FAST AND ROBUST IMAGE FEATURE MATCHING METHODS FOR COMPUTER VISION APPLICATIONS

Vom Fachbereich für Physik und Elektrotechnik

der Universität Bremen

zur Erlangung des akademischen Grades eines

Doktor-Ingenieur (Dr.-Ing.)

genehmigte Dissertation

von

M.Sc.-Ing. Faraj Alhwarin

aus Syrien

Referent:	Prof. Dr.-Ing. Axel Gräser
Korreferent:	Prof. Dr. Phil. Nat. Dieter Silber

Eingereicht am:	20. Januar 2011
Tag des Promationskolloquiums:	06. April 2011

Acknowledgements

I would first of all like to thank my doctoral father Prof. Dr. -Ing. Axel Gräser for giving me the valuable opportunity to work in this interesting research field, for his valuable suggestions and guidance during my doctoral research work and very thoughtful comments for the improvements of this thesis.

Further, I would like to thank Prof. Dr. Dieter Silber for being the second reviewer of this thesis and his very thoughtful comments. My thanks go also to Prof. Dr. –Ing. Walter Anheier and Prof. Dr. –Ing. Alberto Garcia Ortiz for showing interest to be on my dissertation committee.

I would like to thank my colleagues in the IAT institute, who were always there to assist me in providing critics and suggestions to my research work.
Especially I thank Dr. Danijela Risitc Durant for her kind support during the writing of this dissertation. She spent much time revising the manuscript and helped me with her insightful comments.

I would like to thank my wife Najed Alhwarin for being so patient and understanding during the difficult times while going through the doctoral research program. I appreciate all the sacrifices which she has made for me in order to accomplish this work.

Lastly, I would like to thank my family for their endless love that formed the most important part of my growing-up and for always being there when I needed them, helping me to face difficulties. Their endless support and encouragement has made the journey easier and one that I will treasure for many years to come.

Bremen, Mai 2011

Faraj Alhwarin

Abstract

Service robotic systems are designed to solve tasks such as recognizing and manipulating objects, understanding natural scenes, navigating in dynamic and populated environments. It's immediately evident that such tasks cannot be modeled in all necessary details as easy as it is with industrial robot tasks; therefore, service robotic system has to have the ability to sense and interact with the surrounding physical environment through a multitude of sensors and actuators.

Environment sensing is one of the core problems that limit the deployment of mobile service robots since existing sensing systems are either too slow or too expensive.

Visual sensing is the most promising way to provide a cost effective solution to the mobile robot sensing problem. It's usually achieved using one or several digital cameras placed on the robot or distributed in its environment. Digital cameras are information rich sensors and are relatively inexpensive and can be used to solve a number of key problems for robotics and other autonomous intelligent systems, such as visual servoing, robot navigation, object recognition, pose estimation, and much more. The key challenges to taking advantage of this powerful and inexpensive sensor is to come up with algorithms that can reliably and quickly extract and match the useful visual information necessary to automatically interpret the environment in real-time.

Although considerable research has been conducted in recent years on the development of algorithms for computer and robot vision problems, there are still open research challenges in the context of the reliability, accuracy and processing time.

Scale Invariant Feature Transform (SIFT) is one of the most widely used methods that has recently attracted much attention in the computer vision community due to the fact that SIFT features are highly distinctive, and invariant to scale, rotation and illumination changes. In addition, SIFT features are relatively easy to extract and to match against a large database of local features. Generally, there are two main drawbacks of SIFT algorithm, the first drawback is that the computational complexity of the algorithm increases rapidly with the number of key-points, especially at the matching step due to the high dimensionality of the SIFT feature descriptor. The other one is that the SIFT features are not robust to large viewpoint changes. These drawbacks limit the reasonable use of SIFT algorithm for robot vision applications since they require often real-time performance and dealing with large viewpoint changes.

This dissertation proposes three new approaches to address the constraints faced when using SIFT features for robot vision applications, Speeded up SIFT feature matching, robust SIFT feature matching and the inclusion of the closed loop control structure into object recognition and pose estimation systems.

The proposed methods are implemented and tested on the FRIEND II/III service robotic system. The achieved results are valuable to adapt SIFT algorithm to the robot vision applications.

Kurzfassung

Service Robot-Systeme sind entworfen, um Aufgaben wie das Erkennen und Bearbeiten von Objekten, das automatische Verstehen natürlicher Szenen und die Navigation in dynamischen, von Menschen bevölkerte Arbeitsumgebungen zu erledigen. Es ist unmittelbar einsichtig, dass diese Aufgaben nicht in allen notwendigen Details wie der Fall mit Industrierobotern modelliert werden können. Deshalb sollen Serviceroboter die Fähigkeit haben, mit der umgebenden physischen Umwelt durch eine Vielzahl von Sensoren und Aktoren agieren und reagieren zu können.

Die Umwelterfassung ist eine der wichtigsten Grundlagen autonomer Serviceroboter, die den kommerziellen Einsatz mobiler Serviceroboter beschränkt, weil Wahrnehmungssysteme entweder zu langsam oder zu teuer sind.

Visuelle Wahrnehmung ist die versprechendste Variante, um eine kostengünstige Lösung für das Wahrnehmungsproblem von mobilen Robotern darzustellen. Visuelle Wahrnehmung ist in der Regel mit einer oder mehreren digitalen Kameras auf dem Roboter montiert oder ist in seiner Arbeitsumgebung verteilt. Digitale Kameras sind Informationsreiche Sensoren und sind relativ günstig und können verwendet werden, um eine Reihe wichtiger Probleme für die Robotik und andere Autonome Intelligente Systeme durchzuführen, wie z. B. visuelle Servoing, Roboter-Navigation, Objekterkennung, Poseschätzung, und viele andere Anwendungen.

Die zentrale Herausforderung ist es, die diese leistungsstarken und kostengünstigen Sensoren mit Algorithmen zusammen kommen, die zuverlässig und schnell nützliche visuellen Informationen extrahieren und sie automatisch interpretieren können.

Obwohl beträchtliche Forschungen den letzten Jahren durchgeführt worden sind, um die Entwicklung von Algorithmen für Computer- und Robot Vision Probleme zu lösen, gibt es noch offene Forschungsfragen im Zusammenhang mit der Zuverlässigkeit, Genauigkeit und Aufwandzeit.

Skaleninvariante Bildmerkmalen (SIFT) ist eines der am häufigsten verwendeten Methoden, die heutzutage viel Aufmerksamkeit in den Computer-Vision-Community gewidmet werden, aufgrund der Tatsache, dass SIFT Features besonders ausgeprägt sind, und invariant bezüglich auf Skalierung, Rotation und die Beleuchtungsveränderungen sind. Darüber hinaus sind SIFT Features relativ leicht zu extrahieren und gegen eine große Datenbank von lokalen Merkmalen zu vergleichen. Im Allgemeinen, gibt es zwei wesentliche Nachteile von SIFT-Algorithmus: der erste Nachteil ist das die Komplexität des Algorithmus schnell steigt mit der Anzahl der Schlüssel-Punkte, vor allem an dem Matching-Schritt wegen der hohen Dimensionalität des SIFT Feature Deskriptors. Der andere ist, dass die SIFT Features nicht robust gegen große Blickwinkelveränderungen sind. Diese Nachteile beschränken die vernünftige Nutzung des SIFT- Algorithmus für Robot Vision- Anwendungen, da sie häufig Echtzeit-Leistung und den Umgang mit großer Blickwinkelveränderung erfordern. Diese Dissertation stellt drei neue Ansätze zur Bewältigung der Zwänge konfrontiert dar, wenn die SIFT Features für Robot Vision-Anwendungen verwendet werden wird es drei neue Ansätze geben: beschleunigte SIFT Feature Matching, robuste SIFT Feature Matching und die Einbeziehung des geschlossenen Regelkreises in der Objekterkennung und Kamerakalibrierungssysteme.

Die vorgeschlagenen Methoden sind implementiert und an dem FRIEND II/III Service-Robot-System getestet. Die erzielten Ergebnisse sind wertvoll für die Anpassung von SIFT-Algorithmus an den Roboter-Vision-Anwendungen.

Contents

List of Figures

List of Tables

1. Introduction

The primary goal in the field of service robotics is to design autonomous robots, which are capable to move around in the environment, to avoid obstacles, to recognize objects and to interact with them. Therefore service robotic system has to have the ability to sense and interact with the surrounding physical environment through a variety of sensors and actuators. The fundamental requirement for the solution of such problems is the 3D reconstruction of the environment, which means the determining the distance between the robot and its environment points.

Generally, the 3D reconstruction can be performed using active or passive sensing systems.

The active sensing systems can be classified based on the principle that is used to measure distances into time of flight-based [1] and triangulation-based systems [2].

The time-of-flight-based system is a scanner that uses laser light to probe a scene. The most popular type of time-of flight-based system is laser rangefinder. The laser rangefinder finds the distance of a surface by transmitting energy as laser light out into the robot environment, then measuring the return time of reflected energy. Since the speed of light is known, the round-trip time determines the travel distance of the light, which is twice the distance between the scanner and the object surface. The accuracy of a time-of-flight laser scanner depends on how exactly the time can be measured.

The laser rangefinder only detects the distance of one point in its direction of view. Thus, the scanner scans its entire field of view one point at a time by changing the range finder's direction to scan different points.

The triangulation-based system is also a scanner that uses laser light to investigate the environment. In terms of time-of-flight laser scanner the triangulation laser shines a laser on the subject and uses a camera to look for the position of the laser dot. Depending on how far away the laser strikes a surface, the laser dot appears at different places in the camera's field of view. This technique is called triangulation because the camera, laser emitter, and laser dot projected onto the object form a triangle. Since the distance between the emitter and the camera is known and the angle of the laser emitter corner is also known, the angle of the camera corner can be determined by looking at the location of the laser dot in the camera's field of view. These three pieces of information fully determine the shape and size of the triangle and gives the location of the laser dot corner of the triangle.

Scanning systems can produce highly accurate 3D measurements but tend to be expensive. Since scanners operate by scanning a single pixel with every pass and have mechanical components, they are bulky and slow especially when acquiring a significant field of view at useful resolutions.

Despite the passive systems such as stereo vision have a low real-time capability and have no homogenous depth map, they recently received a lot of attention due to their cheap costs.

Stereo vision method works similar to 3D perception in human vision by comparing the similarities and differences between two images and is based on triangulation between the pixels that correspond to the same scene structure projection on each of the images. Two images of the scene are sufficient in order to compute 3D depth information. If a 3D point in the world can be identified as a pixel location in an image, this world point lies on the line

passing by that pixel location and camera projection center. If we use two cameras, we can obtain two lines. The intersection of these lines is the 3D location of the world point.

In order to reconstruct the 3D environment of the robot using stereo vision, two problems have to be solved:

1. Identify pixels in images that match the same world point. This problem is known as the correspondence problem.
2. Identify the 3D coordinates of each pixel in the image and the camera projection center. This problem is known as the camera calibration problem. Camera calibration includes the determination of the optical parameters and the geometrical location of the camera.

Both problems are solved by image-matching techniques. Image matching techniques may find correspondences for only a sparse set of features in the image (feature-based image matching), or attempt to find correspondences for every pixel in the image (dense image matching) [3].

1.1. Motivation

Stereo vision relies on finding the corresponding points on two spatially separated images and then using triangulation to get the 3D measurement. This process of finding the corresponding points is sensitive to geometric and photometric transformations arising from illumination and viewpoint changes. The accuracy of the 3D results of stereo matching depends upon many factors such as image texture, image resolution, focal length and baseline distance. The increase in baseline improves the accuracy at long range but complicates the image matching problem and narrows the field of view (FoV). Higher image resolution increases the accuracy of the results but also may increase the processing time of image matching.

The scale invariant feature transform (SIFT) method proposed in [4] is currently the most widely used for image matching due to the fact that SIFT features are highly distinctive, and invariant to image translation, scaling, and rotation. SIFT features are also partially invariant to illumination changes and affine 3D projections. In addition, SIFT features are relatively easy to extract and to match.

Generally, there are two main drawbacks of SIFT algorithm, the first drawback is that the computational complexity of the algorithm increases rapidly with the number of key-points (high image resolution), especially at the matching step due to the high dimensionality of the SIFT feature descriptor. The other one is that the SIFT features are not robust to large viewpoint changes (wide-base line). These drawbacks limit the reasonable use of SIFT algorithm for robot vision applications since they require often real-time performance and need to deal with large viewpoint changes.

The goal of this dissertation is essentially to address the SIFT disadvantages preserving all its very important advantages. Specifically, we intend to improve SIFT's robustness to viewpoint changes and to accelerate SIFT feature matching, which is very important for robot vision applications.

1.2. Contributions

This thesis makes three main contributions. Firstly, it proposes a new strategy for fast SIFT feature matching by extending SIFT feature by some new attributes. Secondly, it introduces new method for robust SIFT feature matching. This method is based on the prioritized matching. Finally, it includes a fuzzy logic based closed loop system for precise object recognition, pose estimation, and camera calibration.

1. Speeded up SIFT Feature Matching.

Finding correspondences between SIFT features is the part of the matching algorithm that takes the most amount of processing time, especially when the number of features to be compared is relatively large. Most robot vision applications require real-time response. Unfortunately, the existing strategies for speeding up feature matching are inadequate for robot vision applications since they either work for offline matching such as Approximate Nearest Neighbor (ANN) searching methods or give insufficient acceleration such as PCA-SIFT [5], Speeded Up Robust Feature (SURF) [6], Fast Approximated SIFT (FA-SIFT) [74] and Reduced SIFT (R-SIFT) [7].

This thesis proposes a new strategy to speed up feature matching. This strategy is based on the classification of SIFT feature into several clusters through feature extraction phase based on several new introduced attributes computed from SIFT orientation histogram (SIFT-OH) or SIFT descriptor (SIFT-D). Thus, in the feature matching phase only features are compared that share almost the same corresponding attributes. This strategy has speeded up image matching by a factor of about 1000 according to exhaustive search, and has also improved the matching quality significantly.

2. Prioritized SIFT Feature Matching

Some robot vision tasks, such as camera calibration and pose estimation require robust feature matching.

Even though SIFT features are reasonably invariant, they can not accommodate large changes in viewpoint, witch is the core problem of camera calibration and pose estimation. This problem is caused by either the absence of true positive correspondences or their portion is insufficient for fitting methods to work correctly. This research introduces a new procedure to determine the scale factor between images to be matched by dividing SIFT features into different sub-sets based on their octaves. Then the matching process is done in prioritized order, so that only the features of the same scale ratio are compared on each step. At the same time a scale ratio histogram (SRH) is constructed. Only matches of the step corresponding to the highest SRH bin are provided to the fitting method. This restriction decreases the portion of outliers among positive matches leading to improve the performance of the fitting methods, such as Random Sample Consensus (RANSAC) [45] or Least Median of Squares (LMS) methods.

3. Fuzzy logic based closed Loop Control SIFT feature matching

In this research, a fuzzy logic-based closed loop control system is included to increase the accuracy of object recognition, pose estimation and camera calibration. The idea is to extract two different types of SIFT features, from model and query images. These features are

matched separately providing two independent affine transformations. The similarity between these transformations is used as a controlled value and passed to fuzzy controller to select one of these transformations to warp the model image. The matching process is repeated until a termination criterion is met.

1.3. Thesis Organization

The thesis is organized as follows: In chapter 2, basic concepts from the field of computer vision are provided. Firstly, a general description of the service robotic system FRIEND II/III is presented. Furthermore, backgrounds of stereo vision and camera calibration are briefly described, which are the common problems in many computer vision applications. As an example of robot vision applications, visual servoing is described. In chapter 3, image matching methods are briefly reviewed before focusing on the feature-based methods. We also review general aspects of feature extraction, description and matching. Chapter 4 presents SIFT algorithm in details, since it is the main concern of this thesis. In Chapter 5, firstly some aspects of the statistic of circular random variables are described and in this context a new theorem has been introduced and proven. Based on this theorem, several hashing methods are proposed to speed up SIFT feature matching. In Chapter 6, robust SIFT feature matching based on prioritized matching is presented to increase the invariance to affinity. In chapter 7 the inclusion of fuzzy-based closed loop control system for object recognition and pose estimation is demonstrated. Experimental results are included in each Chapter to demonstrate the efficacy of the proposed methods. Finally, Chapter 8 concludes this thesis and discusses possible extensions and future research directions.

2. Robot Vision Tasks

2.1. Service Robotic

The primary objective in service robotics is to design autonomous robots, which are able to move around in its environment, to recognize certain objects, to plan a motion to the destination of objects, possibly to grab them and to control the execution of the task. These systems should be able to work robustly in any environment without reconfiguration.

The area of service robotics has recently received significant attention. Service robots are used for many tasks such as cleaning, observing, and helping human in the carrying out of difficult tasks.

In more recent time, some of the most dominating efforts have been devoted to rehabilitation robotics that are designed to help elderly and disabled people in their activities of daily life, such as preparing and serving a cup of drink, picking up a telephone, or fetching and handling a book.

