Point Detection

Achieving scale invariance is a most important thing for a high-quality key point. For this purpose, BRISK Descriptor go a step forward by checking for the maxima or minima point in both image plane and the scale-space plane using the FAST score ‘s’ as a measure for robustness. To find the true scale point most of the detectors discretize the scale axis at coarser intervals, But the BRISK detector estimates the actual scale of all key point in the continuous scale-space plane [23].

In this detector algorithm, the scale-space stack layers comprise of n octaves ci and n intra-octaves d_i , for i = {0, 1 . . . n − 1} and generally n = 4. The octaves are created from the

repeated half- sampling the base image (represented as c_0). Each intra-octave di is placed in-between 2 adjacent octave layers (as shown in Figure 9). The first intra-octave d_0 is formed by down sampling the base image c_0 by a scaling factor of 1.5, and the remaining intra-octave layers are formed by repeated down sampling by a factor 2. Therefore, if 't' represents scale then t (ci) = 2^i and t (d_i) =2^i· 1.5.

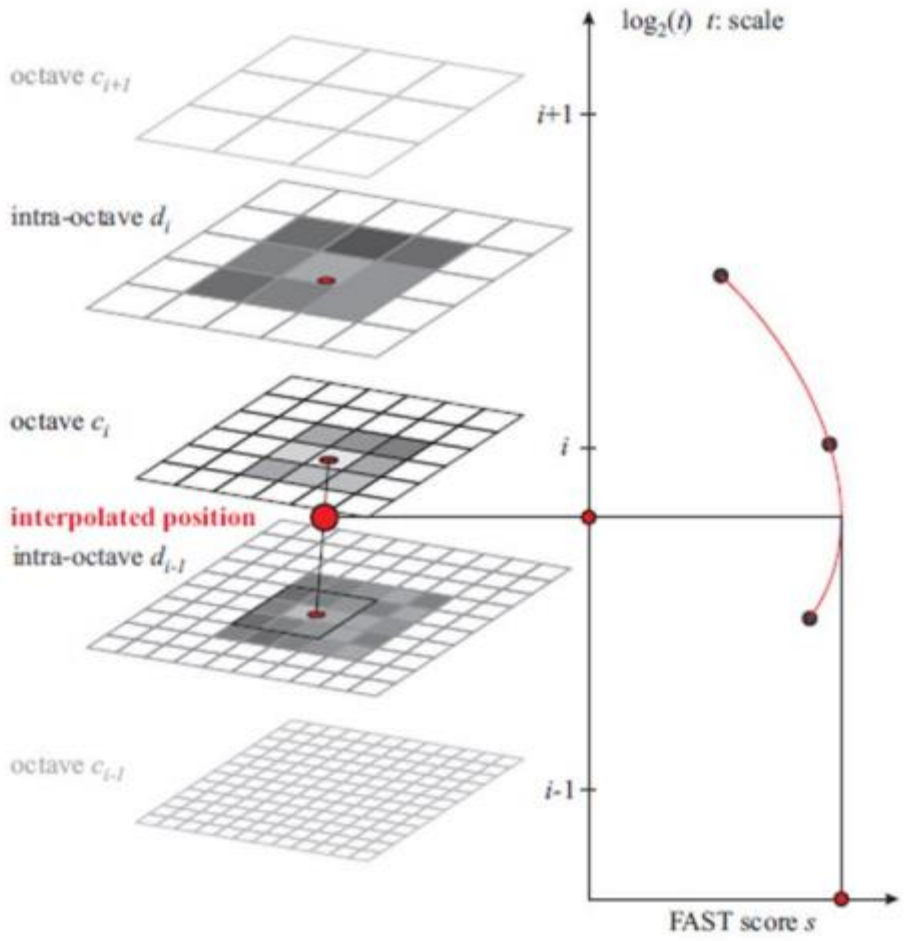


Figure.II.9. Scale-space interest point detection in BRISK detector

In BRISK detector, 9-16 masks are mostly used, which demands at least minimum 9 continuous points in the 16-point circle centered at the candidate point which are of intensity values more than the intensity value of the candidate point plus threshold t or less than the intensity value of the candidate point minus threshold t for the FAST condition to be satisfied [22][23].

In the initial stage, the FAST 9-16 detector is acted on each and every octave and intra-octave individually with the same threshold 'T' to find out the key-point of interest. Then, the candidate point being referred here required to satisfy the maximum condition as for its 8

neighboring FAST scores ‘s’ in the current layer. The score‘s’ is nothing but the maximum value of threshold still accepting an image candidate point as a corner. Also, the scores in the top and beneath layer will must be lower too.

Taking image saliency as a continuous variable over the image as well as along the scale space, we perform a sub-pixel and ceaseless scale refinement for every identified maximum. To find out the actual scale of the interest point which is detected a 1D parabola is fitted at the scale space plane axis. As a last step, re-interpolation of the image coordinates between the patches in the layers next to the determined scale are performed [24].

❷ Key point Description

For a number of key points, the BRISK descriptor is made as a binary string by combining the outputs of simple intensity level comparison tests. In BRISK, we find out the orientation direction of each and every key point for design an orientation-normalized descriptor so that the descriptor becomes rotation invariant which is the most important attraction for a descriptor.

❸ Sampling Pattern and Rotation Estimation

The method used for sample the neighboring elements of the candidate point is the key concept behind BRISK descriptor. The pattern, described in Figure 3, describes N distinct places equally separated on circles centered at the key point. For avoiding the effects of aliasing when sampling the intensity levels of an image at a point pi in the image, we used Gaussian smoothing filter with standard deviation σ_i directly proportional to the distance between the 2 points on the respective circle. If we have N sample points, then we will be having N. (N-1)/2 number of sampling point pairs. Consider a sampling point pairs (p_i , p_j) and I(p_i , σ_i) and I(p_j , σ_j), be the Gaussian smoothed version of these two sampling points. Local gradient g(p_i , p_j) can be calculated using these sampling point pairs, and it is Gaussian smoothed version using this expression [24].

$$\mathbf{g}\,(\mathbf{P_i}, \mathbf{P_j}) = \mathbf{P_i} - \mathbf{P_j} \cdot \frac{\mathbf{I}(\mathbf{P_i}, \mathbf{P_j}) - (\mathbf{Pi} - \boldsymbol{\sigma}\mathbf{j})}{\|\mathbf{Pi} - \mathbf{Pj}\|^2} \qquad \textbf{(II-12)}$$

$$\mathbf{g}\begin{pmatrix} \mathbf{g_x} \\ \mathbf{g_y} \end{pmatrix} = \frac{\mathbf{1}}{\mathbf{L}} \cdot \sum_{(\mathbf{P_i}, \mathbf{P_j}) \in \mathbf{L}} \mathbf{g}(\mathbf{P_i}, \mathbf{P_j}) \qquad \textbf{(II-13)}$$

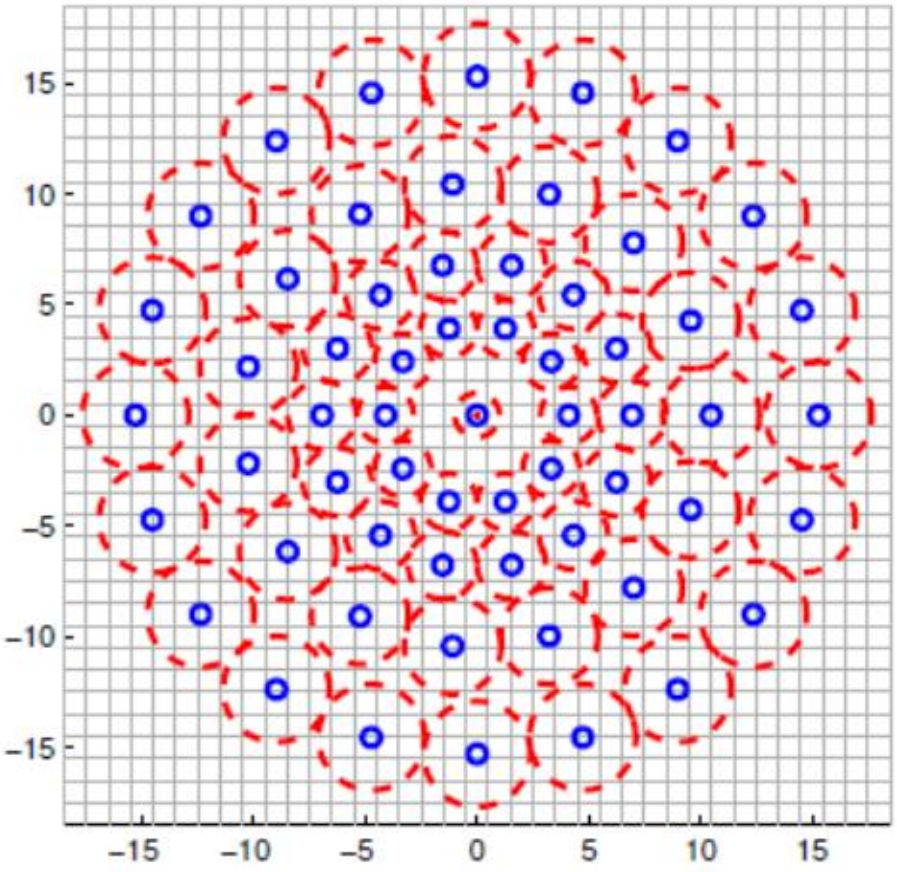


Figure.II.10. The BRISK sampling patterns

❹ Building the Descriptor

To design a rotation- and scale-invariant feature descriptor, BRISK descriptor uses a sampling pattern rotated by $\alpha = \arctan2(g_y, g_x)$ around the interest point k. The bit stream - vector descriptor d_k is created by calculating all the short distance intensity differences of sample point pairs $(P_i^a, P_j^a) \in S$ (i.e. in the rotated pattern), here the component of the binary string bit 'b' is defined as:

$$b = \begin{cases} 1, & I(P_j^a, \sigma_j) > I(P_j^a, \sigma_i) \\ 0 & \text{otherwise} \end{cases} \quad \forall \ (P_i^a, P_j^a) \in S \qquad \text{(II-14)}$$

❺ Descriptor Matching

Matching between two BRISK descriptor vectors can be established by a simple calculation of their Hamming distance. The similarity between 2 descriptors can be measured

by counting the total number of bits which are same for both of them. This operation can implement easily by doing the bitwise XOR operation followed by counting the number of zeros.

Detector/Descriptor	SIFT	SURF	FAST	DAISY	BRISK
Avg Features	487	376	516	n.a	**647**
Detector **ms**/image	1.210	0.640	0.097	n.a	**0.45**
Detector **ms**/feature	1.08	0.25	n.a	0.33	**0.061**
Detector **bytes**/image	128	64	n.a		**64**

Table.II.1. comparison table for different Detector/Descriptor

This table shows the BRISK detector-descriptor outperforms the other detector-descriptor in all respect. Here number of key-points detected, time per detector, and storage bytes are the things compared everywhere BRISK gives the best results. But lot of confusions came during matches with BRISK detector also. So, we need a better descriptor. The next chapter is discussing about proposed modified BRISK descriptor.