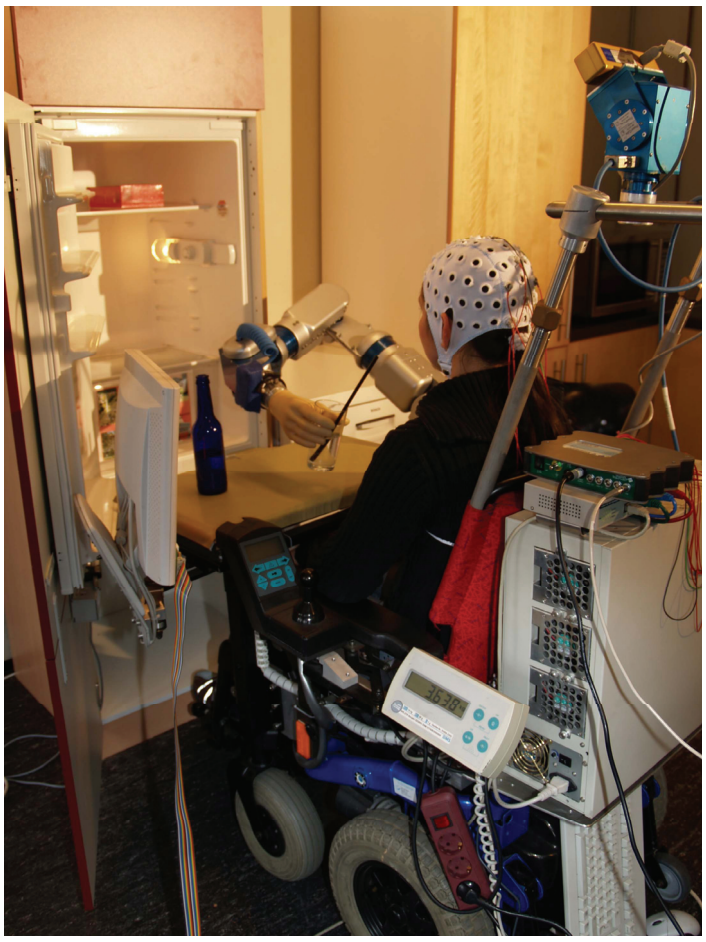


Figure 2.1: FRIEND III rehabilitation robotic system, developed at the University of Bremen, Institute of Automation

FRIEND (Functional Robot arm with frIENdly interface for Disabled people) [9] is a rehabilitation robot controlled based on visual sensing and designed to support disabled and elderly people in their daily life activities (Figure 2.1). FRIEND system has been developed at the Institute of Automation of University Bremen since 1997. FRIEND is equipped with an

electric wheelchair, a 7 degrees of freedom (7-DoF) mounted manipulator with a gripper and a multitude of sensors, including stereo vision system as core components attached to a pan-tilt head. Beside stereo vision system, the robot has additional local sensors that can increase overall robustness of the task executions, for instance, a force/torque sensor is built in a gripper base, witch can be used for contact detection when placing an object on table. The system has an intelligent tray consists of a sensory surface with infrared emitters and receivers.

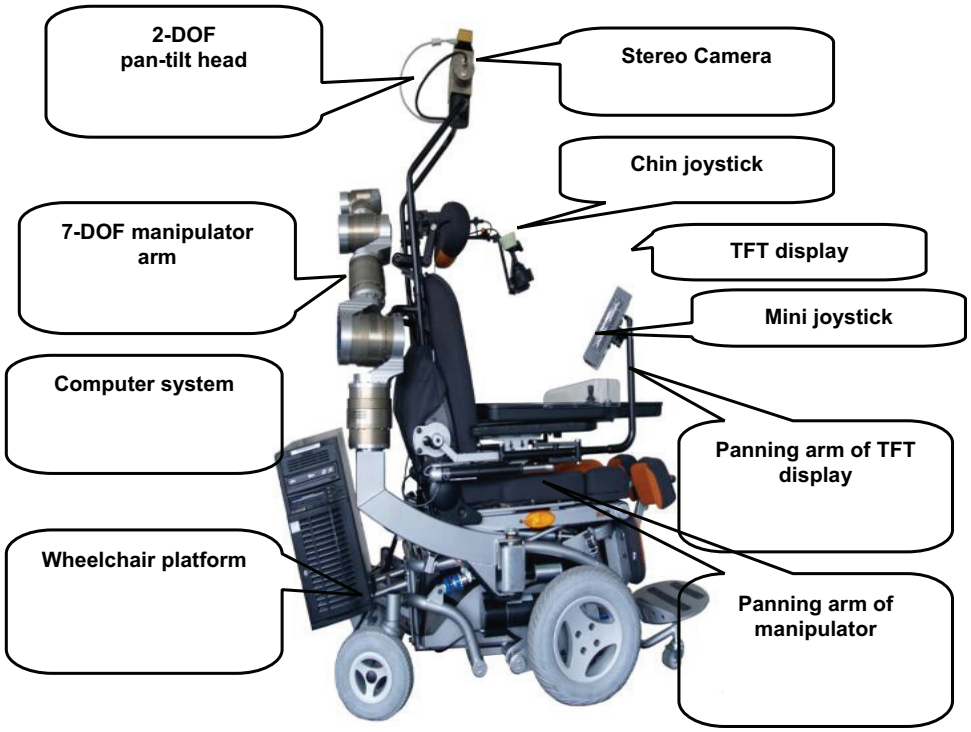


Figure 2.2: components of the rehabilitation robotic system FRIEND II.

For the human-machine interface purpose, the system is equipped with several input devices such as chin joystick, hand joystick, voice control, and brain computer interface (BCI). The input devices are adapted according to the impairments of the user or his preferences.

The objective of the rehabilitation robotic system FRIEND is to help disabled patients in their daily life activities. Thus, the robot operates in a human, unstructured environment, as depicted in the scene from Figure 2.1

To perform its tasks autonomously, the robot must be able to sense its environment witch is the task of the stereo vision system. The stereo vision system is a bumblebee stereo camera system with built-in calibration, synchronization and stereo projective calculation features is used to acquire information of the environment. It is mounted at the top of the robot system on a pan-tilt-head unit. Figure 2.2 presents the main components of the rehabilitation robotic system FRIEND II. For more details about FRIEND system the reader are referred to [9] [10] and [11].

2.2. Camera Calibration

The process of building the relationship between the world coordinate system and that of a captured image is called camera calibration. Camera calibration is a necessary step for many

computer vision applications especially for the functioning of robots that are meant to interact visually with the physical world. These robots can then use a video input device and calibrate in order to figure out where objects it sees might actually be in the real world, in actual terms of distance and direction. The relationship between the 3-D world coordinates and their corresponding image coordinates is usually described by two groups of parameters:

1. Intrinsic camera parameters (Internal geometric and optical characteristics of the camera).

2. Extrinsic camera parameters (position and orientation of the camera in the world coordinate system).

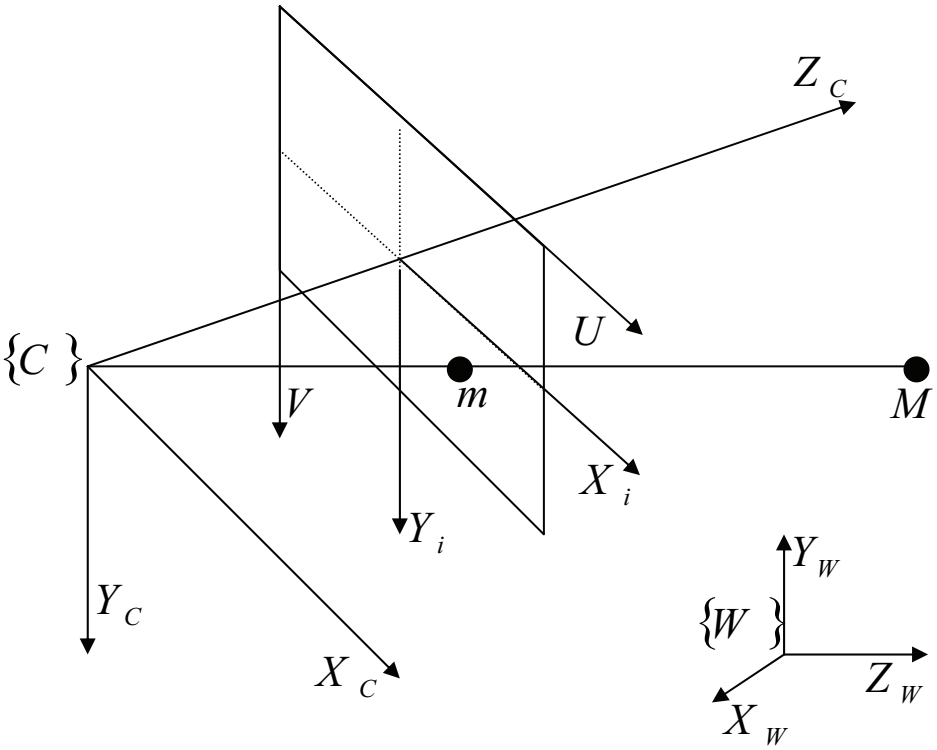


Figure 2.3: The coordinate systems involved in camera calibration.

2.2.1.Intrinsic Camera Parameters (Camera to Image)

Intrinsic camera parameters describe the optical and geometrical characteristics of the camera.

The camera coordinate system has its origin at the center of projection, its z axis along the optical axis, and its x and y axes parallel to the x and y axes of the image, as shown in Figure 2.3.

Assuming that a point M on an object with coordinates $[x_c, y_c, z_c]^T$ measured in the camera coordinate system, is imaged at the point $m(x_i, y_i)$ in the image plane. These coordinates are with respect to a coordinate system whose origin is at the intersection of the optical axis and the image plane, and whose X_i and Y_i axes are parallel to the X_c and Y_c axes. Camera coordinates and image coordinates are related by the perspective projection equations:

$$x_i = \frac{x_c \cdot f}{z_c} \quad and \quad y_i = \frac{y_c \cdot f}{z_c} \tag{2.1}$$

Where f is the focal length (distance from the center of projection to the image plane).

The actual pixel coordinates $m(u,v)$ are defined with respect to an origin in the top left hand corner of the image plane, and will satisfy:

$$u = u_0 + \frac{x_i}{w} \quad and \quad v = v_0 + \frac{y_i}{h} \tag{2.2}$$

where w and h are the width and the height of the pixel respectively.

By substituting equations (2.1) in (2.2) and multiplying both sides by z_c yields:

$$z_c \cdot u = z_c \cdot u_0 + \frac{x_c \cdot f}{w} \quad and \quad z_c \cdot v = z_c \cdot v_0 + \frac{y_c \cdot f}{h} \tag{2.3}$$

The equations (2-3) can be written linearly using the homogeneous coordinates as:

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} f/w & 0 & u_0 & 0 \\ 0 & f/h & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{2.4}$$

where the scaling factor s has value of z_c.

In short hand notation, we write equation (2.4) as

$$\widetilde{U} = K \cdot \widetilde{M}_c \tag{2.5}$$

where $\widetilde{U}$ represents the homogeneous vector of image pixel coordinates, K is the perspective projection matrix, and $\widetilde{M}_c$ is the homogeneous coordinates of a point measured in the camera coordinate system.

There are five camera parameters, namely the focal length *f*, the pixel width, the pixel height and the parameters u_0 and v_0 which are the u and v pixel coordinate at the optical center respectively. However, only four separable parameters can be solved for as there is an arbitrary scale factor involved in *f* and in the pixel size. Thus we can only solve for the ratios $\alpha_u = f/w$ and $\alpha_v = f/h$.

The parameters α_u, α_v, u_0 and v_0 do not depend on the position and orientation of the camera in space, therefore they are called the intrinsic parameters.

2.2.2.Extrinsic Camera Parameters (Camera to World)

A calibration target can be imaged to provide correspondences between points in the image and points in space. It is, however, generally impractical to position the calibration target accurately with respect to the camera coordinate system. As a result, the relationship between the world coordinate system and the camera coordinate system typically also needs to be recovered from the correspondences. The world coordinate system can be any system convenient for the particular design of the target.

Extrinsic camera parameters describe the relationship between a world coordinate system and the camera coordinate system. The transformation from world to camera consists of a rotation and a translation. This transformation has six degrees of freedom, three for rotation and three for translation.

If $M_w = [x_w, y_w, z_w]^T$ are the coordinates of a 3D point M measured in the world coordinate system and $M_c = [x_c, y_c, z_c]^T$ are the coordinates of the same point in the camera coordinate system, then the relationship between M_c and M_w is:

$$M_c = R \cdot M_w + T \tag{2.6}$$

The equation (2-6) can be rewritten in homogeneous coordinates as:

$$\widetilde{M}_c = [R|T] \cdot \widetilde{M}_w \tag{2.7}$$

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},\ T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix},\ [R|T] = \begin{bmatrix} R & T \\ 0_3^T & 1 \end{bmatrix}$$

where T is the translation vector capturing the camera displacement from the world frame origin and R is the rotation matrix encodes the camera orientation with respect to the world coordinate system.

By substituting the equation (2-5) in (2-7), we get the transformation between image and world coordinate system. This transformation is call projection matrix includes intrinsic and extrinsic camera parameters.

$$\widetilde{U} = \overbrace{K \cdot [R|T]}^{P} \cdot \widetilde{M}_w \tag{2.8}$$

2.3. Stereo Vision

The human vision and depth perception is based, in part, on the comparison between the two eyes' images. These two images represent two slightly different projections of the world in the retinas. The fusion of the two images from the right and left eye channel in the brain creates the sensation of depth.

Computer stereo vision tries to imitate this depth perception. The basic idea is to get two different images of the same scene acquired by stereo camera system from two different perspectives. A computer analyses the two images and tries to match them. Once the images have been brought into point-to-point correspondence, recovering depth by triangulation is straightforward; hence, the challenge in stereo vision is to find corresponding points in stereo images. This is a difficult task and time consuming; however, the complexity of this task can be reduced by precisely analysing the geometry of the stereo system configuration. The geometry describing stereo vision is called epipolar geometry.

2.3.1. Epipolar Geometry

The epipolar geometry describes the geometric relations between a 3D point and its projection in two cameras. Any point in the 3D world space together with the centers of projection of two cameras systems, defines an epipolar plane. The intersection of such a plane with an image plane is called an epipolar line as shown in Figure 2.4. Every point of a given epipolar line must correspond to a single point on the corresponding epipolar line. Therefore, the epipolar geometry can be used to constraint the search for corresponding image point in the first image to one dimensional neighborhood in the second image.

In order to present epipolar geometry mathematically, some definitions are needed:

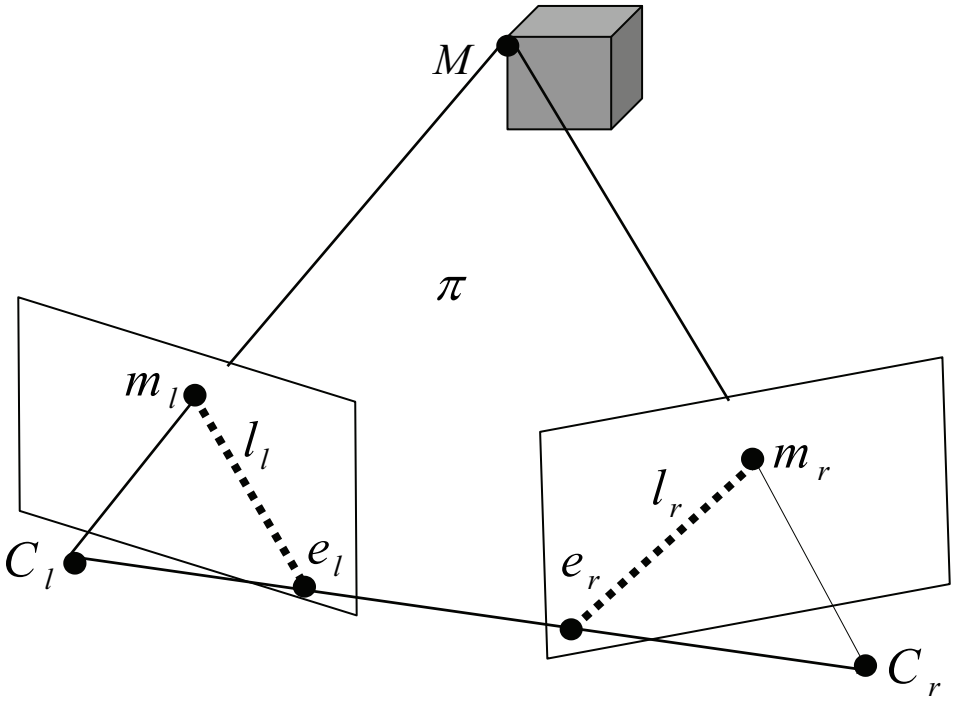


Figure 2.4: The epipolar geometry.

- **Epipole:** The projection of the center point of the left camera in the image plane of the right camera is called epipole. So, let e_l represents the image of right camera's center (C_r) in the left image. Similarly, e_r represents the image of left camera's center (C_l) in the right image. These points, e_l and e_r, are known as epipoles.
- **Epipolar line:** m_l is the image of M in the left camera. The line (e_l; m_l) in the left camera is called an epipolar line. This line is the projection of (C_r; M) in the left camera. The particularity of this line is that it is seen by the right camera as a point and by the left camera as a line that goes through the epipole e_l. So, all epipolar lines intersect at the epipole e_l. Symmetrically (e_r; m_r) defines an epipolar line in the right camera.
- **Epipolar plane:** C_l, C_r and M define an epipolar plane. The epipolar lines associated with M can be seen as the intersection of the epipolar plane with the image planes of the cameras.

The geometrical relations between epipoles, epipolar lines and epipolar planes can be expressed mathematically by introducing a matrix called fundamental matrix.

2.3.2.Fundamental Matrix

Fundamental Matrix F is the algebraic representation of the epipolar geometry between two cameras. This matrix captures the representation of the projective map from m in one image to its corresponding epipolar line in the other image.

The projection of any 3D point M in the left and right pinhole cameras can be written in matrix form:

$$\begin{aligned} m_l &= K_l \cdot M_l \\ m_r &= K_r \cdot M_r \end{aligned} \tag{2.9}$$

where K_l and K_r are respectively the projective matrix of the left and right camera. M_l and M_r are the coordinates of M in the left and right camera coordinate systems respectively.

The coordinate system of the right camera can be transformed into the coordinate system of the left camera through a rotation R and a translation T. Therefore equation (2-9) can be rewritten as:

$$\begin{aligned} m_l &= K_l \cdot \begin{bmatrix} I & 0 \end{bmatrix} \cdot M_l \\ m_r &= K_r \cdot \begin{bmatrix} R & t \end{bmatrix} \cdot M_l \end{aligned} \tag{2.10}$$

These equations can be combined to remove M_l.

$$m_r = \overbrace{K_r \cdot \begin{bmatrix} R|T \end{bmatrix} \cdot K_l^{-1}}^{H} \cdot m_l \tag{2.11}$$

The matrix that maps each pixel in the left image to exactly one corresponding pixel in the right image is called the homography matrix H.

Since each epipole line l has both the corresponding image point m and the epipole e on it, it is defined as:

$$\begin{aligned} l_l &= e_l \times m_l = \begin{bmatrix} e_l \end{bmatrix}_\times \cdot m_l \\ l_r &= e_r \times m_r = \begin{bmatrix} e_r \end{bmatrix}_\times \cdot m_r \end{aligned} \tag{2.12}$$

However, we just saw that H is the transfer mapping of m_l to m_r, this can be written as:

$$\begin{aligned} l_l &= \overbrace{\begin{bmatrix} e_l \end{bmatrix}_\times \cdot H^{-1}}^{F} \cdot m_r \\ l_r &= \overbrace{\begin{bmatrix} e_r \end{bmatrix}_\times \cdot H}^{F} \cdot m_l \end{aligned} \tag{2.13}$$

where $\begin{bmatrix} e \end{bmatrix}_\times$ is the vector product matrix associated with the epipole e:

$$[e]_\times = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix} \tag{2.14}$$

This matrix depends on the intrinsic matrix of the cameras (C_l and C_r) and on their relative position (R and T). Hence, in a static setup where the relative position of the cameras is known and where the cameras have been calibrated, i.e. their intrinsic matrices are known, the fundamental matrix can be computed once and for all.

If the cameras are not calibrated, the fundamental matrix can be estimated using a fitting algorithm from n>8 correspondences points.

$$F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \Rightarrow f^T = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{21} & f_{22} & f_{23} & f_{31} & f_{32} & f_{33} \end{bmatrix} \tag{2.15}$$

Each corresponding point pairs $(x_1, y_1, 1)$ and $(x_2, y_2, 1)$ gives an equation:

$$\begin{bmatrix} x_1 \cdot x_2 & y_1 \cdot x_2 & x_2 & x_1 \cdot y_2 & y_1 \cdot y_2 & y_2 & x_1 & y_1 & 1 \end{bmatrix} \cdot f = 0 \tag{2.16}$$

Stacking n equations from n point correspondences gives linear system $A \cdot f = 0$, where A is an $n \times 9$ matrix.

If rank $A = 8$ then the solution is unique (up to scale) but in reality we seek a least-squares (LS) solution with $n \geq 8$. Then LS solution is the last column of the matrix V in the singular value decomposition (SVD) of matrix A:

$$A = U \cdot D \cdot V^T \tag{2.17}$$

which corresponds to the smallest singular value.

2.3.3. Triangulation

The triangulation is a process to reconstruct the 3D coordinates of a point from its 2D images. Each point in an image plane corresponds to a 3D line in world space which passes through this point and the center of projection of the camera. If two corresponding points in two images are the projection of a common 3D world point M, then the associated 3D lines must intersect at M.

In practice, however, the coordinates of image points cannot be measured with arbitrary accuracy. Instead, various types of noise, such as geometric noise from lens distortion or interest point detection error lead to inaccuracies in the measured image coordinates. As a consequence, the 3D lines do not always intersect in world space. The problem, then, is to find a 3D point which optimally fits the measured image points. In the literature there are multiple proposals for how to define optimality and how to find the optimal 3D point such as Mid-point method or Direct Linear Transformation (DLT) [12].

The 3D position (X, Y, Z) of a point M, can be reconstructed from the perspective projection

of M on the image planes of the cameras, once the relative position and orientation of the two cameras are known. We choose the 3D reference system to be the left camera system. The right camera is translated and rotated with respect to the left camera.

There are two key configurations of stereo vision systems: parallel and non-parallel. In a parallel configuration, the optical axes of two cameras are parallel, and the translation of the right camera is only along the X axis. In a non-parallel configuration, the optical axes of two cameras are non-parallel and the right camera can be located arbitrary with respect to the left camera.

1.1.1.1 Parallel Cameras

If the optical axes of two cameras are parallel, and the translation of the right camera is only along the X axis, the correspondence points lie on the same horizontal line; therefore the correspondence problem becomes a one-dimensional search along corresponding lines.

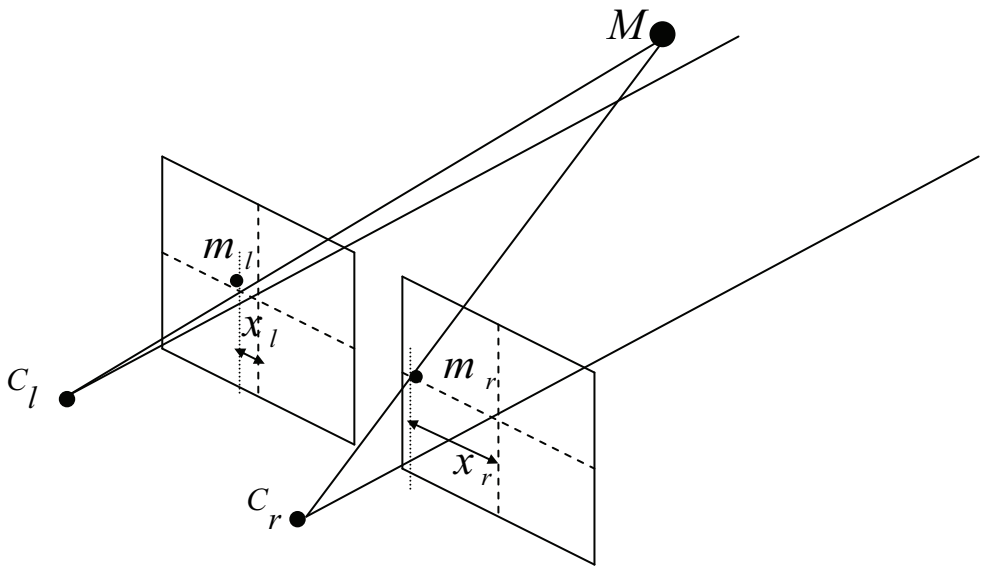


Figure 2.5: Parallel stereo vision system.

The offset between a pixel in the left and its corresponding pixel in the right image is called disparity.

$$D = x_l - x_r \tag{2.18}$$

Once the disparity values are known, the world coordinates of a point can be computed as.

$$\begin{aligned} Z &= \frac{b \cdot f}{D} \\ X &= \frac{x_l \cdot Z}{f} \\ Y &= \frac{y_l \cdot Z}{f} \end{aligned} \tag{2.19}$$

where f is the focal length of both cameras and b is the distance between the two camera projection centers (baseline).

In this configuration the matching process is very simple, but the accuracy of 3D coordinates and the maximum depth that can be measured depend on the length of the baseline. High accuracy would require a longer baseline, which causes a reduction in the common field of view (FoV), so that only a smaller portion of the scene is visible.

The contradiction between the accuracy of 3D reconstruction and the size of the common FoV can be exceeded using the non-parallel configuration.

1.1.1.2 Non-Parallel Cameras

In the non-parallel configuration, the right camera can be translated and rotated with respect to the left one in three directions. Given the translation vector T and rotation matrix R describing the transformation from left camera to right camera coordinates, the equation to solve for stereo triangulation is:

In the non-parallel configuratio, the right camera can be translated and rotated with respect to the left one in three directions. Given the translation vector T and rotation matrix R describing the transformation from left camera to right camera coordinates the equation to solve for stereo triangulation is:

$$m_r = R^T\left(m_l - T\right) \tag{2.20}$$

where m_l and m_r are the coordinates of M in the left and right camera coordinates respectively, and R^T is the transpose (or the inverse) matrix of R.

If a point $M(X,Y,Z)$ in 3D space is given, with two cameras it can be separately projected to two points $m_l(x_l, y_l)$ and $m_r(x_r, y_r)$ respectively. Eventually if m_l and m_r are known, then a line can connect m_l and the projection center of the left camera C_l. Similarly, an other line can connect m_r and C_r. It is obvious that M must be on the line intersection.

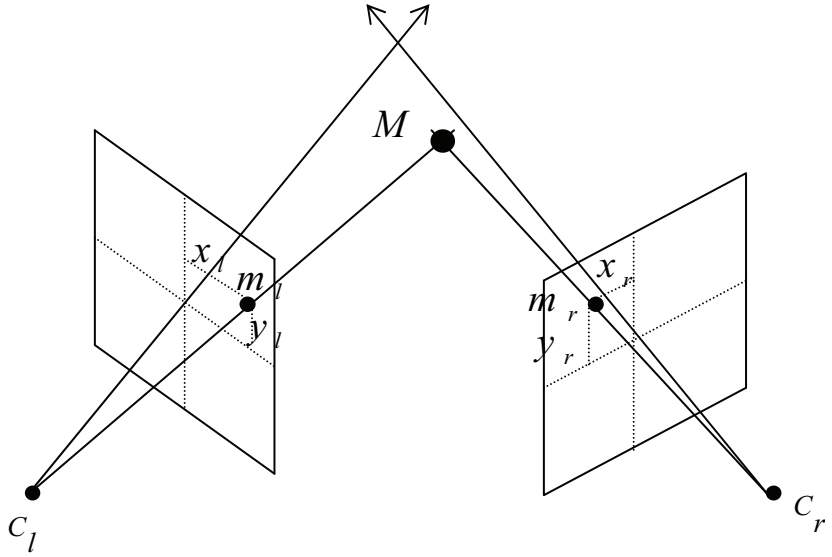


Figure 2.6: Non-Parallel stereo vision system.

The relationships between 3D world point and its images are given as:

$$\begin{aligned} m_l &= P_l \cdot M \\ m_r &= P_r \cdot M \end{aligned} \tag{2-21}$$

where P_l and P_r are the left and the right projection matrices respectively

The above equations can be combined into a form $AM = 0$ which is a linear equation system in M.

The homogeneous scale factor can be eliminated by a cross product which gives three equations for each image point on the left and right stereo images. This can be mathematically expressed as:

$$\begin{aligned} m_l \times P_l \cdot M &= \left[m_l\right]_x \cdot P_l \cdot M = 0 \\ m_r \times P_r \cdot M &= \left[m_r\right]_x \cdot P_r \cdot M = 0 \end{aligned} \tag{2.22}$$

Expanding equations (2.22), we get:

$$\begin{aligned} &x_l\left(p_{l3}^T \cdot M\right) - \left(p_{l1}^T \cdot M\right) = 0 \\ &y_l\left(p_{l3}^T \cdot M\right) - \left(p_{l2}^T \cdot M\right) = 0 \\ &x_l\left(p_{l2}^T \cdot M\right) - y_l\left(p_{l1}^T \cdot M\right) = 0 \\ &x_r\left(p_{r3}^T \cdot M\right) - \left(p_{r1}^T \cdot M\right) = 0 \\ &y_r\left(p_{r3}^T \cdot M\right) - \left(p_{r2}^T \cdot M\right) = 0 \\ &x_r\left(p_{r2}^T \cdot M\right) - \left(p_{r1}^T \cdot M\right) = 0 \end{aligned} \tag{2.23}$$

where p_{li}^T and p_{ri}^T for $\{i = 0,1,2,3\}$ are the rows to considered the left and the right projection matrices respectively.

Since the equations (2.23) are linear in the components of M, an equation of form $AM = 0$ can be then composed as described in equation (2.24).

$$\overbrace{\begin{bmatrix} x_l\left(p_{l3}^T\right) - \left(p_{l1}^T\right) \\ y_l\left(p_{l3}^T\right) - \left(p_{l2}^T\right) \\ x_r\left(p_{r3}^T\right) - \left(p_{r1}^T\right) \\ y_r\left(p_{r3}^T\right) - \left(p_{r2}^T\right) \end{bmatrix}}^{A} \cdot M = 0 \tag{2.24}$$

As described previously, two equations have been included from each stereo images pair, giving a total of four equations in four homogeneous unknowns.

The solution of the above homogeneous equation can be obtained using DLT algorithm. Since the value of A is known, a non-zero solution for M is found using SVD method witch satisfies the equation $AM = 0$.

2.4. Visual Servoing

Visual servoing is a largely used technique which is able to control on-line robots by using the information provided by one or many cameras. Two typical tasks are usually performed using visual serving, positioning and tracking. The former aims at aligning the robot or the gripper with the target object, while the latter aims at keeping a constant relationship between the robot and the moving target object. In both cases, image information is used to measure the error between the current location of the robot and its desired location.

The desired location is defined by an image (called desired image) perceived in such configuration. Through the matching the visual features (such as points, lines and regions) extracted from the desired and initial images, the initial location is obtained according to the desired location. The robot movement can be obtained on-line through the estimation of correspondences between features extracted from images taken sequentially from different positions.

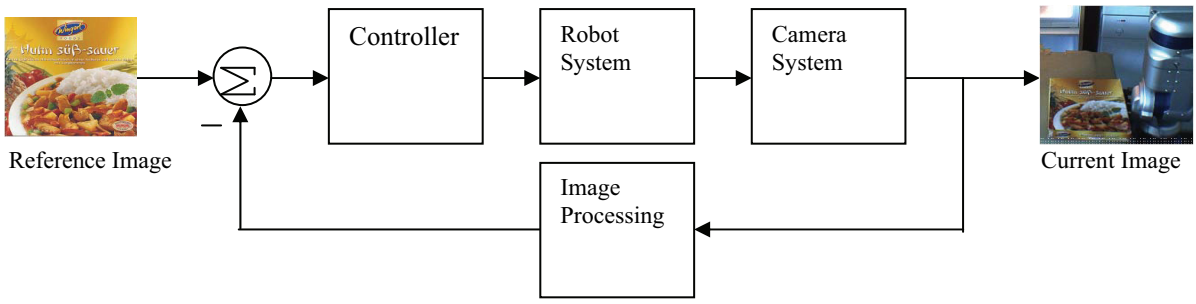


Figure 2.7: Visual servo control system

The basic concept of visual servoing is therefore based on the understanding of the scene geometry by the camera. The scene geometry is used to explain the relation between robot motion in the world and related image motion. In order to describe the geometry of the scene, three coordinate systems are used: camera, robot and world coordinate systems.

In general, visual servoing systems can be classified into two categories: position-based (IBVS) and image-based visual vervoing (PBVS).

In a position-based visual servoing, the system input is computed in the three-dimensional Cartesian space [13]. The pose of the target object with respect to the camera is estimated from image features corresponding to the perspective projection of the target object in the image. The pose estimation methods [14] are usually based on the knowledge of a perfect geometric model of the object and necessitate a calibrated camera to obtain unbiased results.

On the other hand, image-based visual servoing use optical flow along with Jacobian-based control to control the camera, in this case, the input is computed in the image plane [15].

Recently, a new approach has been proposed in [13] that exploit the combination of the two above methods to estimate the camera transformation between the desired and the current pose. They combine the traditional Jacobian-based control with other techniques to form the class of hybrid visual servoing (HVS). These methods yield a decoupled, optimal camera trajectory and possess a large singularity-free task space.

2.4.1.Position-based Visual Servoing

In PBVS, the task function is defined in terms of the pose transformation between the current and the desired position, which can be expressed as the transformation ${}^{c}T_{d}$.

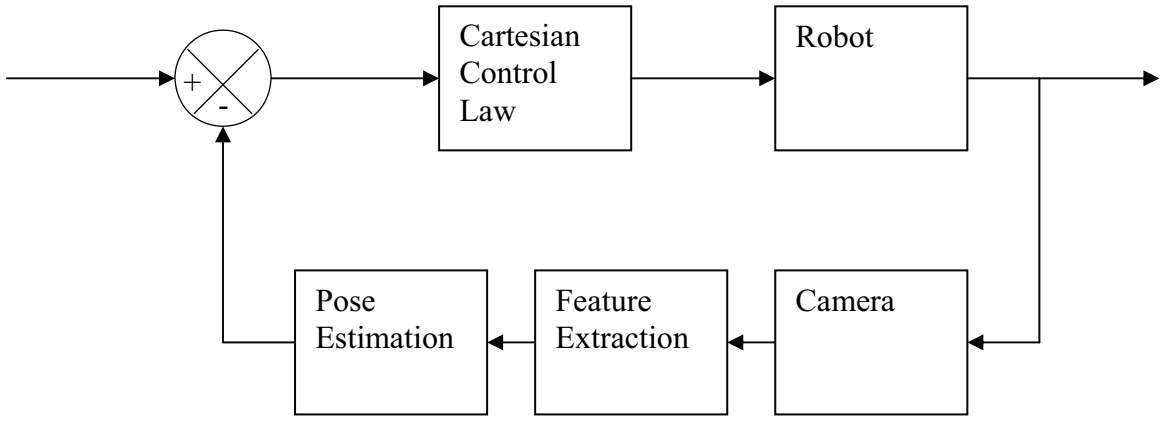


Figure 2.8: Postion-based visual servoing system.