II.3.4. Local Binary Pattern (LBP)

Local Binary Patterns (LBP) characterizes the spatial structure of a texture and presents the characteristics of being invariant to monotonic transformations of the gray-levels. It encodes the ordering relationship by comparing neighboring pixels with the center pixel, that is, it creates an order-based feature for each pixel by com- paring each pixel's intensity value with that of its neighboring pixels. Specifically, the neighbors whose feature responses exceed the central's one is labeled as '1' while the others are labeled as '0'. The co-occurrence of the comparison results is subsequently recorded by a string of binary bits [24.25]. Afterwards, weights coming from a geometric sequence which has a common ratio of 2 are assigned to the bits according to their indices in strings. The binary string with its weighted bits is consequently transformed into a decimal valued index (i.e., the LBP feature response). [28] That is, the descriptor describes the result over the neighborhood as a binary number (binary pattern). On its standard version, a pixel c with intensity g(c) is labeled as

$$S(g_p - g_c) = \begin{cases} 1, & \text{if } g_p \geq g_c \\ 0, & \text{if } g_p < g_c \end{cases} \qquad \text{(II-15)}$$

where pixels p belong to a 3 × 3 neighborhood with gray levels g_p(p = 0, 1,..., 7). Then, the LBP pattern of the pixel neighborhood is computed by summing the corresponding thresholder values $S(g_p - g_c)$ weighted by a binomial factor of 2^k as

$$\mathbf{LBP} = \sum_{\mathbf{k=0}}^{\mathbf{7}} \mathbf{S}\left(\mathbf{g_p} - \mathbf{g_c}\right).\mathbf{2^K} \qquad \text{(II-16)}$$

After computing the labeling for each pixel of the image, a 256-bin histogram of the resulting labels is used as a feature descriptor for the texture. An illustration example for computing LBP of a pixel in a 3 × 3 neighborhood and an orientation descriptor of a basic region in an image is shown in Fig. 12. Also, the LBP descriptor is calculated in its general form as follows

$$\mathbf{LBP_{RN}\ (x,\ y)} = \sum_{\mathbf{i=0}}^{\mathbf{N-1}} \mathbf{S}(\mathbf{n_i} - \mathbf{n_c}).\mathbf{2^i}\ ,\ \mathbf{S(x)} = \begin{cases} \mathbf{1, x \geq 0} \\ \mathbf{0} \quad \mathbf{otherwise,} \end{cases} \qquad \text{(II-17)}$$

where $\mathbf{n_c}$ corresponds to the gray level of the center pixel of a local neighborhood and n_i is the gray levels of N equally spaced pixels on a circle of radius R. Since correlation between pixels decreases with the distance, a lot of the texture information can be obtained from local neighborhoods. Thus, the radius R is usually kept small [35] .

In practice, the signs of the differences in a neighborhood are interpreted as a N-bit binary number, resulting in 2^N distinct values for the binary pattern as shown in Fig. 1. The binary patterns are called uniform patterns, where they contain at most two bitwise transitions from 0 to 1. For instance, “11000011” and “00001110” are two uniform patterns, while “00100100” and “01001110” are non-uniform patterns. Several variations of LBP have been proposed, including the center-symmetric local binary patterns (CS-LBP), the local ternary pattern (LTP), the center-symmetric local ternary pattern (CS-LTP) based on the CS-LBP, and orthogonal symmetric local

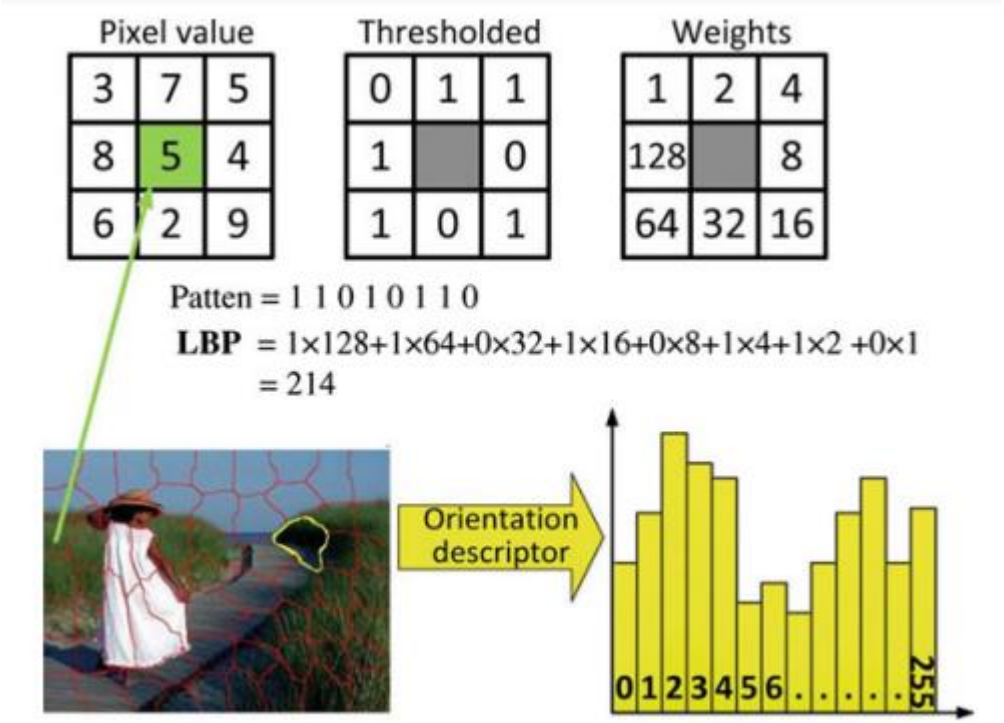


Fig.II.11. Computing LBP descriptor for a pixel in a 3 × 3 neighborhood

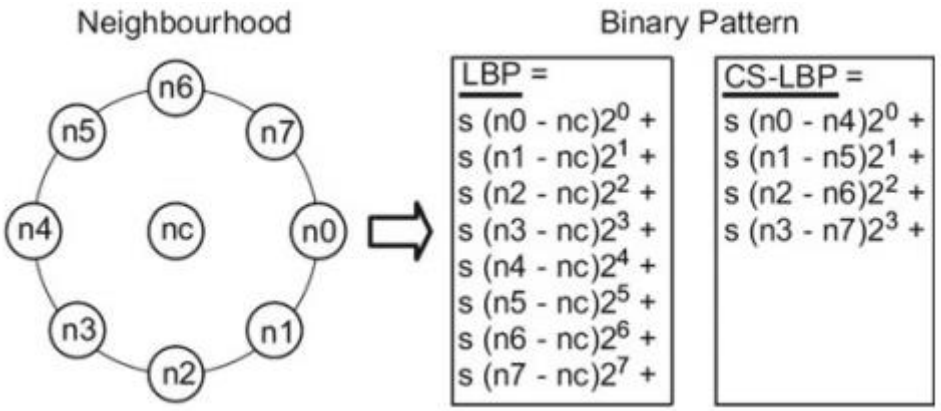


Fig.II.12. LBP and CS-LBP features for a neighborhood of 8 pixels

ternary pattern (OS-LTP) [27]. Unlike the LBP, the CS-LBP descriptor compares gray-level differences of center-symmetric pairs (see Fig. 12). In fact, the LBP has the advantage of tolerance of illumination changes and computational simplicity. Also, the LBP and its variants achieve great success in texture description. Unfortunately, the LBP feature is an index of discrete patterns rather than a numerical feature, thus it is difficult to combine the LBP features with other discriminative ones in a compact descriptor. Moreover, it produces higher dimensional features and is sensitive to Gaussian noise on flat regions [28].

II.3.5. Binary Robust Independent Elementary Features (BRIEF)

(BRIEF), a low-bitrate descriptor, is introduced for image matching with random forest and random ferns classifiers. It belongs to the family of binary descriptors such as LBP and BRISK, which only performs simple binary comparison test and uses Hamming distance instead of Euclidean or Mahalanobis distance. Briefly, for building a binary descriptor, it is only necessary to compare the intensity between two-pixel positions located around the detected interest points. This allows to obtain a representative description at very low computational cost. Besides, matching the binary descriptors requires only the computation of Hamming distances that can be executed very fast through XOR primitives on modern architectures [29].

The BRIEF algorithm relies on a relatively small number of intensity difference tests to represent an image patch as a binary string. More specifically, a binary descriptor for a patch of pixels of size S × S is built by concatenating the results of the following test

$$\tau = \begin{cases} \mathbf{1}, & \mathbf{if\ I(P_j) > I(P_i)}, \\ \mathbf{0} & \mathbf{otherwise}. \end{cases} \qquad \textbf{(II-18)}$$

where $I(p_i)$ denote the (smoothed) pixel intensity value at pi, and the selection of the location of all the pi uniquely defines a set of binary tests. The sampling points are drawn from a zero-mean isotropic Gaussian distribution with variance equal to $\frac{1}{25}S^2$.

For increasing the robustness of the descriptor, the patch of pixels is pre-smoothed with a Gaussian kernel with variance equal to 2 and size equal to 9 × 9 pixels. The BRIEF descriptor has two setting parameters: the number of binary pixel pairs and the binary threshold.

The experiments conducted by authors showed that only 256 bits, or even 128 bits, often suffice to obtain very good matching results. Thus, BRIEF is considered to be very efficient both to compute and to store in memory. Unfortunately, BRIEF descriptor is not robust against a rotation larger than 35◦ approximately, hence, it does not provide rotation invariance [39].

II.3.6. Other Feature Descriptors

A large number of other descriptors have been proposed in the literature and many of them have been proved to be effective in computer vision applications. For instance, color-based local features are four color descriptors based on color information proposed by Weijer and Schmid [30].

The Gabor representation or its variation,[31,32] has been also shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency.

Zernike moments and Steerable filters are also considered for feature extraction and description.

Steerable filters are also considered for feature extraction and description.

Inspired by Weber's Law, a dense descriptor computed for every pixel depending on both the local intensity variation and the magnitude of the center pixel's intensity called Weber Local Descriptor (WLD) is proposed in .

The WLD descriptor employs the advantages of SIFT in computing the histogram using the gradient and its orientation, and those of LBP in computational efficiency and smaller support regions. In contrast to the LBP descriptor, WLD first computes the salient micro patterns (i.e., differential excitation), and then builds statistics on these salient patterns along with the gradient orientation of the current point [33,34].

Two methods for extracting distinctive features from interest regions based on measuring the similarity between visual entities from images are presented.

The idea of these methods combines the powers of two well-known approaches, the SIFT descriptor and Local Self-Similarities (LSS). Two texture features called Local Self-Similarities (LSS, C) and Fast Local Self-Similarities (FLSS, C) based on Cartesian location grid, are extracted, which are the modified versions of the Local Self-Similarities feature based on Log-Polar location grid (LSS, LP). The LSS and FLSS features are used as the local features in the SIFT algorithm. The proposed LSS and FLSS descriptors use distribution-based histogram representation in each cell rather than choosing the maximal correlation value in each bucket in the Log-Polar location grid in the natural (LSS, LP) descriptor. Thus, they get more robust geometric transformations invariance and good photometric transformations invariance. A local image descriptor based on Histograms of the Second Order Gradients, namely HSOG is introduced in for capturing the curvature related geometric properties of the neural landscape.

Dalal and Triggs presented the Histogram of Oriented Gradient (HOG) descriptor, which combines both the properties of SIFT and GLOH descriptors. The main difference between HOG and SIFT is that the HOG descriptor is computed on a dense grid of uniformly spaced cells with overlapping local contrast normalization [36].

Following a different direction, Fan et al. proposed a method for interest region description, which pools local features based on their intensity orders in multiple support regions. Pooling by intensity orders is not only invariant to rotation and monotonic intensity changes, but also encodes ordinal information into a descriptor. By pooling two different kinds of local features, one based on gradients and the other on intensities, two descriptors are obtained: the Multisport Region Order-Based Gradient Histogram (MROGH) and the Multisport Region Rotation and Intensity Monotonic Invariant Descriptor (MRRID). The former combines information of intensity orders and gradient, while the latter is completely based on intensity orders, which makes it particularly suitable to large illumination changes. Several image features are analyzed [38].

Despite of the fact that, a large number of image feature descriptors have been introduced recently, several of these descriptors are exclusively designed for a specific application scenario such as object recognition, shape retrieval, or LADAR data processing. Furthermore, the authors of these descriptors evaluated their performance on a limited number of benchmarking datasets collected specifically for tasks. Consequently, it is very challenging for researchers to choose an appropriate descriptor for their application. In this respect, some recent studies compare the performance of several descriptors: interest region descriptors, binary descriptors , local color descriptors , and the 3D descriptors . In fact, claims that describing image features is a solved problem are overly bold and optimistic. On the other hand, claims that designing a descriptor for general real-world scenarios is next to impossible are simply too pessimistic, given the success of the descriptors in several applications. Finally, there is much work to be done in order to realize description algorithms that can be used for general applications [40]. We argue for further research towards using new modalities captured by 3D data and color information. For real time applications, a further path of future research would be the implementation of the algorithms on parallel processing units such as GPU [39].

II.4. Template based features matching

After extracting the key-points and its corresponding descriptor vector, for both detected and targeted images, feature correspondence must be established. Feature matching is the technique responsible for this. Matching can be established either by image intensity values of the pixels nearby, or by the distance between feature descriptor vectors depending on the specific registration method. The two major classifications under feature matching are area-Similarities measures- Correlation Measures [43].

II.4.1. Similarities measures:

In feature-based methods after the feature extraction step in both reference and detected image pairwise correspondence must be computed. The most commonly used method in this step are methods using spatial relations and the methods using invariant descriptors [43].