The input image is usually used to estimate the camera to object transformation ${}^{c}T_{O}$ which can be composed with the object to desired pose transformation ${}^{O}T_{d}$ to find the transformation from the current to the desired pose. By decomposing the transformation matrices into translation and rotation, this can be expressed as:

$$\begin{aligned} {}^{c}T_{d} = {}^{c}T_{O}\,{}^{O}T_{d} &= \begin{bmatrix} {}^{c}R_{O} & {}^{c}t_{O} \\ O & 1 \end{bmatrix} \begin{bmatrix} {}^{O}R_{d} & {}^{O}t_{d} \\ O & 1 \end{bmatrix} \\ &= \begin{bmatrix} {}^{c}R_{O}\,{}^{O}R_{d} & {}^{c}R_{O}\,{}^{O}t_{d} + {}^{c}t_{O} \\ O & 1 \end{bmatrix} = \begin{bmatrix} {}^{c}R_{d} & {}^{c}t_{d} \\ O & 1 \end{bmatrix} \end{aligned} \tag{2.25}$$

The task function for position is then the vector ${}^{c}t_{d}$.

For orientation, the rotation matrix can be decomposed into axis of rotation r and rotation angle θ, which can be multiplied to get the desired rotational movement.

The rotation angle and rotation axis can be calculated from the elements of the rotation matrix R.

If the elements of the rotation matrix are expressed as:

$$R = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & {}_{33} \end{bmatrix} \tag{2.26}$$

The rotation angle and the direction of rotation axis are given as:

$$\begin{aligned} \theta &= \arccos\left(\frac{a_{11} + a_{22} + a_{33} - 1}{2}\right) \\ r &= \left[(a_{32} - a_{23}) \quad (a_{13} - a_{31}) \quad (a_{21} - a_{12})\right]^{T} \end{aligned} \tag{2.27}$$

In his kind of control, an error between the current and the desired position of the robot is calculated and used by the low level controller to generate the control commands to move the robot to the desired position.

$$e = P - P_d = \begin{bmatrix} t_d & r_d\theta_d \end{bmatrix}^T - \begin{bmatrix} t & r\theta \end{bmatrix}^T \tag{2.28}$$

Thus, the position-based controller can be written:

$$u = -\lambda\left(P - P_d\right) \tag{2.29}$$

The main advantage of this approach is that it directly controls the camera trajectory in Cartesian space. The central disadvantage of PBVS is that the pose estimation is usually based on the knowledge of a perfect geometric model of the object and necessitates a calibrated camera to obtain unbiased results. Therefore, if the camera is coarse calibrated, or if errors exist in the 3D model of the target object, the current and desired camera poses will not be accurately estimated which thus leads to servoing failure.

2.4.2.Image-based Visual Servoing

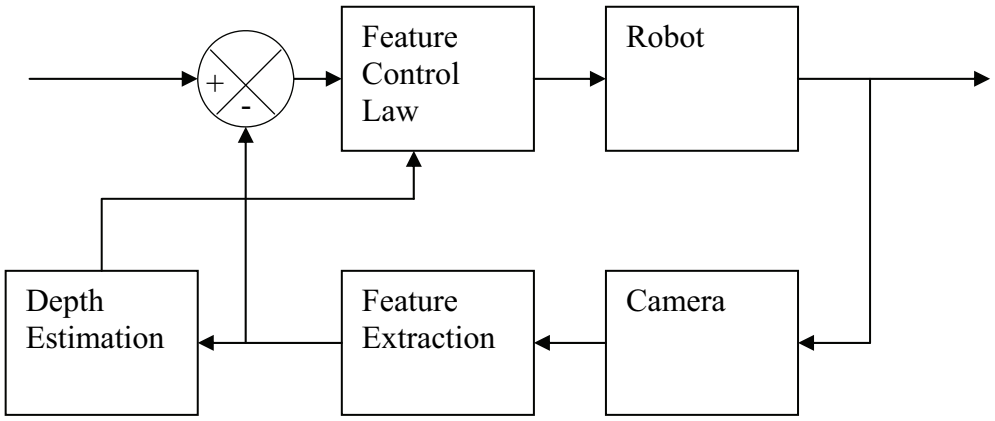


Figure 2.9: Image-based visual servoing system.

IBVS involves the estimation of the robot's velocity screw, so as to move the image plane features $m = \begin{bmatrix} m_0 & m_1 & .... & m_{n-1} \end{bmatrix}^T$ to a set of desired locations $m^* = \begin{bmatrix} m_0^* & m_1^* & .... & m_{n-1}^* \end{bmatrix}^T$ which represents the desired robot position. The error function is defined as a function of distance between these measurements $e = \begin{bmatrix} m_0 - m_0^* & m_1 - m_1^* & .... & m_{n-1} - m_{n-1}^* \end{bmatrix}^T$. This error function is updated in each frame and used together with the image Jacobian to estimate the control input to the robot.

Assuming that a point M_i on a target object with coordinates $\begin{bmatrix} x_i, y_i, z_i \end{bmatrix}^T$ measured in the camera coordinate system, is imaged at the point $m_i\left(u_i, v_i\right)$ in the image plane.

Using a classical perspective projection model, the relationship between each image point and its corresponding 3D world point is given by:

$$\begin{aligned} u_i &= \alpha_u \frac{x_i}{z_i} + u_0 \\ v_i &= \alpha_v \frac{y_i}{z_i} + v_0 \end{aligned} \tag{2.30}$$

where a_u, a_v, u_0 and v_0 are the intrinsic camera parameters. The equations (2.30) can be written as:

$$m_i = f(M_i) \tag{2.31}$$

When the time derivative of this equation is taken we obtain the relationship between the image point velocity and a 3D velocity screw:

$$\begin{aligned} \frac{\partial m_i}{\partial t} &= \frac{\partial f(M_i)}{\partial M_i}\frac{\partial M_i}{\partial t} \\ \dot{m}_i &= J(M_i)\dot{M}_i \end{aligned} \tag{2.32}$$

where $J(M_i)$ is the image Jacobian matrix given by:

$$J(M_i) = \frac{\partial f(M_i)}{\partial M_i} = \begin{bmatrix} -\frac{f}{z_i} & 0 & \frac{u_i}{z_i} & \frac{u_i v_i}{f} & \frac{-1-u_i^2}{f} & v_i \\ 0 & -\frac{f}{z_i} & \frac{v_i}{z_i} & \frac{1+u_i^2}{f} & \frac{-u_i v_i}{f} & -u_i \end{bmatrix} \tag{2.33}$$

where f is the focal length of the camera.

The image Jacobian represents the differential relationship between the scene frame and the camera frame (where either the scene or the camera frame is usually attached to the robot).

The image point velocity and the 3D screw velocity are given by:

$$\begin{aligned} \dot{m}_i &= \frac{\partial m_i}{\partial t} = \begin{bmatrix} u_i - u_i^* & v_i - v_i^* \end{bmatrix} \\ \dot{M}_i &= \frac{\partial M_i}{\partial t} = \begin{bmatrix} T_x & T_y & T_z & \omega_\alpha & \omega_\beta & \omega_\gamma \end{bmatrix} \end{aligned} \tag{2.34}$$

The image Jacobian matrix relates the motion of 2D points in the image plane (which is the effect) to the motion of the corresponding 3D points in the Cartesian space (witch is the cause).

When considering n 3D points together with their projections on the image plane, the Jacobian matrix J for the complete set of features is:

$$J = \begin{bmatrix} J(M_0) & J(M_1) & .... & J(M_n) \end{bmatrix}^T \tag{2.35}$$

In IBVS systems, the control error function is defined directly in 2D image plane. If image positions of point features are used as measurements, the error function is defined simply as a difference between the current and the desired feature positions as follows:

$$e_i = m_i - m_i^* \tag{2.36}$$

The most common approach to generate the control signal for the robots is the use of a simple proportional control [18] for an optimal control approach:

The control law can be obtained from equations (2.32) and (2.33) for at least three corresponding features:

$$u = K\dot{M} = K\left(J^T J\right)^{-1} J^T \dot{m} = K\left(J^T J\right)^{-1} J^T e \tag{2.37}$$

where K is a constant gain matrix.

In general, image-based visual servoing is known to be robust not only with respect to camera but also to robot calibration errors [19]. However, its convergence is theoretically ensured only in a region around the desired position.

2.4.3.Hybrid Visual Servoing

Malis et al. [16] proposed a hybrid control scheme (called 2,5D visual servoing). It combines the classical position-based and image-based approaches in order to overcome their respective drawbacks: contrarily to the position based visual servoing, it does not need any geometric 3D model of the object.

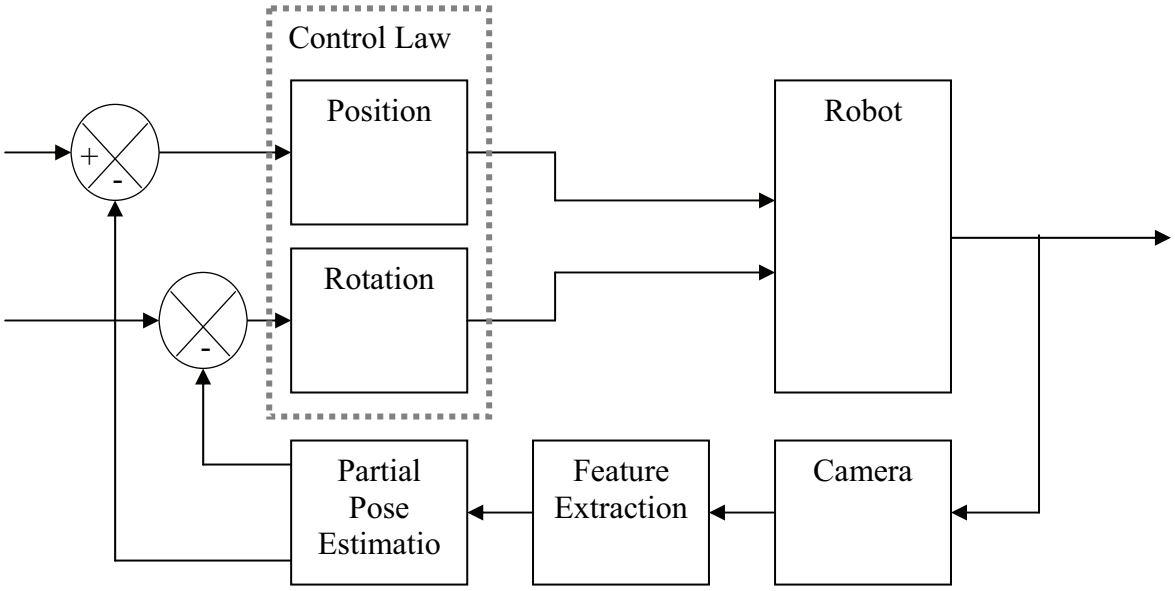


Figure 2.10: Hybrid visual servoing system.

In contrast to the image-based visual servoing, it guarantees the convergence of the control law to zero error in the whole task space and does not need for depth estimation when calculating the image Jacobian. This control is based on the estimation of the partial camera transformation from the current to the desired camera poses.

In each iteration, the rotation and the scaled translation of the camera between the current and the desired views of the object are estimated from the homography matrix. Visual features extracted from the partial transformation are used to design a decoupled control law.

The feature point velocity vector is augmented with depth and rotation information

$$\dot{\tilde{m}} = \begin{bmatrix} u - u^* & v - v^* & \log(\rho) & \theta \cdot r \end{bmatrix}^T \tag{2.38}$$

where ρ ist he ratio $\frac{z}{z^*}$ and θ and r are the angle and rotation axis of the rotation matrix extracted from the homography matrix.

Furthermore, ρ can be directly calculated from the homography matrix as:

$$\rho = |H| \frac{\begin{bmatrix} m^{*T} & 1 \end{bmatrix} n^*}{\begin{bmatrix} m^T & 1 \end{bmatrix} n} \tag{2.39}$$

While the rotation angle and the direction of rotation axis of the rotation matrix are computed from rotation matrix according to equation (2.27)

Malis et al [16] define the motion control law as:

$$\dot{M} = -K\tilde{J}^{-1}(r)\dot{m} \tag{2.40}$$

With

$$\tilde{J}^{-1} = \begin{bmatrix} \hat{d}^* \rho J_t^{-1} & -\hat{d}^* \rho J_t^{-1} J_r \\ 0 & I_3 \end{bmatrix} \tag{2.41}$$

where J_t and J_r are the translational and rotational portions of the image Jacobian matrix, composed of the first three and last three columns of the Jacobian respectively, and $\hat{d}^*$ is an estimate of the distance between the focal point and the feature point plane.

3. Image Matching

In order to measure the similarity between two images, the visual content of each image has to be transformed into quantitative characteristics that can be measured and compared with relatively little ambiguity. These quantitative characteristics are usually called image features and the process of comparing image features is also referred to as the image matching which tries to find corresponding features in two or more images. Image matching is a necessary step for many computer vision applications such as image registration, camera calibration, 3D reconstruction, visual serviong, and robot navigation.

In general, Image matching techniques can be classified into two categories: intensity-based and feature-based image matching.

Intensity-based methods compare intensity patterns in images via correlation metrics, while feature-based methods find correspondences between image features such as corners, edges, and blobs.

The Intensity-based methods are usually easy to implement but they can only be applied to matching the images with similar viewing conditions. These conditions are hard to satisfy in practice, especially in robot vision applications where images come with many shapes and appearances. In addition, these methods are not robust to deformation, occlusion and background clutter.

Feature-based methods are based on the establishment of the correspondences between a numbers of points in images. Therefore they are more robust to both clutter and occlusion. The feature-based matching approaches typically involve the following steps:

- Feature Detection
- Feature Description.
- Feature Matching.

3.1. Feature Detection

Feature detection refers to process that looks for positions in a given image where a particular feature of a given type can be located.

Visual feature is defined to be the description of an image region which contains significant structural information, such as edges, corners, and other patterns. In order to detect interest regions of an image, a saliency measure is defined and looked for its local Extrema across the image pixels and across different sizes of the region. The idea of checking different image sizes is to be able to detect the same region even if the region is present at different scales in different images. This leads to so called scale invariant detection.

The selection of the saliency measure Extrema is to make detection process more repeatability. The feature repeatability is defined as the probability that the same feature will be detected in two or more different images of the same scene, even under different capturing conations.

In literature, there are many types of features that can be extracted from a digital image such as edges, corners, and blobs.

Edges mark the boundaries between different areas in the image, for example areas of different brightness levels, or texture statistics. Corners are found at the peaks in the auto-correlation function or points where edges intersect. Blobs are found in the stable centers of uniform regions.

Based on feature type, feature detectors can be divided into tree groups: Edge, corner and blob detectors.

3.1.1.Edge Detectors

Edges are located where intensity values in the two-dimensional image function undergo a sharp change from one state to another, such as from a white square to a black background.

These points are the local maxima of the gradient of the image. Canny edge detection [20] is an efficient process that produces a binary edge image in which every point is labeled as an edge or otherwise.

Edge detection is a problem of fundamental importance in image analysis. In typical images, edges characterize object boundaries and are therefore useful for segmentation, registration, and object recognition in a scene.

An edge is a boundary between two image regions represented as a jump in intensity. In general, the cross section of an edge can be of arbitrary shape (usually ramp). In practice, edges are usually defined as sets of points in the image which have a strong gradient magnitude.

For a continuous image $I(x,y)$, where x and y are the row and column coordinates respectively, we typically consider the two directional derivatives g_x and g_y.

Of particular interest in edge detection are two functions that can be expressed in terms of these directional derivatives: the gradient magnitude and the gradient orientation.

The gradient magnitude is defined as:

$$m(x,y) = \sqrt{(g_x)^2 + (g_y)^2} \tag{3.1}$$

And the gradient orientation is given by:

$$\theta(x,y) = \tan^{-1}\left(g_x / g_y\right) \tag{3.2}$$

Where $g_x = \frac{\partial I(x,y)}{\partial x}$ and $g_y = \frac{\partial I(x,y)}{\partial y}$

Local maxima of the gradient magnitude justify edges in $I(x,y)$ which is the basic idea of the first order derivative- based edge detectors. An odd symmetric filter will approximate a first derivative, and peaks in the convolution output will correspond to edges in the image. Often, the first derivative of the digital image is expressed as a convolution of the digital image with a convolution mask which is also always called edge operator, and then the resulting outputs are processed to give a gradient map.

The magnitude of the gradient map is calculated and serves as input of a non-maxima suppression process. Finally the resulting map of local maxima is thresholded to produce the edge map.