A. Methods using spatial relations

When the detected features are unable represented as a descriptor vector (the knowledge about the neighborhood is not properly available), then methods under the above classes are used [23].

B. Methods using invariant descriptors

Here in this method feature descriptor corresponds to the interest point is used for establishing the matching or correspondence between reference and sensed image. Feature descriptors which can handle the expected image deformation are preferably used. The descriptor which is used here should satisfy various criteria.[23] The most important property is the invariance (even if the sensed image undergoes rotation, scaling and/or any locally varying geometric deformations the descriptor vector of the corresponding key-points from the targeted and detected image must not show any difference), uniqueness (two distinct features shouldn’t have same descriptions), stability (the descriptor vector of a key-point and slightly deformed version of the same key-point in should be close to each other), and independence (the elements of the feature description vector should be mathematically independent). However, generally it is not required to satisfy all these criteria together and it is required to find out a suitable trade-off. Feature elements from the detected and target images with the maximum closer invariant descriptions are said to matching pairs. The choice of the type of the detector to be used depends on the feature properties and the expected geometric changes

occurred with the images. When trying to find out the best matching feature pairs between sensed and reference images, the minimum distance rule is often used [33].

II.4.2. Correlation Measures:

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 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 [36].

Sum of Absolute Differences (SAD) is one of the simplest of the measures which is calculated by subtracting pixels within a square neighborhood between the reference image I1 and the target image I2 followed by the aggregation of absolute differences within the square window; If the left and right images exactly match, the resultant will be zero . 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 [45]

Cross Correlation is even more complex to both SAD and SSD algorithms as it involves numerous multiplications, 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 [38].

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, several 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.

• Imposing symmetry condition to keep only pairs in which each point is the other's strongest match. This increases the chances that the two points in the matched pairs correspond to projections of the same physical scene point [47].

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

Table II.1: The most known correlation criteria

II.5. Descriptor Based features matching

II.5.1. SIFT/SURF descriptors-based features matching

In this method, the best candidate match for each key-point is found by identifying its nearest neighbor in the database of key-points from training images. The nearest neighbors 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 neighbor to the distance of the second closest [51].

Lowe rejected all matches in which the distance ratio is greater than 0.8, which eliminates90% of the false matches while discarding fewer than 5 % of the correct matches.

With the Nearest-neighbor 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 2 [41].

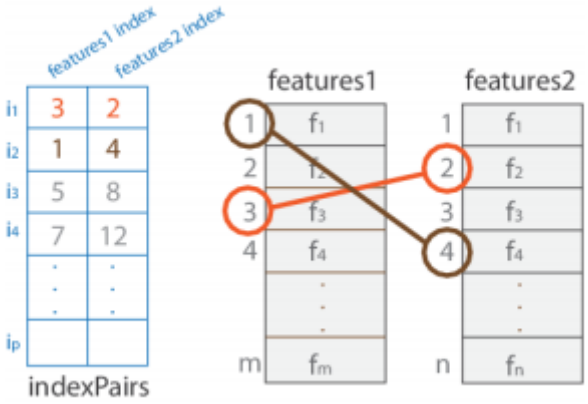


Fig.II.13. Index pairs of the matched features.

II.5.2. 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 suite vents 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 [52].

II.6. Conclusion :

In this chapter, we presented a very wide application of image registration. We have stated the descriptions of both classical and modern features with presentation of the uses and their principles. We also talked about the template and descriptor Based Features Matching where we highlighted the approaches of both SIFT / SURF and FREAK/ BRISK to give a review on the characteristics of each one.

In the next chapter we will present our tests results for multiple detections we have used, then we will visualize the simulation results of our experiments.

Chapter III:

Implementation Results

III. Implementation Results

III.1. Introduction

The idea behind feature detection is that you can run a test to determine whether a feature is supported in the current browser, and then conditionally run code to provide an acceptable experience both in browsers that *do* support the feature, and browsers that *don't*. If you don't do this, browsers that don't support the features you are using in your code won't display your sites properly and will just fail, creating a bad user experience.

III.2. Tools and Data Set

In this section we will present the development environment and the software used to develop our application.

III.2.1 Development environment

III.2.2 Hardware environment

- We used a computer that has the following characteristics:
- Type: PC/ Microsoft surface Pro 4.
- Processor: Intel® Core™i5-6300 CPU @ 2.40GHz 2.50GHz.
- Installed Memory (RAM) : 4.00Go.
- System Type: 64bits operating system, x64 processor.
- Camera: 5.0MP front camera, 8.0MP rear camera.

III.2.3 Software environment

The application has been implemented in JAVA language under the Netbeans 8.2 environment.

- 1.2.1. MATLAB: MATLAB (an abbreviation of "matrix laboratory") is a proprietary multi-paradigm programming language and numerical computing environment developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.

III.3. Description of the Interface

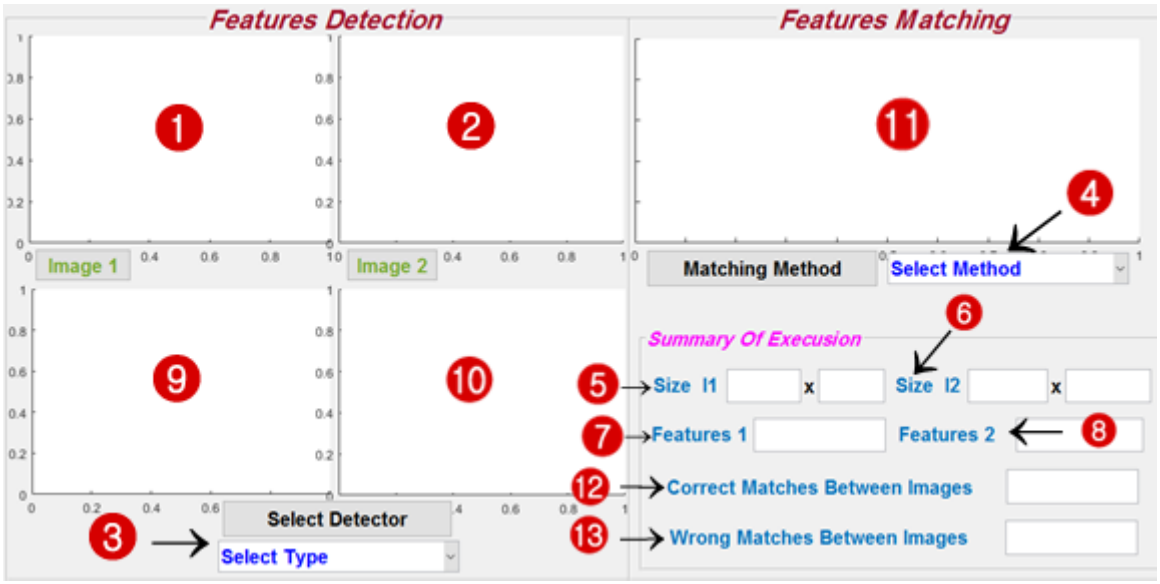


Figure 1II.1: Interface of our system

1. For uploading the first image.
2. For uploading the second image.
3. For selecting the type of Detector.
4. For selecting the type of Matching.
5. The size of the first image.
6. The size of the second image.
7. Features of the first image.
8. Features of the second image.
9. Display results of features detection in the first image.
10. Display results of features detection in second image.
11. Display features matching between image one and two.
12. Display correct matches between images.
13. Display wrong matches between images.

III.4. Implementation of Features Detection

II.4.1. Evaluation of Performance

As shown in the figures below, we have worked to discuss results of some feature detectors used in different pictures which are structured in a way that shows all the steps to use the interface, starting of showing uploading pictures until trying different detectors. And displaying results:

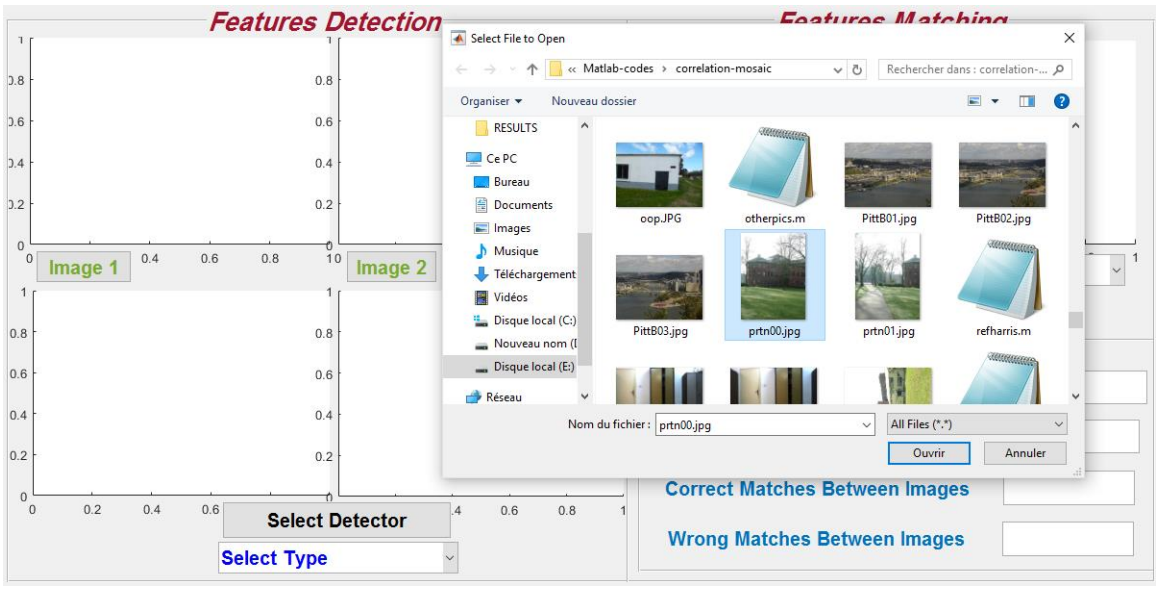


Figure1II.2: Uploading image 1

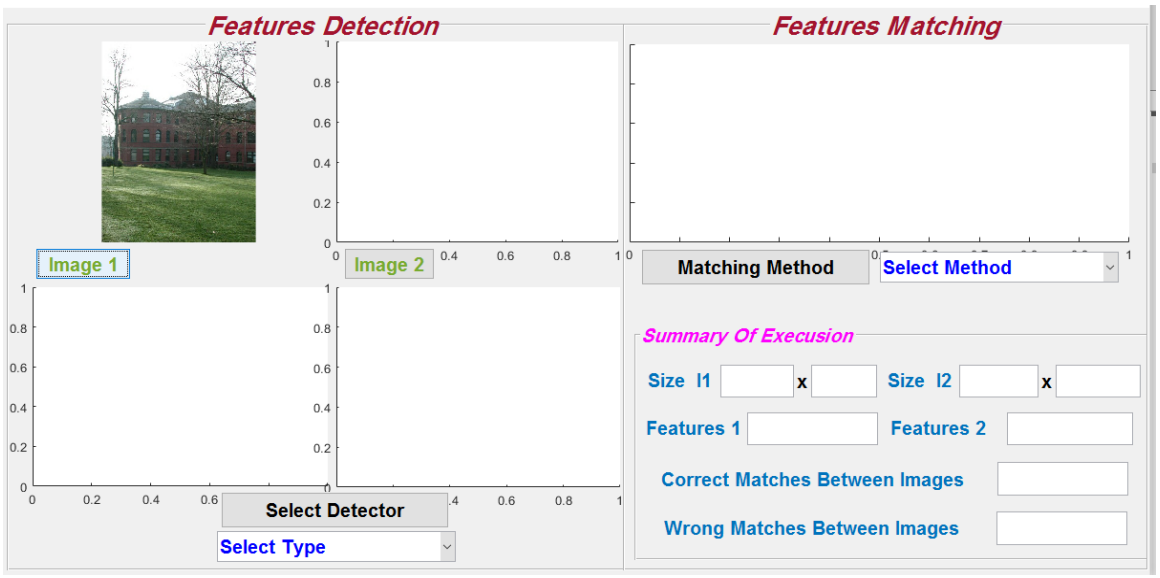


Figure 1II.3: Display image 1

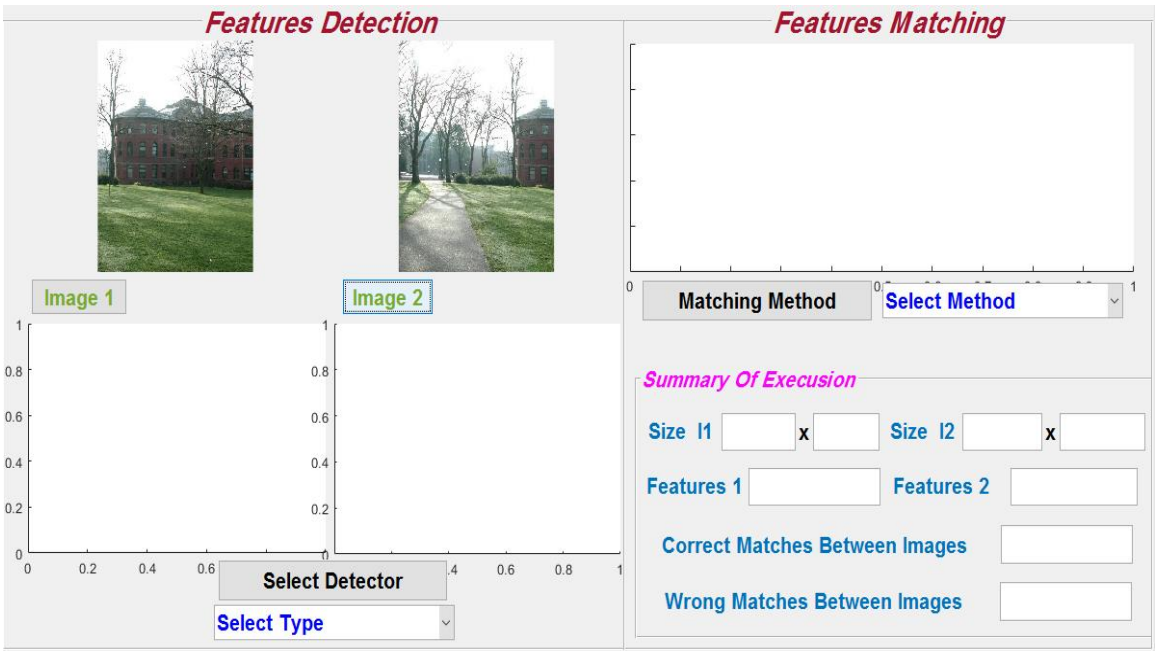


Figure 1II.4: Uploading and displaying image 2

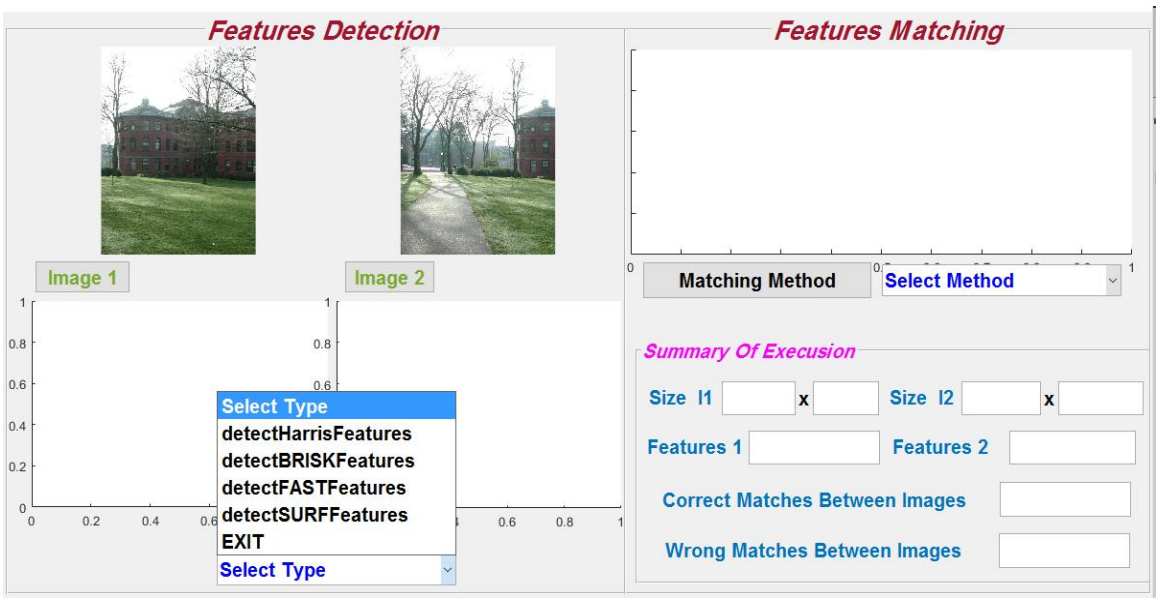


Figure 1II.5: Selecting detector type

Figure 1II.6: Selecting matching method

In the image below you can see the keypoints after using Harris detector:

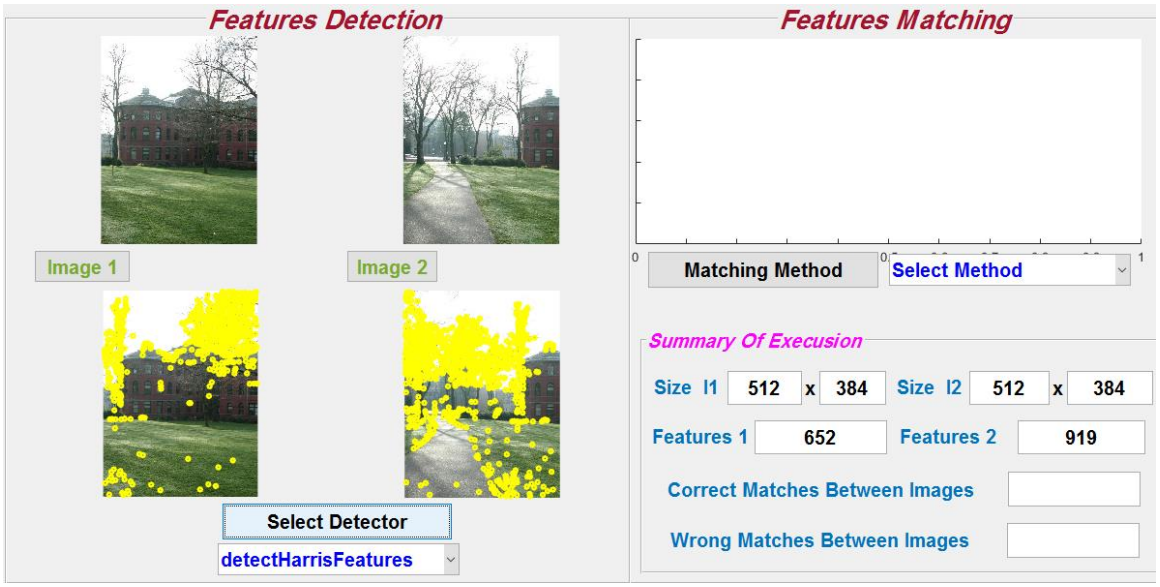


Figure 1II.7: Features Detection using Harris features

Trying BRISK detector now to see the variation of our options:

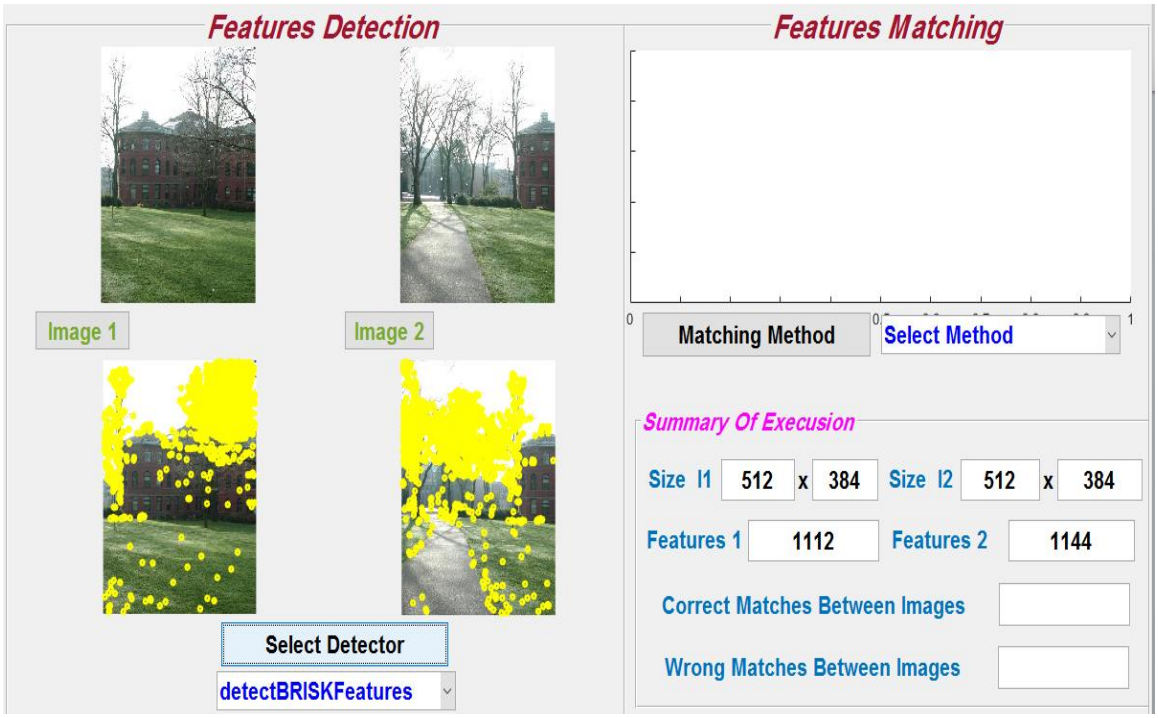


Figure 1II.8: Features Detection using BRISKF features

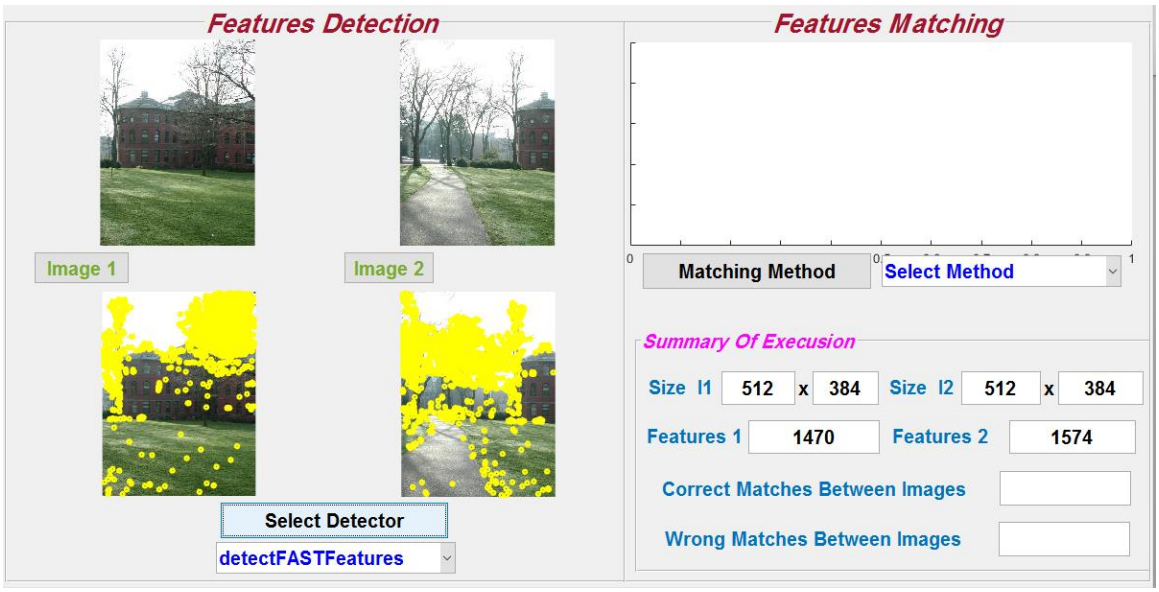


Figure 1II.9: Features detection using FAST features

In the image below when we used SURF detector, we can see that keypoints are different from the previous one:

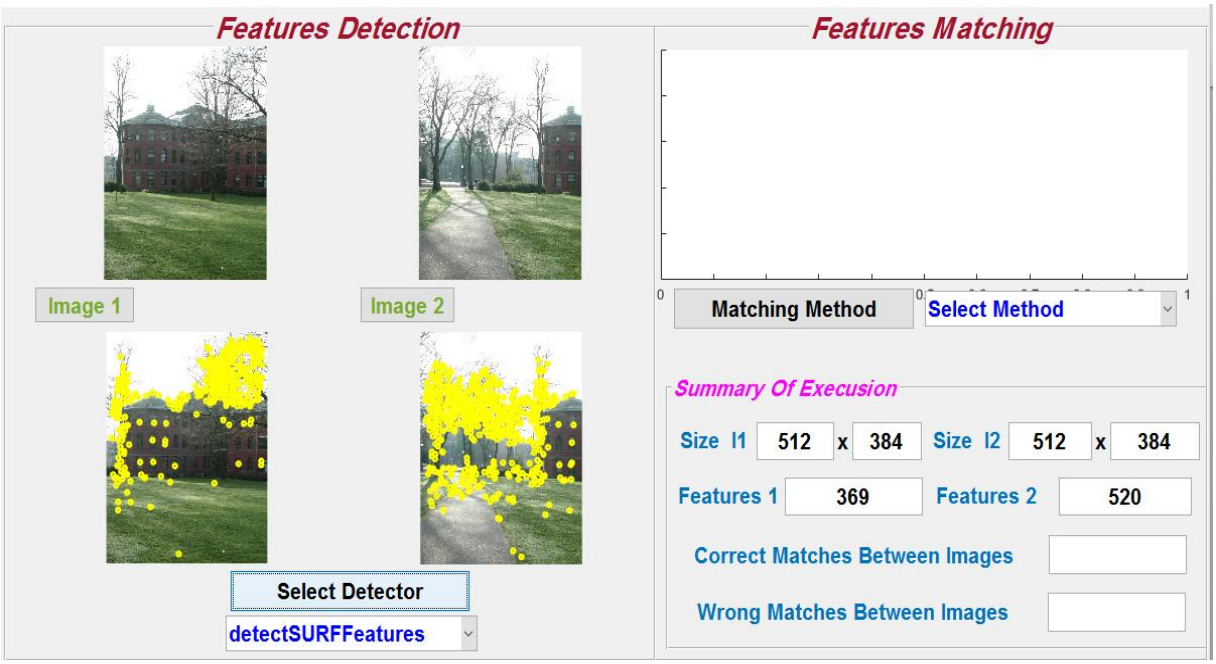


Figure 1II.10: Features detection using SURFF features

II.4. 2. Results Discussion

II.4.2.1 Advantages of feature detections

- SURF and SIFT are often considered to be the best feature detectors out there, for good reasons, they are very robust and very fast in most situations.
- Based on our work, we realized that the Harris corner detector provides good repeatability under changing illumination and rotation, and therefore, it is more often used in stereo matching and image database retrieval.
- As known, the SURF feature is a speed up version of SIFT, which uses an approximated DoG and the integral image trick.
- SIFT has good recall rates (accuracy), included in the OpenCV library, features are robust to occlusion and clutter and relatively efficient compared to older algorithms.

- The advantage of integral image is that after an image is computed into an integral image, it can compute block subtraction between any 2 blocks with just 6 calculations. With this advantage, finding SURF features could be several orders faster than the traditional SIFT features based on the results we analyzed.
- We discovered that the Harris / FREAK descriptor give better robust matching than the correlation method, applied to various categories of images.
- When we selected BRISK as a detector in our interface, we noticed that combined with the characteristics of BRISK and ORB, the algorithm is optimized and improved, which makes the algorithm have both excellent illumination robustness and fast computing power, and BRISK scale invariance. Under the condition of optimal algorithm parameters, the number of feature points can be increased by 3%, and the effective matching point can be improved by nearly 2%.
-
- We all know that Keypoint detection usually results in a large number of keypoints which are mostly clustered, redundant, and noisy. These keypoints often require special processing like Adaptive Non-Maximal Suppression (ANMS) to retain the most relevant ones that is which makes our choices limited.
- SIFT is still quite slow (SURF provides similar performance while running faster) and generally does not work well with lighting changes and blur and also it is not effective for low powered devices.
- During implementing our results, we know now that Fast-Hessian provide lower repeatability scores than the corner extractors (Harris and FAST).
- We noticed that the Harris Detector has lower accuracy.
- BRISK is rotation and scale invariant, but it takes more time to detect the feature points than other detectors we used. It has a behavior very similar to ORB with a little more cpu load since ORB in most cases works better in both terms of robustness and performances.
- On the other hand, FAST, as the name suggests, takes less time to detect the key points, but something we need to put in mind is that it is not scale invariant.

III.5. Implementation of Features Matching

II.5.1. Evaluation of Performance

Here as you can see in the following figures, we tried to use different matching features and see how we can improve our results by using the right type to obtain our objective:

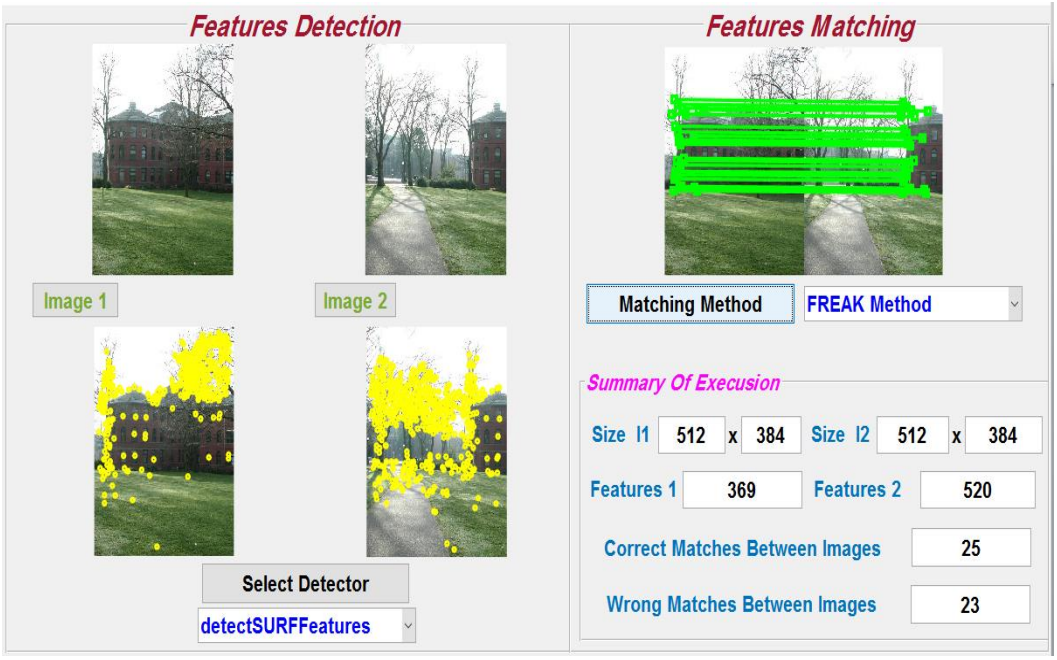


Figure 1II.11: Detect SURF features and FREAK matching method

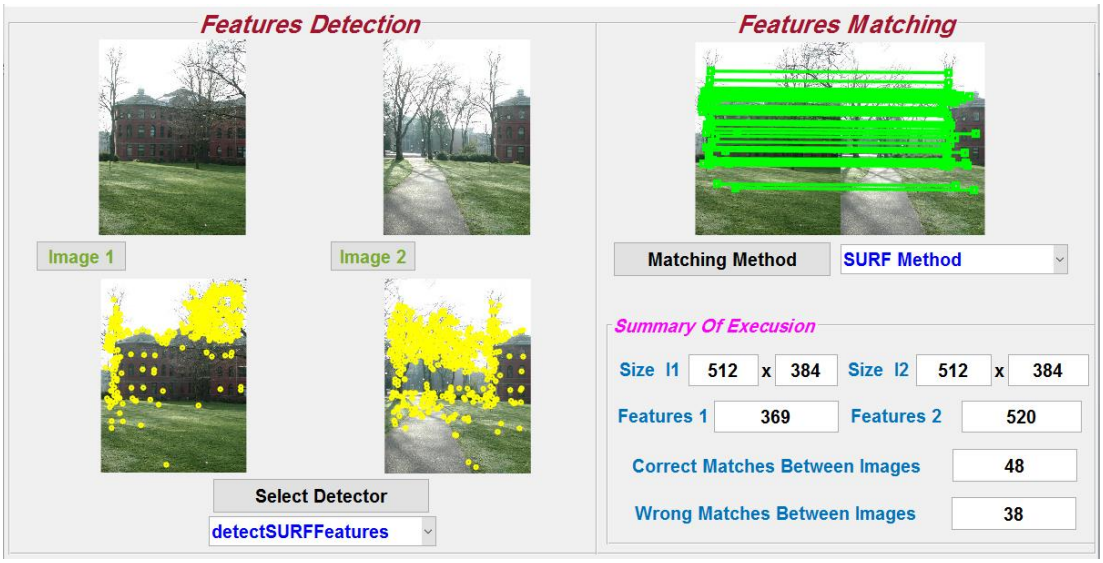


Figure 1II.12: Detect SURF features and SURF matching method

Here, we are matching between two images using SSD and SURF methods with Harris detector :

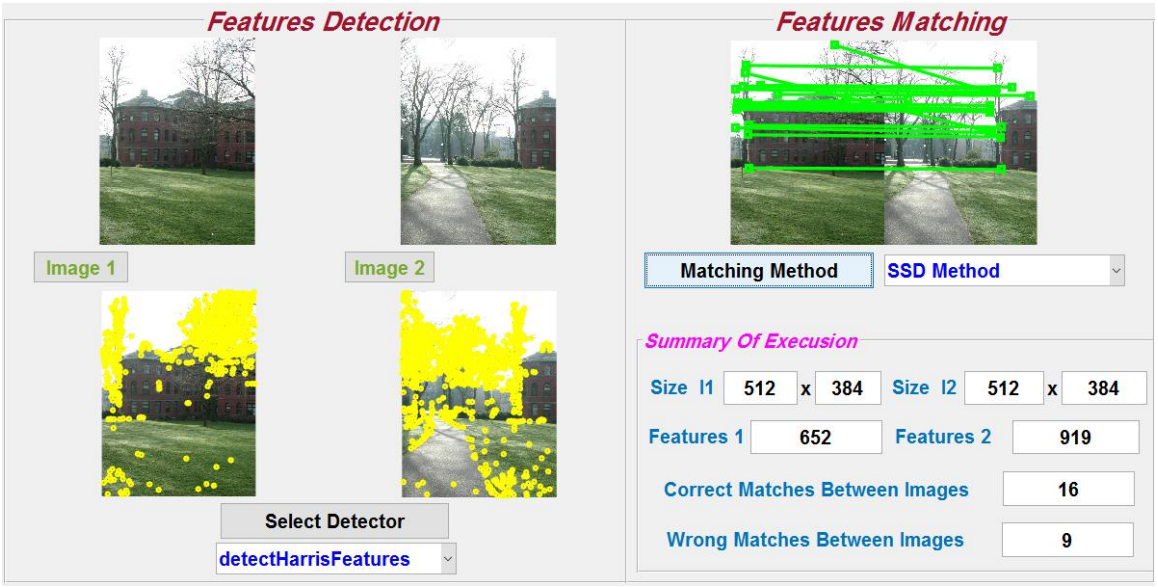


Figure 1II.13: Detect Harris features and SSD matching method

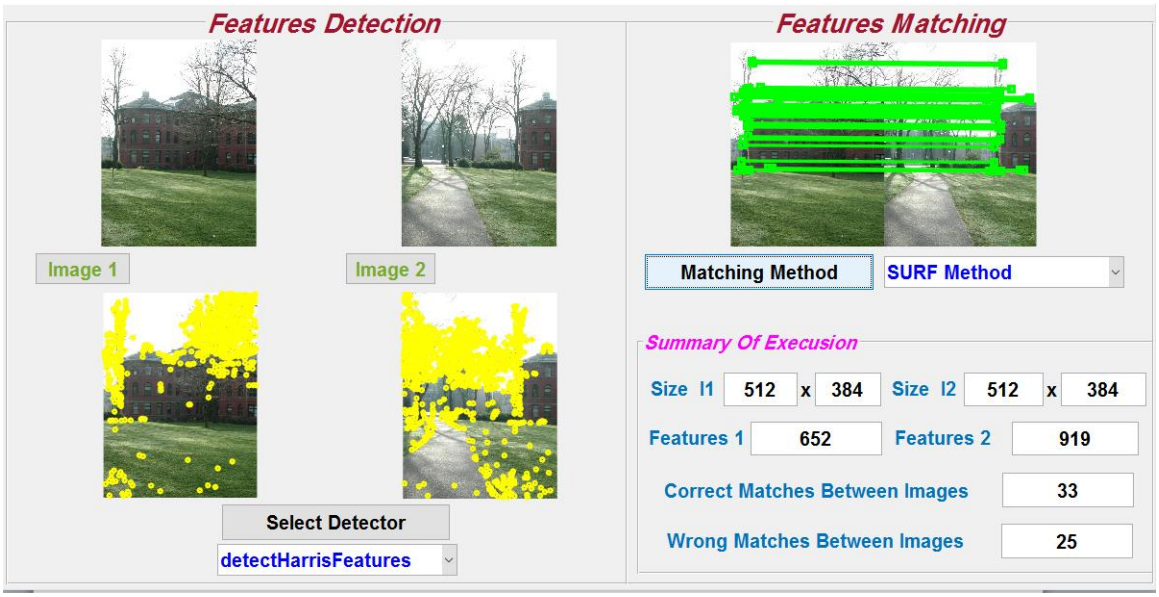


Figure 1II.14: Detect Harris features and SURF matching method

Matching between 2 images of a lab using SSD method:

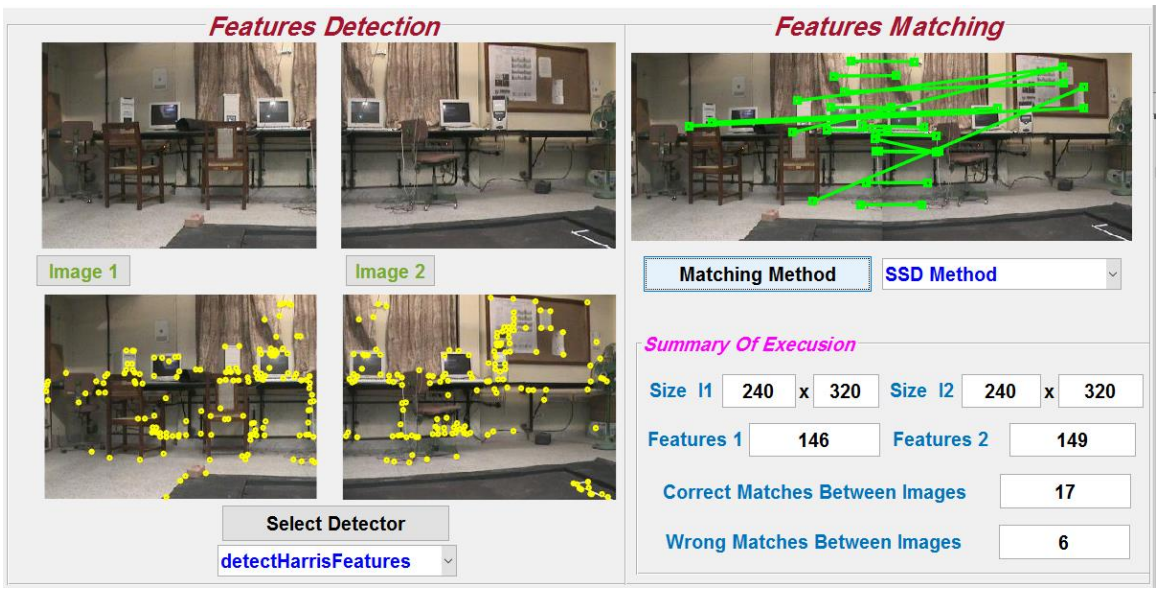


Figure 1II.15: Example 2 of detect Harris features and SSD matching method

Figure 1II.16: Example 2 of detect Harris features and SURF matching method

Figure 1II.17: Example 3 of detect Harris features and SSD matching method

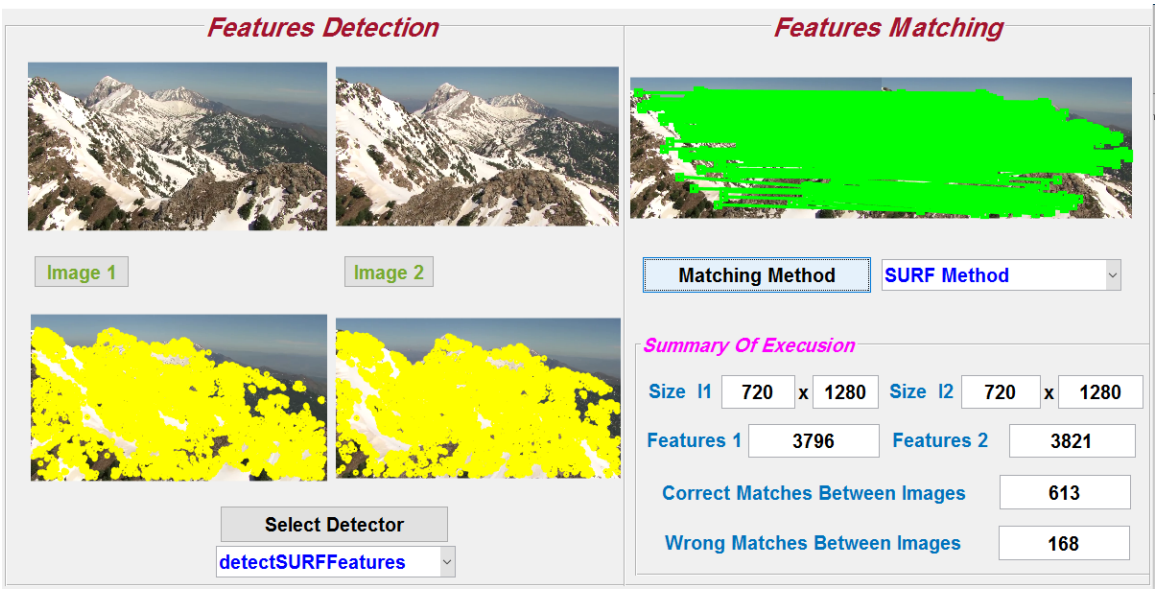


Figure 1II.18: Detect SURF features and SURF matching method

Figure 1II.19: Detect BRISK features and FREAK matching method

Figure1II.20: Detect FAST features and SSD matching method

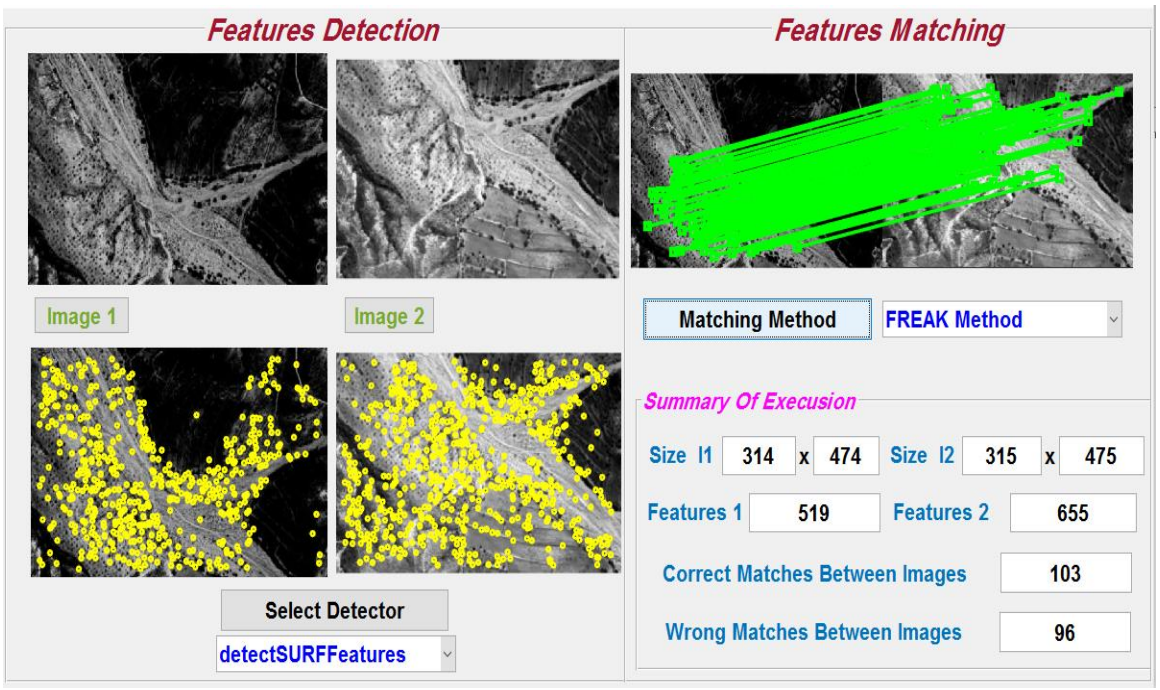


Figure 1II.21: Detect SURF features and FREAK matching method

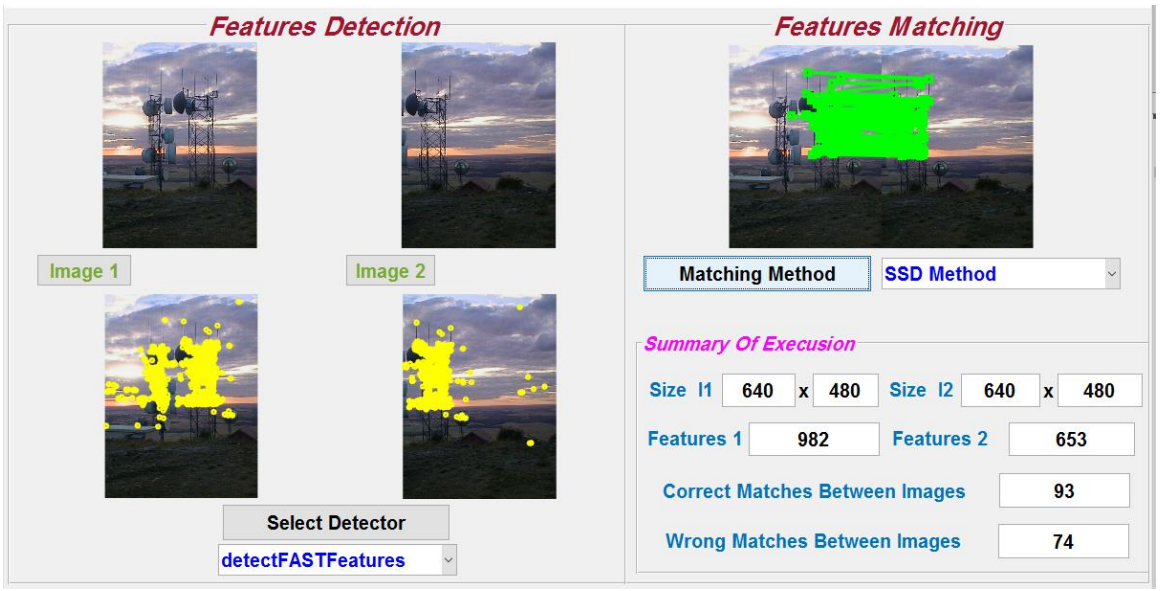


Figure 1II.22: Example 2 of detect FAST features and SSD matching method

II.5.2 Results Discussion

After presenting the figures about the features matching that we have selected to show the difference between each feature and the results we want to obtain. Based on the results and the displays of each figure we could test our experiment and conclude the strengths and weakness of each one.