While the first derivative achieves a maximum, the second derivative is zero. For this reason, an alternative edge-detection strategy is to locate zeros of the second derivatives of $I(x,y)$. The differential operator used in these so-called zero-crossing edge detectors is the Laplacian:

$$\nabla I = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2} = g_{xx} + g_{yy} \tag{3.3}$$

The zero crossing detectors such as Marr- Hildreth and Laplacian of Gaussian (LoG) edge detectors [21] look for places in the Laplacian of an image where the value of the Laplacian passes through zero *i.e.* points where the Laplacian changes sign. Such points often occur at edges in images *i.e.* points where the intensity of the image changes rapidly.

The starting point for the zero crossing detector is an image which has been filtered using the LoG filter.

3.1.2.Corner Detectors

Generally, a corner is defined as the intersection of two edges, but in images, corners referred to as pixels that correspond to maxima in the autocorrelation function

A number of algorithms for corner detection have been reported in recent years. They can be divided into two groups. Algorithms in the first group involve extracting edges and then finding the points having maxima curvature or searching for points where edge segments intersect. The second group consists of algorithms that search for corners directly from the grey-level image, so that corner can also be defined as a point for which there are two dominant and different edge directions in a local neighborhood of the point.

The quality of a corner detector is often judged based on its ability to detect the same corner in multiple images, which are similar but not identical, for example having different lighting, translation, rotation and other transforms.

One of the earliest interest point detection algorithms is the Moravec corner detector [22]. In the algorithm, a slide window around a pixel is moved in four directions and the gray-level change in four directions are computed $E(x,y)$. $E(x,y)$ is very small of each direction if the pixel is on a smooth region. At edges, $E(x,y)$ changes only in one direction. For a corner point, $E(x,y)$ changes greatly in all directions Therefore, the corner strength at a pixel is defined as the smallest sum of squared differences between the patch and its neighboring patches.

$$E(x,y) = \sum_u \sum_v \left(I(u,v) - I(u+x, v+y)\right)^2 \tag{3.4}$$

The problem with Moravec corner detector is that the patches only in horizontal, vertical, and diagonal directions are considered; that is the algorithm is not isotropic.

An alternative approach for corner detection used frequently is based on a method proposed by Harris and Stephens [23], which in turn is an improvement of a method by Moravec. The Harris corner detector is based on the local auto-correlation function of a signal; where the local auto-correlation function measures the local changes of the signal with patches shifted by a small amount in different directions. The Harris corner detector also computes a cornerness value, $C(x,y)$, for each pixel in an image. A pixel is declared as corner if the value of *C* is below a certain threshold. where $C(x,y)$ is calculated as:

$$C(x,y) = \sum_u \sum_v w(u,v) \cdot \left(I(u,v) - I(u+\Delta x, v+\Delta y)\right)^2 \tag{3.5}$$

$I(u+\Delta x, v+\Delta y)$ can be approximated by a Taylor expansion. Let I_x and I_y be the partial derivatives of $I(x,y)$, such that

$$I(u+\Delta x, v+\Delta y) \approx I(u,v) + I_x(u,v) \cdot \Delta x + I_y(u,v) \cdot \Delta y \tag{3.6}$$

By substituting equation (3.5) into equation (3.6), we obtain the following approximated cornerness value.

$$C(x,y) \approx \sum_u \sum_v w(u,v) \cdot \left(I_x(u,v) \cdot \Delta x + I_y(u,v) \cdot \Delta y\right)^2 \tag{3.7}$$

Through rewriting equation (3.7) in matrix form we get:

$$C(x,y) \approx \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \cdot A \cdot \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3.8}$$

where A is the Harris matrix.

$$A = \sum_u \sum_v w(u,v) \cdot \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix} \tag{3.9}$$

In the equation (3.9), the angle brackets denote averaging (i.e. summation over (*u*,*v*)). If a circular window (or circularly weighted window, such as a Gaussian) is used, then the response will be isotropic.

A corner is characterized by a large variation of $C(x,y)$ in all directions of the vector $\begin{bmatrix} \Delta x & \Delta y \end{bmatrix}$. By analyzing the eigenvalues of A, this characterization can be expressed in the following way:

The matrix A should have two large eigenvalues for a corner point. Based on the magnitudes of the eigenvalues, the following inferences can be made:

Assuming that λ_1 and λ_2 are the eigenvalues of the matrix A. There are three cases to be considered:

1. If both λ_1 and λ_2 are small, so that the local auto-correlation function $C(x, y)$ changes slightly in any direction, the windowed image region is of approximately constant intensity; this indicates a flat region.
2. If one of the eigenvalues is big and the other is small, so the local auto-correlation function is ridge shaped, then only local shifts in one direction (along the ridge) cause weak change in $C(x, y)$ and significant change in the orthogonal direction; this indicates an edge.
3. If both eigenvalues are big, so the local auto-correlation function is sharply peaked, then shifts in any direction cause significant change; this indicates a corner.

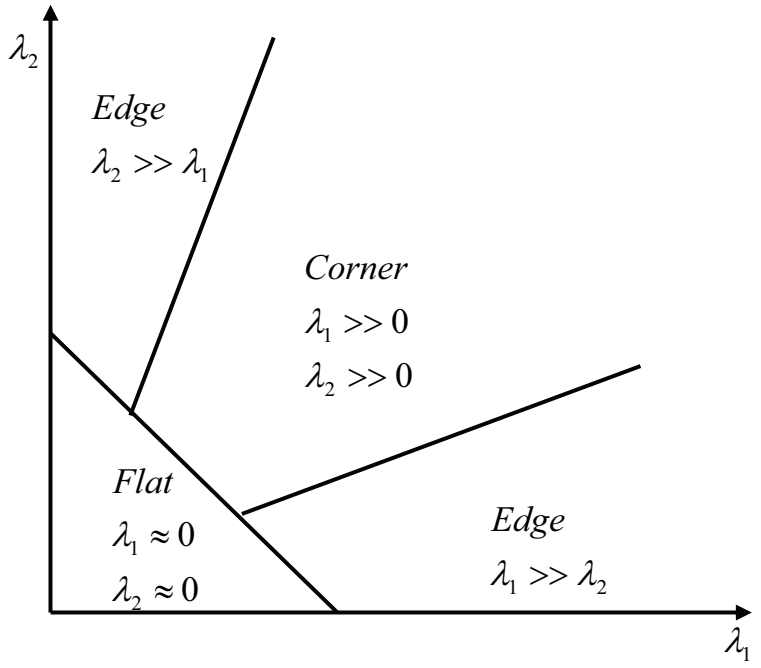


Figure 3.1: The Harris and Stephens corner detector [23].

Harris and Stephens note that exact computation of the eigenvalues is computationally expensive, since it requires the computation of a square root, and instead they suggest the value M_c given by the equation (3.10):

$$M_c = \lambda_1 \cdot \lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 = \det(A) - \kappa \cdot (trac(A))^2 \qquad (3.10)$$

where κ is a tunable sensitivity parameter

Therefore, the algorithm does not have to actually compute the eigenvalue decomposition of the matrix *A* and instead it is sufficient to evaluate the determinant and trace of *A* to detect corners. The value of κ has to be determined empirically, but in the literature values in the range 0.04 - 0.15 are commonly used.

3.1.3. Blob Detectors

In the computer vision community, a blob refers to uniform region in the image that is either brighter or darker than its surrounding.

There are two main classes of blob detectors, watershed-based blob and differential blob detectors.

The watershed-based detector developed by Lindeberg [24] is based on local extremum in the intensity. Detecting watershed-based blobs in a one-dimensional function is trivial. In this case it suffices to start from each local maximum point and initiate search procedures in each one of the two possible directions. Every search procedure continues until it finds a local minimum point. As soon as a minimum point has been found the search procedure is stopped and the grey-level value is registered. The base-level of the blob is then given by the maximum value of these two registered grey-levels. From this information the grey-level blob is given by those pixels that can be reached from the local maximum point without descending below the base-level.

The two-dimensional case is more elaborate, since the search then may be performed in a variety of directions. In [24] Lindeberg proposed a methodology that avoids the search problem by performing a global blob detection based on a pre-sorting of the grey-levels. In order to extract both dark blobs and bright blobs, watersheds are typically extracted from the gradient image. In practice, the bottleneck of the watershed-based detector is the inherent noise sensitiveness which leads typically to over segmented results. To overcome this, it would be helpful to incorporate information about shape and size of the desired blobs into the process of watershed detection, which is hardly feasible.

The differential detectors are based on derivative expressions such as Laplacian of Gaussian (LoG), Deference of Gaussian (DoG) and Determinant of Hessian (DoH). The Laplacian of Gaussian filter (LoG) is a combination of a Laplacian and Gaussian filter. This filter first applies a Gaussian blur, and then applies the Laplacian filter. The first stage of the filter uses a Gaussian kernel to blur the image in order to make the Laplacian filter less sensitive to noise.

$$g(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \tag{3.11}$$

Then, the Laplacian operator is computed, which usually results in strong positive responses for dark blobs of extent $\sqrt{\sigma}$ and strong negative responses for bright blobs of similar size.

A main problem when applying this operator at a single scale, however, is that the operator response is strongly dependent on the relationship between the size of the blob structures in the image domain and the size of the Gaussian kernel used for pre-smoothing. In order to automatically detect blobs of different unknown size, the scale-normalized LoG is applied at the scale space representation.

$$\sigma\nabla L=\sigma\left(L_{xx}-L_{yy}\right) \tag{3.12}$$

where $L=I(x,y)*g(x,y,\sigma)$ is Gaussian blurred image.

The scale space representation is constructed by iteratively convolving the high resolution image with Gaussian based kernels of different size.

Lindeberg [25] proposed a method for detecting blobs like features in a scale-space representation. In order to detect blobs and compute their scale, a search for Extrema of scale-normalized Laplacian of Gaussian is performed.

The DoG operator can be used as an approximation to the LoG to find very stable interest points in the center of stable blobs. In a similar way as for the LoG, blobs can be detected from scale-space Extrema of DoG.

Anther blob detector is based on the scale-normalized determinant of the Hessian (DoH) [46] as explained by the equation below.

$$\sigma \det(H) = \sigma\left(L_{xx} L_{yy} - L_{xy}^2\right) \tag{3.13}$$

where $H = \begin{pmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{pmatrix}$ is the Hessian matrix.

In terms of scale selection, blobs defined from scale-space Extrema of the scale-normalized DoH also have slightly better scale selection properties under non-Euclidean affine transformations than the other two popular blob detectors, LoG and DoG

3.2. Feature Description

Once the interesting locations in the image have been detected, the task remains is to describe these locations quantitatively. The obtained quantitative descriptions are called feature descriptors. The descriptors are usually histograms of image measurements derived from interest local regions. In order to be effective, the descriptor has to be distinctive and at the same time robust to noise and to changes in both viewpoint and photometric imaging conditions, hence a good trade-off between robustness and distinctiveness should be achieved while designing the description procedure. It is in essence a targeted data reduction which gives particular information about an area in a compact form.

In computer vision, several visual descriptors have been proposed for representing the visual content of images. These descriptors can be generally classified depending on the elementary characteristics of interest into tree major groups: color, texture and shape descriptors.

3.2.1. Color Descriptors

Color is one of the most widely used visual features in image description, similarity, and retrieval tasks. Color features are invariant to rotation, translation, and scaling, but not invariant to illumination changes. An important issue for color feature description is the choice of the color space. The color space is a multi-dimensional coordinate system, and each dimension represents a specific color component such as RGB, HSV.

In the last two decades, many color descriptors for images and image regions have been proposed [26] such as Color Histogram (CH), Color Moments (CM) and Color Coherence Vector (CCV). The CH is the basic color descriptor, which describes the color distribution of the image or the image region. CH is computed by dividing color space into n discrete representative colors, and counting the number of pixels having the same color.

However, the main disadvantage of the color histogram is that it is not robust to significant appearance changes because it does not include any spatial relationships among colors.

The CCV [27] is an extension of color histograms, in that each pixel is classified as coherent or non-coherent based on whether the pixel and its neighbors have similar colors. Color Correlogram is proposed to characterize how the spatial correlation of pairs of colors is changing with the distance [28]. Color Correlogram provides much better performance than CH and the CCV.

3.2.2.Texture Descriptors

In general, Texture refers to the visual properties of surface such as smoothness or roughness. Texture can be seen almost anywhere. For example, trees, grass, sky, roads and buildings appear as different types of texture. Describing textures in images by appropriate texture descriptors provides powerful means for similarity matching. A wide variety of texture descriptors have been recently proposed. Texture descriptors can be classified into two categories: homogeneous and non-homogeneous texture descriptors. The homogeneous texture descriptor (HTD) provides a quantitative characterization of homogeneous texture regions that has homogenous properties. It is based on computing the local spatial-frequency statistics of the texture using the Gabor transform [30].

Because non-homogeneous textures have statistical and structural properties, non homogeneous texture descriptors can be categorized into statistical and structural texture descriptors [29]. In structural approaches, statistical distributions of texture primitive such as edges are used to describe texture patterns. As an example for structural texture descriptor is edge histogram descriptor (EHD) [31]. This descriptor captures spatial distribution of edges in the image. In order to construct EHD, edges are classified in five edge categories: vertical, horizontal, 45 , 135 , and non-directional edge. Hence EHD is expressed as a 5-bin histogram. Therefore EHD is scale and rotation invariant.

For statistical approaches, statistical distributions of individual pixel values such as gray level histogram and co-occurrence matrix are computed to discriminate different textures. The co-occurrence matrix is a two dimensional histogram of the distribution of the co-occurrence between two grey level values at a given distance [32].

Texture descriptors, are usually computed over the entire image and result in one feature vector per image, and therefore are not robust to occlusion and clutters. In recent years, some very discriminative local texture descriptors have been proposed such as Scale Invariant Feature Transform (SIFT) [4], Speeded Up Robust Feature (SURF)[6] and Gradient Location and Orientation Histogram (GLOH)[8]. Local descriptors are computed at multiple points in the image and describe image patches around these points, and thus are more robust to clutter and occlusion.

3.2.3. Shape Descriptors

In many computer vision applications, the shape representations provide powerful visual features for similarity matching. In image matching, it is usually required that the shape descriptor is invariant to scaling, rotation, and translation. There are generally two types of shape representations, boundary-based and region-based. Boundary-based methods such as

chain codes [33] and Fourier descriptors [34] need only contour pixels. Boundary-based shape descriptors may not be suitable to describe regions that have complex shapes.

Region-based methods, however, rely not only on the contour pixels but also on all pixels enclosed within the region of interest, hence they are more suitable for describing regions of complex shapes.

A region can be described by considering scalar measures based on its geometric properties. The simplest property is given by its area. Area is rotation invariant, but changes with changes in scale. Another simple property is defined by the perimeter of the region. Based on the area and perimeter it is possible to characterize the compactness of region, which is defined by the ratio of perimeter to area. The most popular shape descriptors are based on moments, which describe the shape and the intensity distribution in images.

A general definition of moment functions m_{pq} of order $(p+q)$, of an image intensity function $I(x,y)$ can be given as follows:

$$m_{pq} = \sum_x \sum_y x^p y^q I(x,y) \tag{3.14}$$

Geometric moments are invariant to rotation and scale changes, but not invariant to translation since the output would depend on the relative pixel positions within the image. To achieve translation invariant, central moments are derived from geometric moments by shifting the image so that the image centoid $(\bar{x},\bar{y})$ coincides with the origin of the image coordinate system:

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x,y) \tag{3.15}$$

Where $\bar{x} = m_{10}/m_{00}$ and $\bar{y} = m_{01}/m_{00}$

In [35] Hu used central moments to derive seven invariant moments that were then widely used in pattern recognition:

$$\begin{aligned}
h_1 &= (\mu_{20}+\mu_{02})/\mu_{00} \\
h_2 &= \left[(\mu_{20}-\mu_{02})^2 + (2\mu_{11})^2\right]/(\mu_{00})^2 \\
h_3 &= \left[(\mu_{30}-\mu_{12})^2 + (\mu_{21}-\mu_{03})^2\right]/(\mu_{00})^2 \\
h_4 &= \left[(\mu_{30}+\mu_{12})^2 + (\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^2 \\
h_5 &= (\mu_{30}-3\mu_{12})(\mu_{30}-\mu_{12})\left[(\mu_{30}-\mu_{12})^2 - 3(\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^4 + \\
&\quad (3\mu_{21}-\mu_{03})(\mu_{21}+\mu_{03})\left[3(\mu_{30}+\mu_{12})^2 - (\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^4 \\
h_6 &= (\mu_{20}-\mu_{02})\left[(\mu_{30}+\mu_{12})^2 - (\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^3 + \\
&\quad 4\mu_{11}(\mu_{30}+\mu_{12})(\mu_{21}+\mu_{03})/(\mu_{00})^3 \\
h_7 &= (3\mu_{21}-\mu_{03})(\mu_{30}+\mu_{12})\left[(\mu_{30}+\mu_{12})^2 - 3(\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^4 + \\
&\quad (3\mu_{30}-3\mu_{12})(\mu_{21}+\mu_{03})\left[3(\mu_{30}+\mu_{12})^2 - (\mu_{21}+\mu_{03})^2\right]/(\mu_{00})^4
\end{aligned} \tag{3.16}$$

Hu moments are calculated from central geometric moments of order up to the 3rd. Their main drawback refers to the large values of geometric moments, which lead to numerical instabilities and noise sensitivity. Since the basic function $x^p y^q$ for geometric moments is not orthogonal. Thus, Hu moments are not orthogonal and as a consequence, the calculated moments will have redundant information and cause less accuracy of representing images.

Teague proposed Zernike moments based on the basis set of orthogonal Zernike polynomials [36]. The orthogonal property of Zernike polynomial avoids any redundancy between moments of different orders. Zernike polynomials provided very useful moment kernels. They present native rotational invariance and are far more robust to noise. Scale and translation invariance can be achieved using moment normalization. For 2D images, the Zernike moment of order p with repetition q is defined as follows:

$$Z_{pq} = \frac{p+1}{\pi} \iint_{x^2+y^2 \le 1} I(x,y) \cdot V_{pq}^{*}(r,\theta) dxdy \tag{3.17}$$

where $p \ge 0$, $p - |q| \ge 0$, $r = \sqrt{x^2 + y^2}$, $\theta = \tan^{-1}(y/x)$, $V_{pq}(r,\theta) = R_{pq}(r) e^{jq\theta}$, and $R_{pq}(r)$ is radial polynomial of order p with coefficients depending on both p and q [36].

3.3. Feature Matching

As a consequence of image feature detection and description, each image is abstracted as a set of local features. Feature descriptors are usually represented as histograms.

In order to match two image, it is needed a searching technique that compares each pair of features from each image based on a similarity measure (Euclidian, or Mahalanobis) of their respective descriptions and then makes a decision based on a matching strategy.

The feature matching procedure therefore consists of three parts: the similarity measure, the matching strategy and the searching technique.

3.3.1. Similarity Measures

If the feature is represented as a histogram, the similarity between two features can be evaluated using any distance measure suitable for histograms.

There are two main types of similarity measures: bin-by-bin and cross-bin measures [37].

Bin-by-bin techniques, like the Minikowski only compare corresponding histogram bins, without regarding information in nearby bins. The Minkowski distance of order p is defined by the following equation:

$$d_p(V^1, V^2) = \sqrt[p]{\sum_{i=1}^{N} \left| v_i^1 - v_i^2 \right|^p} \tag{3.18}$$

where V^1 and V^2 are feature descriptors from the first and the second image respectively.

$$V^1 = \begin{bmatrix} v_1^1 & v_2^1 & ..... & v_N^1 \end{bmatrix} \tag{3.19}$$

$$V^2 = \left[v_1^2 \quad v_2^2 \quad ..... \quad v_N^2 \right]$$

The Euclidean distance, which is a special case of Minkowski distance when $p = 2$, is the most common distance measures used in practice.

In contrast, cross-bin techniques take into account non-corresponding bins as well, and are thus more powerful. As an example for cross-bin measure is the quadratic form distance (QFD), which computes the minimal cost for flowing bin matter from one histogram to form the other. The QFD is defined as follows:

$$d\left(V^1, V^2\right) = \left(V^1 - V^2\right)^T A \left(V^1 - V^2\right) \tag{3.20}$$

where $A = \left\lfloor a_{ij} \right\rfloor$ is a bin-similarity matrix whose elements a_{ij} are given by:

$$a_{ij} = 1 - \frac{d_{ij}}{\max\left(d_{ij}\right)} \tag{3.21}$$

where $d_{ij} = \left| v_i^1 - v_j^2 \right|$ is the distance between two histograms bins.

If the bin-similarity matrix is positive-definitive, then the QFD becomes the L_2-norm between the linear transformations of V^1 and V^2.

A special case of QFD when the bin-similarity matrix is the inverse of the covariance matrix is the Mahalanobis distance. The Mahalanobis distance is adapted better than the Euclidian distance to describe similarities in multidimensional spaces when non-isotropic distributions are involved.

3.3.2. Matching Strategies

There are three common strategies to make a decision whether two features are correctly matched each other according to the matching measure: absolute threshold, thresholded nearest neighbor and nearest neighbor distance ratio.

Absolute Threshold:

Two features are considered as a correct match if the absolute distance between them is less than a pre-set threshold. Under this matching strategy, each feature from the first feature set may match to more than one feature from the second feature set.

Thresholded Nearest Neighbor (TNN):

Each feature from the first feature set are matched to its nearest neighbor feature from the second set if the absolute distance between them is less than a pre-set threshold. In this case, only some features form the first feature set may find corresponding features from the second feature set.

Nearest Neighbor Distance Ratio (NNDR):

For each feature from the first feature set, its distances to the nearest and the second nearest neighbor features of the second feature set are firstly computed. If the ratio between these distances is less than a pre-set threshold, then the feature and its nearest neighbor feature are considered as a match.

3.3.3. Searching Techniques

The simples search algorithm for nearest neighbor (NN) is the exhaustive search, where each feature in the first feature set is compared with all features in the second feature set. The main drawback of exhaustive search is its very high complexity. In order to overcome this problem, many methods have been proposed for approximate nearest neighbor (ANN) search. Generally ANN searching techniques can be classified into two groups: Hierarchical space partition-based and hash-based methods.

Hierarchical space partition-based methods

The first group involves all tree-based approaches such as k-d tree. The k-d tree was proposed by Bentley [38] and is likely the most widely used ANN method. The k-d tree is a binary search tree in which each node represents a partition of the k-dimensional space. The root node represents the entire space, and the child nodes represent sub-spaces which are part of their parent node's space. Every node has a key value associated with one of the k-dimensions. At each node, its space is divides into two parts, left subspace contains all features whose k^{th} component is less than the key value and the right sub-space contains all features whose k^{th} components is greater than the key value.

When the tree is searched, the corresponding component of query feature q is compared against the node key value, and the appropriate branch is followed. Once a leaf node is reached, the query feature is tested against all the features in the leaf node and the closest feature p is determined.

It may happen that the true nearest neighbor p lies in a different leaf node. This will occur when the distance between q and the boundary of its bin region is less than the distance between q and p.

Therefore, p is guaranteed to be the true nearest neighbor if the sphere centered at q with radius $\|q-p\|$ is completely contained within the bin region. This is known as the ball-within-bounds (BWB) test.

If the BWB test fails, then p may not be the true nearest neighbor, and it is necessary to backtrack up the tree and test points contained in alternate paths.

Another test which must be regarded when the tree is searched is called the bounds- overlap-ball (BOB) test. BOB test determines whether or not the sphere centered at q intersects with some region, which may therefore contain the true nearest neighbor. All points contained in all bin regions that pass the BOB test must be considered during backtracking. If a new nearest neighbor is encountered, then the sphere radius is adjusted downward, the BWB test is repeated, and the backtracking resumes if necessary.

There are many other methods based on hierarchical space partition for ANN searching such as R-trees [39] and B-trees [40].

However, all the above methods do not work well for high dimensional searching space, because the increase of the dimensionality of the searching lead to highly unbalanced trees due to most of the tree leaves are empty.

Hash-based methods

The second category consists of hash-based approaches which trade accuracy for efficiency, by returning approximate closest neighbors of a query point. The most popular hash-based method is locality sensitive hashing (LSH) [41]. The basic idea of LSH is to use a set of hash

functions that map similar features into the same hash bucket with a probability higher than non-similar features. At indexing time, all the features of the dataset are inserted in L hash tables corresponding to L randomly selected hash functions.

At query time, the query feature q is also mapped onto the L hash tables and the corresponding L hash buckets are selected as candidates to contain features similar to the query feature. A final step is then performed to filter the candidate features by computing their distance to the query feature.

More formally, let V be a dataset of N d-dimensional features in $\Re^d$ under the L_2 - norm. For any point v belong to $\Re^d$, the notation $\|v\|_2$ represents the L_2 - norm of the vector v.

Assuming that $G = \{g : \Re^d \to \aleph^k\}$ be a Group of hash functions such as:

$$g(v) = [h_1(v) \quad h_2(v) \quad .... \quad h_k(v)] \tag{3.22}$$

where the functions h_i belongs to a locality sensitive hashing function family $H = \{h : \Re^d \to \aleph\}$.

The function family H is called $(R, cR, p_1, p_2) - sensitive$ for L_2 - norm if for any $q, v \in \Re^d$

$$\begin{aligned} &p(h(q) = h(v)) \geq p_1 \quad when \quad \|q - v\|_2 \leq R \\ &p(h(q) = h(v)) \leq p_2 \quad when \quad \|q - v\|_2 \geq cR \end{aligned} \tag{3.23}$$

where $c > 1$ and $p_1 > p_2$.

Intuitively, that means that nearby features within distance R have a greater chance of being hashed to the same value than features that are far away (distance greater than cR).

For the L_2 - norm, the typically used LSH functions are defined as:

$$h(v) = \left\lfloor \frac{a \cdot v + b}{w} \right\rfloor \tag{3.24}$$

where $a \in \Re^d$ is a random vector with entries chosen independently from a Gaussian distribution and $b \in \Re$ a real number chosen uniformly from the range $[0, w]$.

For ANN searching tasks, the LSH indexing method works as follows:

1. L hash functions $[g_1 \quad g_2 \quad .... \quad g_L]$ from G are selected independently and uniformly at random, so that each hash function is the concatenation of k LSH functions randomly generated from H.

   $g_i = [h_1^i(v) \quad h_2^i(v) \quad .... \quad h_k^i(v)]$

2. Each one of the L hash functions is used to construct one hash table (resulting in L hash tables).
3. All points $v \in V$ are inserted in each of the L hash tables by computing the corresponding L hash values.

During the creation of the LSH hash tables, the algorithm stores each data point in the dataset into buckets $g_j(v)$, for all $j \in [1, L]$. Then, during the processing of a query q, the algorithm searches all buckets $(g_1(q) \quad g_2(q) \quad .... \quad g_L(q))$.

For each feature v found in a bucket, the algorithm computes the distance from q to v, and reports the features if and only if their distances to query feature are less than certain threshold

While this method is very efficient in terms of time, tuning such hash functions depends on the distance of the query point to its closest neighbor.

4. SIFT Algorithm

The Scale Invariant Feature Transform (SIFT) method, proposed by Lowe [4] is one of the most widely used methods for image matching which is useful for almost all computer vision tasks. The algorithm intends to detect similar feature points in each of the available images and then describe these points with a feature vector which is invariant to scale and rotation, and partially invariant to illumination and viewpoint changes. In addition to these properties, SIFT features are highly distinctive and relatively easy to extract and to match, but the extraction as well as the matching of these features involves a considerable computational cost. In order to use SIFT algorithm for matching purpose, SIFT features which correspond to different views of the same scene should have similar feature vectors.

The image matching methods that use SIFT features, consists of two parts, SIFT feature extraction and SIFT feature matching. Extraction involves finding and describing interest regions or points, while matching means finding of the correspondences among features in different images.

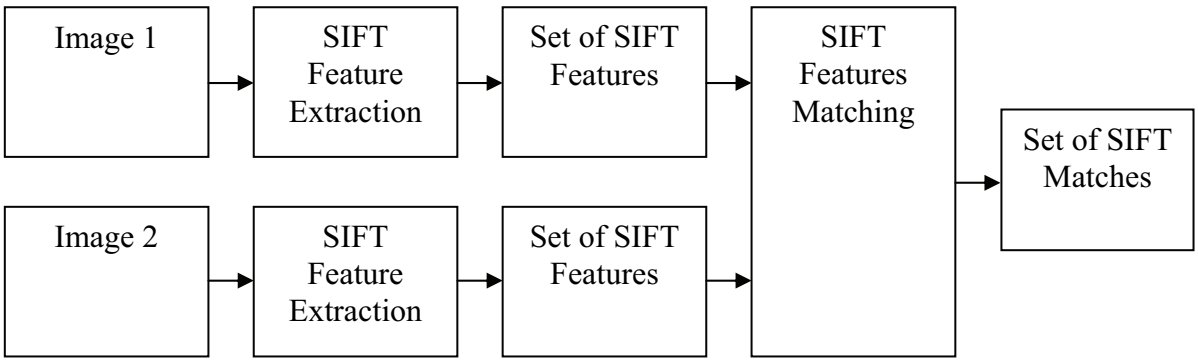


Figure 4.1: SIFT algorithm (SIFT feature extraction and matching).

4.1. SIFT Feature Extraction

SIFT algorithm extracts key-points invariant to scale and rotation using the Gaussian difference of the images in different scales to ensure invariance to scale. Rotation invariance is achieved by assigning one or more orientations to each key-point location based on local image gradient directions. The result of all this process is a 128 dimensional descriptor of gradients arranged together according to their orientation and location, which provides an efficient tool to describe an interest point, allowing an easy matching against a database of key-points. The extraction of SIFT features can be decomposed into four major stages:

1. Scale-space Extrema detection: The first stage searches over scale space using a Difference of Gaussian (DoG) function to identify potential interest points.
2. Key-point localization: The sub-pixel location and scale of each candidate point is determined and key-points are filtered by retaining only those that are robust to noise and illumination changes.
3. Orientation assignment: One or more orientations are assigned to each key-point based on local image gradient directions.
4. Key-point descriptor: A descriptor vector is generated for each key-point from local image gradient data at the key-point scale.

4.1.1. Scale-Space Extrema Detection

The locations of potential interest points in the image are determined by detecting the Extrema (Maxima and Minima) of DoG scale space.

In order to construct DoG scale space, it is needed firstly to build a Gaussian scale-space representation of the image. The GSS is built from the convolution of the input image $I(x,y)$ with a variable-scale Gaussian:

$$L(x,y,\sigma) = I(x,y) * G(x,y,\sigma) \tag{4.1}$$

where $*$ is the convolution operator in x and y directions and $G(x,y,\sigma)$ is the Gaussian kernel given by:

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2}\exp\left(\frac{-(x^2+y^2)}{2\sigma^2}\right) \tag{4.2}$$

As illustrated in Figure 4.2, the G-SS consists of a series of smoothed images at discrete values of σ over a number of octaves where the size of the image is down-sampled by two at each octave. Because of the recursive property of the Gaussian function, in each octave each image can be calculated from the previous one. Since $L(x,y,\sigma)$ are blurred with increasing σ, images of the next octave can be down-sampled as shown in Figure 4.2, without losing important information. This reduces the computational complexity significantly.

In SIFT method, the σ of the Gaussian scale space is quantized in logarithmic steps arranged in O octaves, where each octave is further subdivided in S scale levels. The value of σ at a given octave o and scale level s is given by:

$$\begin{aligned} \sigma(o,s) &= \sigma_0 \cdot 2^{\left(o+\frac{s}{S}\right)} \\ s &\in [0, S-1], o \in [0, O-1] \end{aligned} \tag{4.3}$$

where σ_0 is the base scale level.

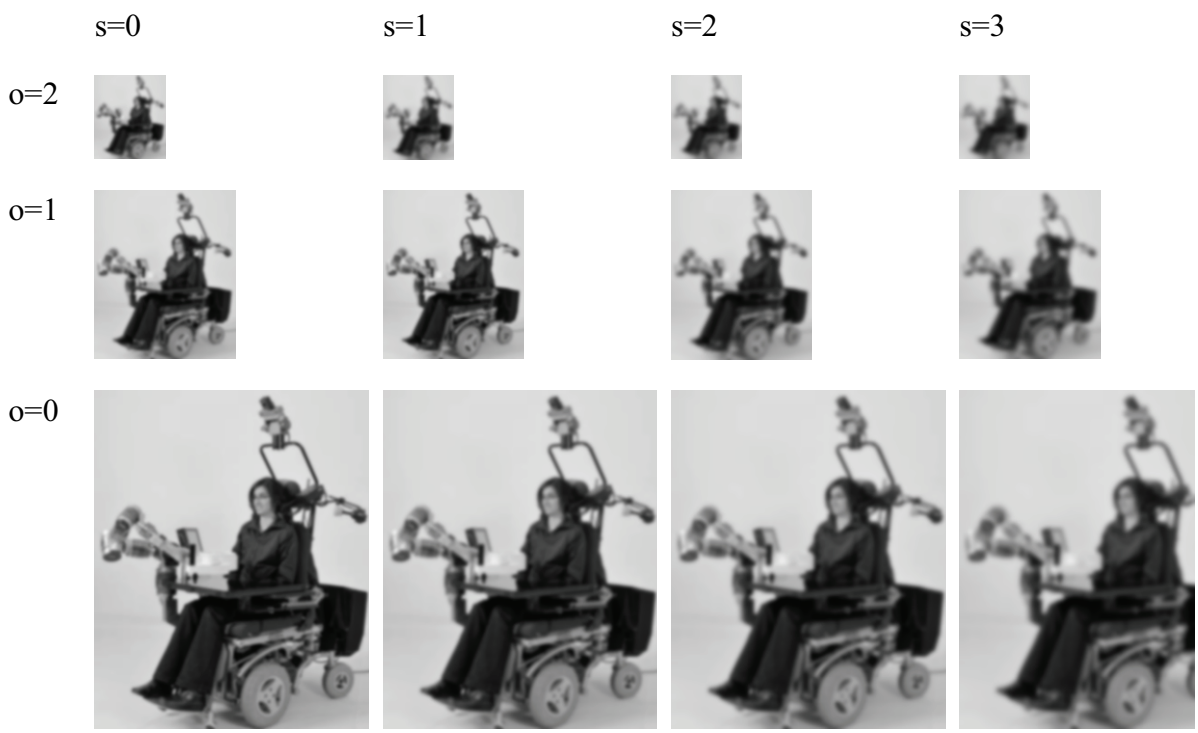


Figure 4.2: A Gaussian scale space consists of 3 octaves, each octave has 4 scale levels.