II.5.2.1 Advantages of Feature Matching

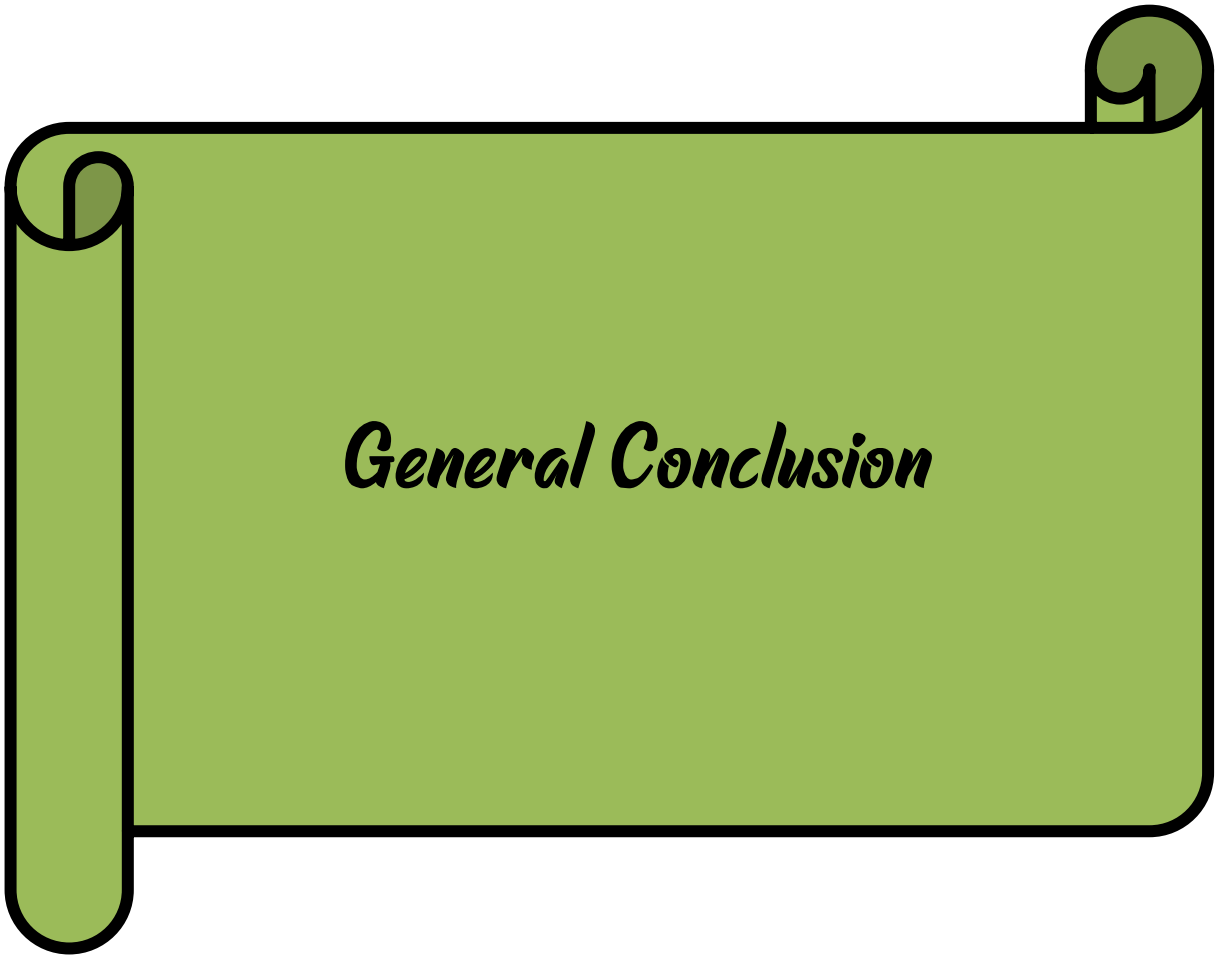
- One of the strengths we found in feature matching is that the calculation time is very satisfactory, for eventual use in real time.
- The computed gradient at the images edges was not necessarily equal to the true gradient. Changing the computation order can help improve it since we struggled in the first time and had unsatisfying results.
- Based on the examples we had. SURF is much faster than the 128-dimensional SIFT at the matching step.
- When we worked with SSD method, it turns out that it speeds up the process by eliminating the need of the region proposal network. To recover the drop in accuracy, SSD applies a few improvements including multi-scale features and default boxes. These improvements allow SSD to match the Faster R-CNN's accuracy using lower resolution images, which further pushes the speed higher.
- While working with FREAK descriptor our matching step takes advantage of the coarse-to-fine structured of it.

II.5.2.2 Areas of improvement

- For facial recognition uses the pretreatments used to improve skin detection can negatively influence feature extraction.
- Also, the Poor illumination can affect feature extraction.
- When we use very small images (face away) that have a very negative effect on feature extraction, which affects the whole system.
- While working on this part, we noticed that in some cases, Template Matching is not satisfactory for recognition.

II.6 Conclusion

In this chapter we have presented the design of our system in detail and the algorithms designed, the approaches implemented, as well as the interfaces of our application and some test results in the different cases. We can say that the realized program allows us to detect and match between different images where this modular evaluation provides insights into the weaknesses and strengths of individual detection and description algorithms. This can help improving existing algorithms to cope better with far-infrared imagery and define future research avenues to integrate the thermal modality in the field.

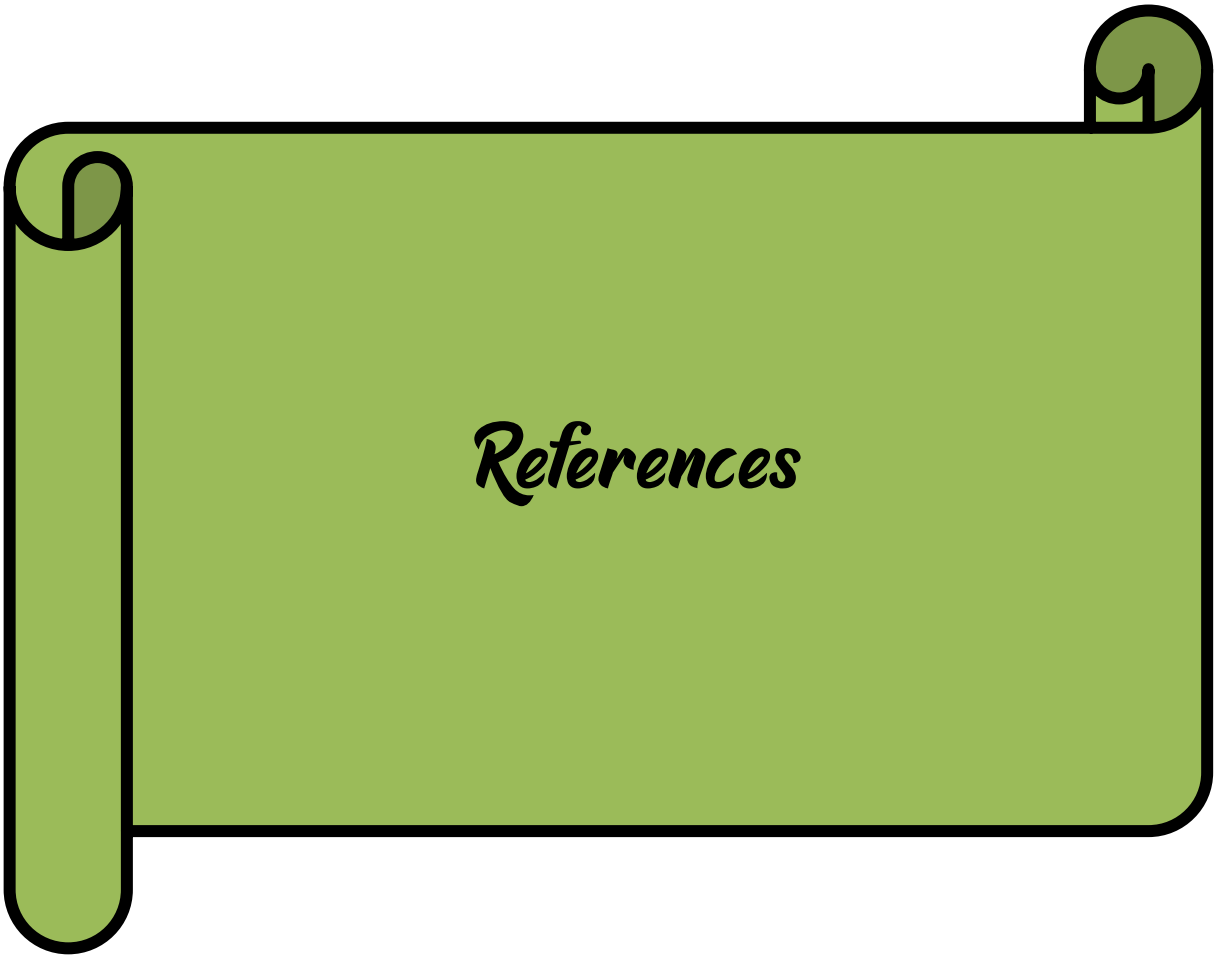

General Conclusion

General Conclusion

In this paper we have discussed image features where Image representation present an elementary problem in any image processing application. The straightforward method is to represent an image by point-to-point. Regarding biological tasks of image processing, such as recognition, retrieval, tracking and categorizing, such a method would be very uneconomical. The neighboring points are directly correlated with each other in natural images, so there exists a considerable number of redundancies in natural images. The biological image processing should compress these redundancies as much as possible, which would significantly benefit the following classification, recognition, or retrieval tasks. To achieve this goal, pictorial information should be processed in such a way that the most considerable possible proportion of redundant information is filtered out. In this chapter, we first summarize the state-of-the-art processing of image representation by arranging them into basic processing and advanced processing categories, resulting in basic features and advanced features, respectively. In addition, feature learning is investigated to generate more efficient features for biological image-processing tasks. The feature selection and feature extraction techniques are used in feature learning. Then we also talked modern feature detection and matching where we stated types, uses and descriptions f each one both classical and modern ones. And last and not least we gave an overview about the implementation we had done and results we discussed using our interface.

The experiments presented in this work provided interesting insights into the performance of different feature extraction and description algorithms applied to far-infrared imagery. From repeatability to the computation times, each algorithm showed weaknesses and strengths. In order to opt for one algorithm or a combination (detector/descriptor), one needs to state the specific requirements of the targeted application. This paper attempts to provide a visual comparative study of the existing feature detectors, namely, difference-of Harris, FAST, SURF and BRIKS. Detection of feature points between images will help solve the dilemma of increasing the effective area of the nonlinear loads. We performed feature detection using different detectors and then matched the extracted features to obtain the final results. The merits and demerits of the various detectors were realized. For instance, some form of compromise between robustness and speed is mandatory to reach one’s objectives.

As feature detection and match are two necessary steps in most computer vision tasks, we give a review on the algorithms of feature detection and match in this paper. We present the mathematical models and introduce the applications of each algorithm. To show the difference of their performances, we conduct a series of experiments and make a discussion about the results. Some conclusions can be acquired through our review and experiments. Harris and FAST are grayscale-based algorithms with no scale or rotation invariance and show poor performance in feature match. But FAST shows higher speed in detecting features that's why we recommend it in this area. In the other hand, Fast. SIFT, SURF shows better overall performance and prove suitable in feature match. To compare between SURF and SIFT, in terms of speed, we recommend SURF then SIFT, while in terms of accuracy, the order is SIFT, then SURF. SIFT has no advantage in terms of speed, is the best option if high accuracy is required. These conclusions can serve as a reference for research who want to select proper methods for tasks requiring feature extraction and match.


References

References

[1] J. Arrillaga, N.R. Watson, « Power system harmonics », Second Edition, John Wiley & Sons, Ltd ISBN: 0-470-85129-5. 2003.

[2] Pulkit Sharma. « Machine Learning and Deep Learning», Computer Vision Tutorial: A Step-by-Step Introduction to Image Segmentation Techniques (Part 1) URL https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python/

[3] Charter Global, « What is Image Binarization in AI? ». URL https://www.charterglobal.com/what-is-image-binarization-in-ai/

[4] Otsu, N., "A threshold selection method from gray-level histograms", IEEE Trans. Systems, Man, and Cybernetics, 9(1), pp.62–66 (1979).

[5] 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.