Once the Gaussian scale space has been obtained, the DoG scale space is computed by subtracting each two consecutive images of each octave as shown in Figure 4.3.

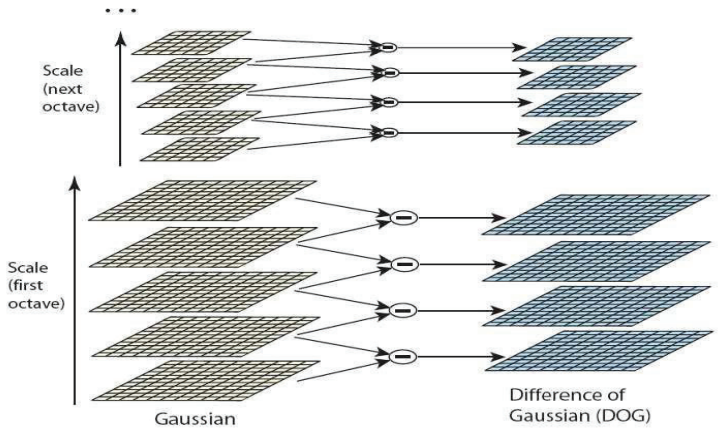


Figure 4.3: Constructing the DoG scale space from the Gaussian scale space [4].

$$D(x, y, \sigma(o, s)) = L(x, y, \sigma(o, s)) - L(x, y, \sigma(o, s-1)) \tag{4.4}$$

The DoG function can be treated as an approximation to the scale-normalized Laplacian of Gaussian [42], which is in fact the general family of solutions to the diffusion equation (4.5):

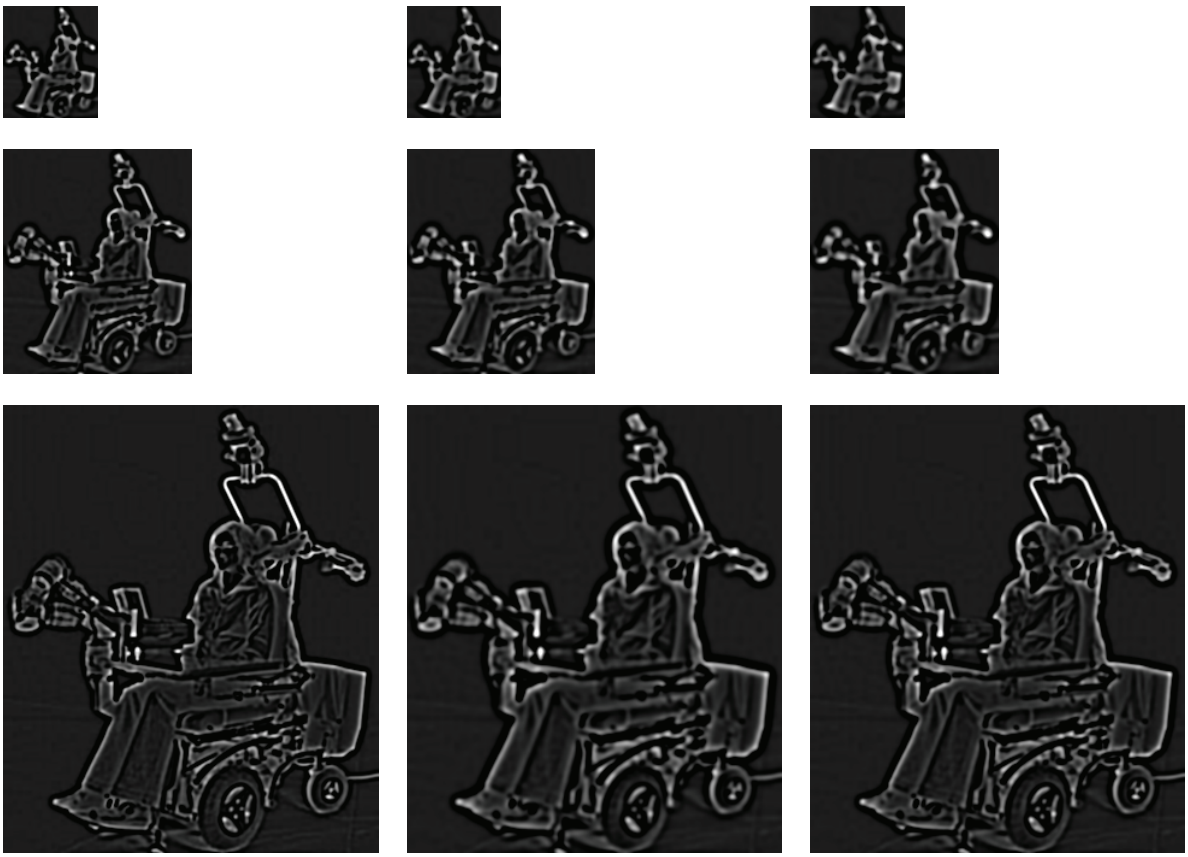


Figure 4.4: The Difference of Gaussian Scale Space

Figure 4.4 presents the DoG resulted from the Gaussian scale space illustrated in Figure 4.2.

$$\frac{\partial L}{\partial \sigma} = \sigma \cdot \nabla^2 L \tag{4.5}$$

Thus the DoG is an approximation to the normalized Laplacian, which is needed for true scale invariance.

$$\sigma \cdot \nabla^2 L \approx \frac{L(x, y, k\sigma) - L(x, y, \sigma)}{k\sigma - \sigma} \Rightarrow D(x, y, \sigma) \approx (k-1)\sigma^2 \cdot \nabla^2 L \tag{4.6}$$

This indicates that the DoG-SS has scales differing by a constant factor, while it incorporates the σ scale normalization required for the scale-invariant Laplacian.

Interest points are characterized as the Extrema (Maxima and Minima) in the 3 dimensional real function $D(x, y, \sigma)$. For searching scale space Extrema, each pixel in the DoG images is compared with the pixels of all its 26 neighbors (8 neighbors at the same scale and 9 neighbors above and 9 neighbors below that scale) as shown in Figure (4.5). If the pixel is lower/larger than all its neighbors, then it is labeled as a candidate interest point.

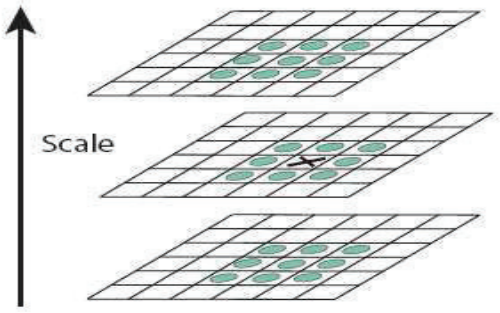


Figure 4.5: Scale-space extrema detection [4].

The scale space (SS) representation of an image mimics the visual perception of the imaged scene viewed at different distances, therefore the feature points extracted from the SS are scale invariant.

4.1.2. Key-Points Localization

Once a key-point candidate has been found by comparing a pixel to its neighbors, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge (and are therefore not enough distinctive).

Each of these key points is exactly localized by fitting a 3D quadratic function computed using a second order Taylor expansion around key-point location.

$$D(z_0 + z) \approx D(z_0) + \left(\frac{\partial D}{\partial z}\bigg|_{z_0} \right)^T z + \frac{1}{2} z^T \left(\frac{\partial^2 D}{\partial z^2}\bigg|_{z_0} \right)^T z \qquad (4.7)$$

where D and its derivatives are evaluated at the key-point location $z_0 = [x_0, y_0, \sigma_0]^T$ and z is the offset from this point.

The location of the extremum $\hat{z}$, is determined by taking the derivative of this function with respect to z and setting it to zero, giving:

$$\hat{z} = -\left(\frac{\partial^2 D}{\partial z^2}\bigg|_{z_0} \right)^{-1} \cdot \left(\frac{\partial D}{\partial z}\bigg|_{z_0} \right) \qquad (4.8)$$

The offset $\hat{z}$ may be estimated using standard difference approximations from neighboring sample points in the DoG resulting in a 3 x3 linear system which may be solved efficiently.

If the offset $\hat{z}$ is larger than 0.5 in any dimension, then it means that the extremum lies closer to a different sample point. In this case, the sample point is changed and the interpolation performed instead about that point. The final offset $\hat{z}$ is added to the location of its sample point to get the interpolated estimate for the location of the extremum. The function value at

the extremum, $D(\hat{z})$, is useful for rejecting unstable Extrema with low contrast. This can be obtained by substituting equation (4.8) into (4.7), giving:

$$D(\hat{x},\hat{y},\hat{\sigma}) = D(z_0 + \hat{z}) \approx D(z_0) + \frac{1}{2}\left(\left.\frac{\partial D}{\partial z}\right|_{z_0}\right)^T \cdot \hat{z} \tag{4.9}$$

All Extrema with a value of $|D(\hat{z})|$ less than a certain threshold are discarded. In the standard SIFT method, a threshold with a value between 0.01 and 0.04 was used assuming that image pixel values are in the range [0,1]

A final test is performed to remove any features located on edges in the image since these will suffer an ambiguity if used for matching purposes. A peak located on an edge in the DoG will have a large principle curvature across the edge and a low principle curvature along it whereas a well defined peak will have a large principle curvature in both directions. The principal curvatures can be computed from a 2x2 Hessian matrix, H, computed at the location and scale of the key-point:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{4.10}$$

The derivatives are estimated by taking differences of neighboring sample points. The eigenvalues of H are proportional to the principal curvatures of D. Borrowing from the approach used by Harris and Stephens [23], we can avoid explicitly computing the eigenvalues, as we are only concerned with their ratio. Assuming that λ_1 is the eigenvalue with the largest magnitude and λ_2 is the smaller one. Then, we can compute the sum of the eigenvalues from the trace of H and their product from the determinant as explained in the following equations:

$$\begin{aligned} Tr(H) &= D_{xx} + D_{yy} = \lambda_1 + \lambda_2 \\ Det(H) &= D_{xx} \cdot D_{yy} - (D_{xy})^2 = \lambda_1 \cdot \lambda_2 \end{aligned} \tag{4.11}$$

In the unlikely event that the determinant is negative, the curvatures have different signs so the point is discarded as not being an extremum. Let r be the ratio between the largest magnitude eigenvalue and the smaller one, so that $r = \lambda_1 / \lambda_2$ Then:

$$\frac{(Tr(H))^2}{Det(H)} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \cdot \lambda_2} = \frac{(r+1)^2}{r} \tag{4.12}$$

The quantity $(r+1)^2/r$ is at a minimum when the two eigenvalues are equal and it increases with r. Therefore, to check that the ratio of principal curvatures is below a certain threshold τ, we only need to check:

$$\frac{(Tr(H))^2}{Det(H)} < \frac{(\tau+1)^2}{\tau} \tag{4.13}$$

which is very efficient to compute.

The standard SIFT method use a value of $\tau = 10$, which eliminates key-points that have a ratio between the principal curvatures greater than 10.

4.1.3. Orientation Assignment

An orientation is assigned to each interest point that combined with the scale provides a scale and rotation invariant coordinate system for the descriptor. Orientation is determined by building a histogram of gradient orientations from the key-point neighborhood.

For each pixel in a certain region R around the key-point location, the first order gradients are calculated. The pixel difference approximations are used to derive the corresponding gradient according to the following equations:

$$\begin{aligned} g_x &= L(x+1, y, \sigma) - L(x-1, y, \sigma) \\ g_y &= L(x, y+1, \sigma) - L(x, y-1, \sigma) \end{aligned} \tag{4.14}$$

where $L(x, y, \sigma)$ is the grey value of the pixel $P(x, y)$ in the image blurred by a Gaussian kernel whose size is determined by the scale of the keypoint σ.

The gradient magnitude and orientation for each pixel are computed respectively as follows:

$$\begin{aligned} m(x, y) &= \sqrt{(g_x)^2 + (g_y)^2} \\ \theta(x, y) &= arctn(g_x / g_y) \end{aligned} \tag{4.15}$$

From gradient data (magnitudes and orientations) of pixels within the region R, a 36-bin orientation histogram is constructed covering the range of orientations [-180°, 180°] (each bin covers 10°). The gradient orientation determines which bin in the histogram should be used for each pixel. The value added to the bin is then given by the gradient magnitude weighted by a Gaussian-weighted circular window with σ that is 1.5 times of the scale of the key-point centered on the feature point, thus limiting to local gradient information. The histogram is calculated according to following formulas:

$$\begin{aligned} ori(i) &= \text{int}(\theta(x, y)/10) - 17 \\ mag(i) &= \sum_R m_i(x, y) \Big/ \sum_R m(x, y) \end{aligned} \tag{4.16}$$

where $\theta(x, y) \in [0°, 360°)$ and $m_i(x, y)$ are gradient magnitudes of pixels that have discrete gradient orientations equal to $Ori(i)$.

The orientation of the SIFT feature is defined as the orientation corresponding to the maximum bin of the orientation histogram according to:

$$\theta_{\max} = ori\left(\arg\max\left(mag(i)\right)\right) \tag{4.17}$$

In order to improve the accuracy of determining the key-point orientation, a three point parabola is fit to the peaks of the orientation histogram.

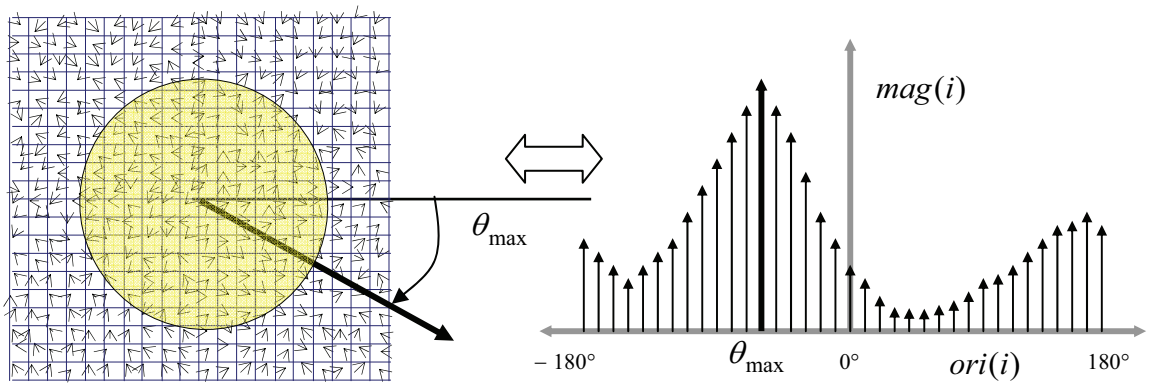


Figure 4.6: A 36 bins orientation histogram constructed using local image gradient data around key-point.

If the histogram has more than one distinct peak then multiple copies of the feature are generated for the direction corresponding to the histogram maximum, and any other direction within 80% of the maximum value. Figure 4.6 explains an example of an orientation histogram for a SIFT feature.

4.1.4. Key-Points Description

The gradient image patch around key-point is rotated to align the feature orientation computed in the previous section with the horizontal direction in order to provide rotation invariance.

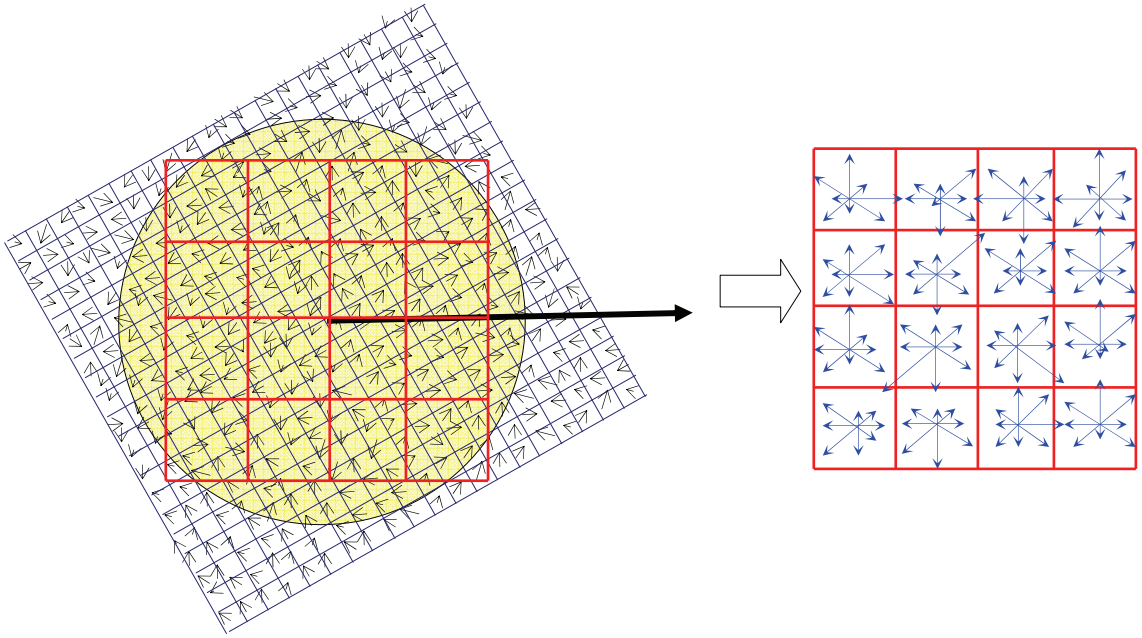


Figure 4.7: SIFT descriptor construction

After that the region around key-point with size related to key-point scale is selected and subdivided into 16 square sub-regions. For each sub-region, an 8 bin orientation histogram is built from pixels within corresponding sub-region. The weight of each pixel is given by the magnitude of the gradient as well as a scale dependent Gaussian window centered on the key-point. During the histogram formation tri-linear interpolation is used to add each value. This consists of interpolation of the weight of the pixel across the neighboring spatial bins based on distance to the bin centers as well as interpolation across the neighboring angle bins. This leads to reduce boundary effects as samples move between positions and orientations.