[6] R.S. Choras. Image feature extraction techniques and their applications for CBIR and

biometrics systems. International Journal of Biology and Biomedical Engineering, 6–16, 2007. URL http://www.naun.org/journals/bio/bio-2.pdf.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] Harris, C., Stephens, M.: A combined corner and edge detector. In: The Fourth Alvey Vision Conference, pp. 147–151. Manchester, UK (1988)

[18] Smith, S.M., Brady, J.M.: A new approach to low level image processing. Int. J. Comput. Vis. 23(1), 45–78 (1997)

[19] Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: International Conference on Computer Vision (ICCV'05), pp. 1508–1515. Beijing, China, 17–21 Oct2005

[20] Rosten, E., Drummond, T.: Machine learning for high speed corner detection. In: 9th European Conference on Computer Vision (ECCV'06), pp. 430–443. Graz, Austria, 7–13 May 2006

[21] Lowe, D.: Distinctive image features from scale-invariant keypoints, cascade filtering approach. IJCV 60 (2004) 91 – 110

[22] Department of Electrical Engineering National Institute of Technology, Rourkela-769008-2015

[23] Ojala, T., Pietikäinen, M., Mäenpää, M.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. 24(7), 971–987 (2002)

[24] Heikkiläa, M., Pietikäinen, M., Schmid, C.: Description of interest regions with local binary patterns. Pattern Recogn. 42(3), 425–436 (2009)

[25] Tian, H., Fang, Y., Zhao, Y., Lin, W., Ni, R., Zhu, Z.: Salient region detection by fusing bottom-Vup and top-down features extracted from a single image. IEEE Trans. Image Process. 23(10),V4389–4398 (2014)

[26] Huang, M., Mu, Z., Zeng, H., Huang, S.: Local image region description using orthogonalVsymmetric local ternary pattern. Pattern Recogn. Lett. 54(1), 56–62 (2015)

[27] Hong, X., Zhao, G., Pietikäinen, M., Chen, X.: Combining LBP difference and feature corre-lation for texture description. IEEE Trans. Image Process. 23(6), 2557–2568 (2014)

[28] Calonder, M., Lepetit, V., Özuysal, M., Trzcinski, T., Strecha, C., Fua, P.: BRIEF: computing a local binary descriptor very fast. IEEE Trans. Pattern Anal. Mach. Intell. 34(7), 1281–1298 (2012)

[29] Van De Weijer, J., Schmid, C.: Coloring local feature extraction. In: European Conference on Computer Vision (ECCV), pp. 334–348. Graz, Austria, 7–13 May 2006

[30] Subrahmanyam, M., Gonde, A.B., Maheshwari, R.P.: Color and texture features for image indexing and retrieval. In: IEEE International Advance Computing Conference (IACC), pp. 1411–1416. Patiala, India, 6–7 Mar 2009

[31] Zhang, Y., Tian, T., Tian, J., Gong, J., Ming, D.: A novel biologically inspired local feature descriptor. Biol. Cybern. 108(3), 275–290 (2014)

[32] Chen, Z., Sun, S.: A Zernike moment phase-based descriptor for local image representation and matching. IEEE Trans. Image Process. 19(1), 205–219 (2010)

[33] Chen, B., Shu, H., Zhang, H., Coatrieux, G., Luo, L., Coatrieux, J.: Combined invariants to similarity transformation and to blur using orthogonal Zernike moments. IEEE Trans. Image Process. 20(2), 345–360 (2011)

[34] Freeman, W., Adelson, E.: The design and use of steerable filters. IEEE Trans. Pattern Anal. Mach. Intell. 13(9), 891–906 (1991)

[35] Liu, J., Zeng, G., Fan, J.: Fast local self-similarity for describing interest regions. Pattern Recogn. Lett. 33(9), 1224–1235 (2012)

[36] Huang, D., Chao, Z., Yunhong, W., Liming, C.: HSOG: a novel local image descriptor based on histograms of the second-order gradients. IEEE Trans. Image Process. 23(11), 4680–4695 (2014)

[37] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Confer- ence on Computer Vision and Pattern Recognition (CVPR'05), pp. 886–893. San Diego, CA, USA, 20–26 June 2005

[38] Fan, B., Wu, F., Hu, Z.: Rotationally invariant descriptors using intensity order pooling. IEEE Trans. Pattern Anal. Mach. Intell. 34(10), 2031–2045 (2012)

[39] Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. 27(10), 1615–1630 (2005)

[40] Figat, J., Kornuta, T., Kasprzak, W.: Performance evaluation of binary descriptors of local features. Lect. Notes Comput. Sci. (LNCS) 8671, 187–194 (2014)

[41] Burghouts, G., Geusebroek, J.M.: Performance evaluation of local color invariantss. Comput. Vis. Image Underst. 113(1), 48–62 (2009)

[42] Moreels, P., Perona, P.: Evaluation of features detectors and descriptors based on 3D objects. Int. J. Comput. Vis. 73(2), 263–284 (2007)

[43] Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., Kwok, N.M.: A comprehensive performance evaluation of 3D local feature descriptors. Int. J. Comput. Vis. First online, 1–24(2015)

[44] A.Allouache. Construction 3d à partir de vues multiples.magister, laboratoire robo- tique et productique, ecole militaire polytechnique (emp). algérie. 2014.

[45] 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.

[46] Tinne Tuytelaars, Krystian Mikolajczyk, et al. Local invariant feature detectors: a survey. Foundations and trends R in computer graphics and vision, 3(3):177–280, 2008.

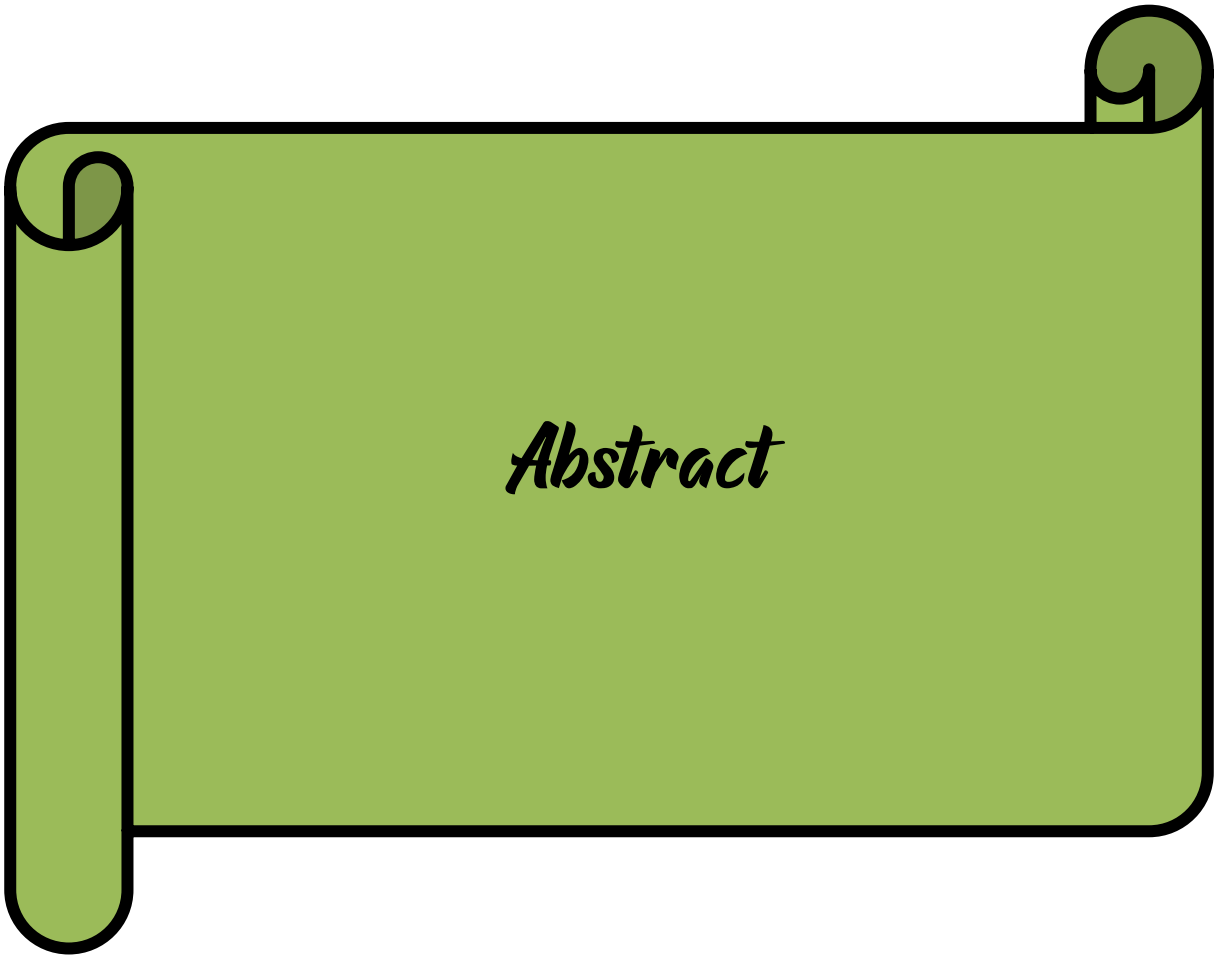
[47] Chris Harris and Mike Stephens. A combined corner and edge detector. In Alvey vision conference, volume 15, pages 10–5244. Citeseer, 1988.

[48] 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.

[49] ascal Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In International joint conference on arti- ficial intelligence (IJCAI), number CVLAB-CONF-1991-001, 1991

[50] A Nemra. Robust airborne 3D visual simultaneous localisation and mapping. PHD Thesis. Cranfield University, 55(4-5):345–376, 2011.

[52] Antonio Ezio Frascarelli. Object detection. master degree thesis . malardalen univer- sity sweden.


Abstract

Résumé

Dans les applications de vision par ordinateur, les primitives basées sur des points clés sont importantes et fréquemment utilisées dans les algorithmes de traitement d'image. Plusieurs techniques ont été développées dans les littératures pour la détection et l'association de primitives, et chaque approche présente certains avantages et inconvénients. Le détecteur de coin Harris est largement utilisé dans différents algorithmes d'ingénierie, puis vient SIFT (Scale Invariant Feature Transform) et SURF (Speeded-Up Robust Features) pour surmonter les inconvénients de la variation à grande échelle associée à l'algorithme de Harris. Les techniques de mise en correspondance des points clés appartiennent à deux catégories importantes ; l'un basé sur la corrélation et l'autre basé sur des descripteurs. Dans ce travail, nous proposons la mise en œuvre de différentes techniques de détection et de mise en correspondance de points clés sur Interface Matlab, l'interface graphique implémentée est facile à utiliser pour tous les utilisateurs et ne nécessite aucun apprentissage. Nous avons testé notre implémentation sur différentes scènes d'images et nous avons fait quelques discussions et conclusions.

Mots clés : Détection des points clés, correspondance des points clés, corrélation, descripteurs, interface Matlab.

Abstract

In computer vision applications, key points-based features are important and frequently used in image processing algorithms. Several techniques were developed in literatures for features detection and matching, and each approach has some advantages and drawbacks. Harris corner detector is widely used in different engineering algorithms, and then comes SIFT (Scale Invariant Feature Transform) and SURF (Speeded-Up Robust Features) to overcome disadvantages of large-scale variation associated with Harris algorithm. Features matching techniques are of two important categories: one based on correlation and other based on descriptors. In this work, we propose the implementation of various key points' detection and matching techniques on MATLAB Interface, the implemented graphical interface is easy to use for any users and does not require any learning. We have tested our implementation on different scenes of images, and we have done some discussions and conclusions.

Key words : Key points detection, Key points matching, Correlation, Descriptors, MATLAB Interface.

ملخص

في تطبيقات رؤية الكمبيوتر، تعتبر الميزات القائمة على النقاط الرئيسية مهمة وتستخدم بشكل متكرر في خوارزميات معالجة الصور. تم تطوير العديد من التقنيات في اعمال سابقة لالتقاط هذه النقاط و مطابقتها ، ولكل طريقة بعض المزايا والعيوب. يستخدم ملتقط الزوايا "هاريس" على نطاق واسع في خوارزميات هندسية مختلفة، ثم يأتي SIFT و SURF للتغلب على عيوب التباين واسع النطاق المرتبط بخوارزمية هاريس. تنقسم تقنيات مطابقة الميزات الى فئتين اساسيتين؛ واحد يعتمد على الارتباط والآخر يعتمد على الواصفات. في هذا العمل ، نقترح تنفيذ تقنيات اكتشاف ومطابقة النقاط الرئيسية المختلفة على واجهة MATLAB ، الواجهة الرسومية المنجزة سهلة الاستخدام لأي مستخدم ولا تتطلب أي تعلم. لقد اختبرنا تنفيذنا على مشاهد مختلفة للصور وقمنا ببعض المناقشات والاستنتاجات.

كلمات مفتاحية : التقاط النقاط الرئيسية ، مطابقة النقاط الرئيسية ، الارتباط ، الواصفات ، واجهة MATLAB .