Finally, all 16 resulting eight bin orientation histograms are transformed into 128-D vector. The vector is normalized to unit length to achieve the invariance against illumination changes. Figure 4.7 shows the descriptor generated from the gradient image patch around key-point. Therefore the SIFT feature consists of four attributes, a location $P(x, y)$ (x and y are the coordinates of the key-point in the image), a scale σ (level of scale space where is the key-point), an orientation $\theta_{\max}$ and a 128-D descriptor vector V that describes the local image region around the key-point location. Hence, SIFT feature can be written as $F(P(x, y), \sigma, \theta_{\max}, V)$.

4.2. SIFT Feature Matching

4.2.1. SIFT Correspondences Search

In order to match two images using SIFT algorithm, SIFT features will be extracted from images and stored into feature sets, then the corresponding features are found using a nearest neighbor search (NNS) method that is able to detect the similarities between SIFT descriptors.

The similarity measure between two SIFT features is defined by the Euclidean distance between its describing 128-vectors.

Essentially each feature F_i^q from the query image is compared to all the features F_j^t in the test image by computing the Euclidean distances $d_{ij}\left(F_i^q, F_j^t\right)$.

$$d_{ij}\left(F_i^q, F_j^t\right) = \sqrt{\sum_{k=1}^{k=128}\left(d_k^q - d_k^t\right)^2} \tag{4.18}$$

The feature pairs with the smallest Euclidian distances are considered as possible positive matches. However, many features from the test image will not have any corresponding feature in the query image because they arise probably from background clutter or are not detected in the query image. Therefore, it is necessary to have a strategy to discard mismatches. A global threshold strategy on distance to the closest feature does not perform well since some descriptors are much more discriminative than others.

Lowe proposed [4] a strategy (called Nearest Neighbor Distance Ratio (NNDR)) to discard mismatches. In this strategy, for each feature from the query image, the Euclidian distances to the nearest and next nearest neighbor features of the test image, are compared. If the ratio between the nearest and the second nearest distances is below a certain threshold, then the match is considered as correct. This approach provides reliable feature matching because the correct matches need to have the closest neighbor significantly closer than the closest

incorrect one. For false matches, it is more likely that the distances to the nearest and next nearest neighbors are similar to each other due to the high dimensionality of the feature space.

The exhaustive search for the nearest neighbor is computationally expensive when the feature length and the number of features are large. The computational expensive problem can be solved by replacing the exhaustive search by Approximate Nearest Neighbor (ANN) search algorithms.

The most widely used algorithm for ANN search is the kd-tree [38,43], which successfully works in low dimensional search space, but performs poorly when feature dimensionality increases. kd-tree algorithm provides no speedup over exhaustive search for more than about 10 dimensional spaces. In [4] Lowe used the Best-Bin-First (BBF) method, which is expanded from kd-tree by modification of the search ordering so that bins in feature space are searched in the order of their closest distance from the query feature and stopping search after checking the first 200 nearest-neighbors. For a database of 100,000 SIFT features, the BBF provides a speedup factor of 2 times faster than exhaustive search while losing about 5% of correct matches.

4.2.2. Mismatches Discarding

Mismatches always occur when features are matched. A set of matches between two images are frequently used to calculate geometrical transformation models like affine transformation, homography or the fundamental matrix. The geometrical transformation model is used to discard mismatches that do not fit it. There are many algorithms that have demonstrated good performance in model fitting, some of them are the Least Median of Squares (LMeds) [44] and Random Sample Consensus (RANSAC) algorithm [45]. Both are randomized algorithms and are able to cope with a large proportion of outliers.

Lowe [4] used Hough Transform to cluster reliable model hypotheses to search for keys that agree upon a particular model pose. Hough transform identifies clusters of features with a consistent interpretation by using each feature to vote for all object poses that are consistent with the feature. The 6 DoF object pose can be approximated by an affine transform with only 4 parameters. Therefore, Lowe used broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times the maximum projected training image dimension (using the predicted scale) for location.

Each identified cluster with at least 3 matches is then subject to a verification procedure in which a linear least squares solution is performed for the parameters of the affine transformation relating the model to the image.

The affine transformation of a model point $\begin{bmatrix} x & y \end{bmatrix}^T$ to an image point $\begin{bmatrix} u & v \end{bmatrix}^T$ can be written as:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{4.19}$$

where the model translation is $\begin{bmatrix} t_x & t_y \end{bmatrix}^T$ and the affine rotation, scale, and stretch are represented by the parameters m_{11}, m_{12}, m_{21} and m_{22}. To solve for the transformation parameters the equation above can be rewritten to gather the unknowns into a column vector.

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_2 & y_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ x_3 & y_3 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_3 & y_3 & 0 & 1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{bmatrix} \cdot \begin{bmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ \ldots. \end{bmatrix} \tag{4.20}$$

This equation shows 3 matches which at least needed to provide a solution, but any number of further matches can be added, with each match contributing two more rows to the first and last matrix. Equation (4.20) can be written in the shorthand form as:

$$Ax = B \tag{4.21}$$

where A is a known *m*-by-*n* matrix (usually with $m > n$), x is an unknown *n*-dimensional parameter vector, and B is a known *m*-dimensional measurement vector.

The solution of the system of linear equations is given by the pseudo inverse of the matrix *A*:

$$x = \left(A^T \cdot A\right)^{-1} . A^T \cdot B \tag{4.22}$$

which minimizes the sum of the squares of the distances from the projected model locations to the corresponding image locations.

Outliers can now be removed by checking for agreement between each image feature and the model, given the parameter solution. Given the linear least squares solution, each match is required to agree within half the error range that was used for the parameters in the Hough transform bins.

As outliers are discarded, the linear least squares solution is re-solved with the remaining points, and the process iterated. If fewer than 3 points remain after discarding outliers, then the match is rejected. In addition, a top-down matching phase is used to add any further matches that agree with the projected model position, which may have been missed from the Hough transform bin due to the affine transform approximation or other errors.

5. Fast SIFT Feature Matching

5.1. Introduction

Matching a given image with one or many others is a key task in many computer vision applications such as object recognition, images stitching and 3D stereo reconstruction. These applications require often real-time performance. The matching is usually done by detecting and describing key points in the images then applying a matching algorithm to search for correspondences.

Classic key-point detectors such as Difference of Gaussians (DoG) [4], Harris Laplacian [46], Laplacian of Gaussians (LoG) [47], Difference of Means (DoM) [6] and the Harris corner detector [23] use simple attributes like blob-like shapes or corners.

For the key-point description a variety of key-point descriptors have been proposed such as the Scale Invariant Feature Transform (SIFT) [4], Speeded Up Robust Features (SURF) [6] and Gradient Location and Orientation Histogram (GLOH) [8].

To robustly match the images, point-to-point correspondences are determined using similarity measure for Nearest Neighbour (NN) search such as Mahalanobis or Euclidean distance. After that, the RANdom Sample Consensus (RANSAC) method [45] is applied to the positive correspondences set to estimate the correct correspondences (inliers).

The combination of the DoG detector and SIFT descriptor proposed in [4] is currently the most widely used in computer vision applications due to the fact that SIFT features are highly distinctive, and invariant to scale, rotation and illumination changes. In addition, SIFT features are relatively easy to extract and to match against a large database of local features. However, the main drawback of SIFT is that the computational complexity of the algorithm increases rapidly with the number of key-points, especially at the matching step due to the high dimensionality of the SIFT feature descriptor.

In order to overcome the main SIFT drawback, various modifications of the SIFT algorithm have been proposed. In general, the strategies dealing with the acceleration of SIFT features matching can be classified into three different categories: reducing the descriptor dimensionality, parallelization and exploiting the power of hardware (GPUs, FGPAs or multi-core systems) and Approximate Nearest Neighbor (ANN) searching methods.

Ke and Thankar [5] applied Principal Components Analysis (PCA) to the SIFT descriptor. The PCA-SIFT reduces the SIFT feature descriptor dimensionality from 128 to 36, so that the PCA-SIFT is fast for matching, but seems to be less distinctive than the original SIFT as demonstrated in a comparative study by Mikolajczyk et al. [8]. In [6] Bay et al. developed the Speeded Up Robust Feature (SURF) method that is a modification of the SIFT method aiming at better run time performance of features detection and matching. This is achieved by two major modifications. In the first one, the Difference of Gaussian (DoG) filter is replaced by a Difference of Means (DoM) filter. The use of the DoM filter speeds up the computation of features detection due to the exploiting integral images for a DoM implementation. The second modification is the reduction of the image feature vector length to half the size of the SIFT feature descriptor length (from 128 components down to 64), which enables quicker features matching. These modifications result in an increase computation speed by a factor 3

compared to the original SIFT method. However, this is insufficient for real-time requirements. Additionally, in contrast to SIFT, SURF does not provide the number of correspondences which are required for some computer vision applications such as pose estimation and 3D reconstruction [48].

In recent years, several papers [49,50] were published addressing the use of the parallelism of modern graphics hardware (GPU) to accelerate some parts of the SIFT algorithm, focused on features detection and description steps. In [51] GPU power was exploited to accelerate features matching. These GPU-SIFT approaches provide 10 to 20 times faster processing allowing real-time application. Other papers such as [52] addressed implementation of SIFT on a Field Programmable Gate Array (FPGA) achieved about 10 times faster processing. Zhan et al. [53] presented that SIFT features extraction rate can be increased by a factor of 6.7 by parallelizing it on an 8-core system, or by a factor 25 on a 32-core chip multiprocessor (CMP) simulator.

The matching step can be speeded up by searching for the Approximate Nearest Neighbor (ANN) instead of the exact nearest neighbor. The most widely used algorithm for ANN search is the kd-tree [54], which successfully works in low dimensional search space, but performs poorly when feature dimensionality increases. In [4] Lowe used the Best-Bin-First (BBF) method, which is expanded from kd-tree by modification of the search ordering so that bins in feature space are searched in the order of their closest distance from the query feature and stopping search after checking the first 200 nearest-neighbor candidates. The BBF provides a speedup factor of 2 times faster than exhaustive search while losing about 5% of correct matches. Silpa-Anan et al. [55] proposed an improved version of the kd-tree algorithm in which multiple randomized kd-trees are created. In contrast to original kd-tree algorithm which splits the data in half at each level of the tree on the dimension for which the data has the greatest variance, in improved version the randomized trees are constructed by selecting the split dimension randomly from among a few dimensions in which the data has high variance. In [41] Gionis et al proposed the Locality Sensitive Hashing (LSH) method, which hashes features using several hash functions into subsets (so called buckets). The main idea is to ensure the collision of similar features with high probability. Like KD-trees, LSH also has a problem when dealing with very high dimensional data. In [56] Heng Yang et al proposed the Randomized Sub-Vector Hashing (RSVH) algorithm for high-dimensional feature matching. The essential idea of RSVH is that two feature vectors are considered similar when the L2 norms of their corresponding randomized sub-vectors are approximately same. RSVH can be executed averagely about 11 times faster than exhaustive search for databases of few ten thousands of SIFT features. In [57] Eduardo Valle et al. introduced multi-curves scheme for indexing high dimensional features to perform ANN search with good compromise between precision and speed. This technique is an improvement to the space- filling curves method aiming at resolve the boundary effects problem. In [58] Michael E. Houle et al. introduced a practical index for approximate similarity queries of large multi-dimensional data sets, called the Spatial Approximation Sample Hierarchy (SASH), which is a multi-level structure of random samples, recursively constructed by building a SASH on a large randomly selected sample of data objects, and then connecting each remaining object to several of their approximate nearest neighbors from within the sample. Queries are processed by first locating approximate neighbors within the sample, and then using the pre-established connections to discover neighbors within the remainder of the data set. In [59] Muja and Lowe compared

many different algorithms for approximate nearest neighbor search on datasets with a wide range of dimensionality and they found that two algorithms obtained the best performance, depending on the dataset and the desired precision. These algorithms used either the hierarchical k-means tree (HKMT) or multiple randomized kd-trees (MRKDTs).

ANN search algorithms are usually based on constructing a multi-cell data structure (eg. tree, hash table,..) in which features are restored, and then applying a search procedure among the cells of this data structure to answer a query, which requires not only matching time but also build time and an additional memory usage. Therefore ANN algorithms are especially suitable for nearest neighbor searching in large databases, since they need offline training and complex data structures.

In this Chapter, a novel strategy which is distinctly different from all three of the above mentioned strategies, is introduced to accelerate the SIFT features matching step. The contribution is summarized in two points. Firstly, in the key-point detection stage, the SIFT features are split into two types, Maxima and Minima, without extra computational cost and at the matching stage only features of the same type are compared. The idea behind this is that no match can be expected between two features of different types. Secondly, SIFT feature is extended by few new angles without extra computational cost. These angles are computed from orientation histogram (OH) and/or sub-orientation histograms (SOHs) of the SIFT descriptor.(SIFT-D). Hence SIFT features are divided into a few clusters based on their angles and, at the matching stage, only features that have almost the same angles are compared since no match can be expected between two features whose angles differ from each other for more than a pre-defined threshold. In comparison to the original SIFT method, where exhaustive search is used for matching, the proposed modifications allow more than **1000** times faster processing in the matching step without losing a noticeable portion of correct matches.

In contrast to ANN search algorithms, proposed strategy requires neither build time nor memory overhead, therefore it is suitable for all applications, especially when online matching is required.

The proposed method can be generalized for all local feature-based matching algorithms which detect two or more types of key-points (e.g. DoG, LoG, DoM) and whose descriptors are rotation invariant, where few different orientations can be assigned (e.g. SIFT, SURF, GLOH). Furthermore, the presented strategy can be combined with other above mentioned strategies to reach a higher factor of features matching speedup.

Since the proposed strategy is mainly based on the statistical distributions of circular random variables (angles), we first give a brief review of the statistical analysis of circular random variables.

5.2. Circular Random Variables

Circular variables [60, 61] take values on the circumference of a circle i.e. they are angles in the range $[0,2\pi)$ radians. Many environmental data are circular in nature such as wind direction, compass bearing, clock and others. To analyze this type of data, it is needed to use techniques differing from those of the usual Euclidean type variables because the circumference is a bounded closed space, for which the concept of origin is arbitrary or undefined. Thus, the techniques that have been used for continuous linear data do not work with circular variables because they assume that variables are linear (the lowest value is

farthest from the highest value). Therefore, to analyze circular variables, an entire field of circular statistics has been developed. In circular statistics, each datum is defined by its length and its angle from a chosen point on the circle.

Circular statistics include tests of uniform direction around the circle, confidence intervals, circular probability density functions, correlations, and regression, among others.

In the following we will study the probability density function of the sum/ difference of two or more independent circular random variables (ICRVs).

5.2.1. PDF of Sum/Difference of Uniformly-Distributed ICRVs

From the probability theory, it is known that the probability density function $g(x)$ of the sum of two independent random variables X_1 and X_2, each of which has a probability density function $g_1(x)$ and $g_2(x)$ respectively, is the convolution of their individual density functions:

$$g(x) = \int_{-\infty}^{+\infty} g_1(\lambda) \cdot g_2(x-\lambda) d\lambda = g_1(x) * g_2(x) \tag{5.1}$$

If X_1 and X_2 are uniformly distributed in the interval $[0,2\pi]$, then the PDF of the sum $X = X_1 + X_2$ is triangular-distributed in the interval $[-2\pi,2\pi]$ because the convolution of two rectangular functions is triangular.

If X_1 and X_2 are circular variables with period 2π, then the sum is also periodic with the same period. Hence the left part of the PDF of the sum in the interval $[-2\pi,0]$ can be shifted to right by the period 2π and summed to the right part in the interval $[0,2\pi]$ to produce the total PDF of the sum $X = X_1 + X_2$.

Therefore the sum of two independent uniformly-distributed circular random variables is also uniformly-distributed. This outcome is graphically illustrated by Figure 5.1.

The same result is also valid for the difference because the difference between two values can be expressed as the sum of the first one and the negative of the second:

$$X_1 - X_2 = X_1 + (-X_2) \tag{5.2}$$

To prove this, it is sufficient to prove that the PDF of $(-X_2)$ is equal to the PDF of X_2

Because $(-X_2)$ is periodic with period 2π, its PDF $g_2(-x)$ can be shift to right by its period.

$$g_2(-x) = g_2(-x + 2\pi) = g_2(x) \tag{5.3}$$

which leads to the fact that the PDF of $(-X_2)$ is equal to the PDF of X_2.

Hence the PDF of the sum/ difference of two independent uniformly distributed circular variables is uniformly-distributed. The same result can be easy generalized to any number of independent uniformly circular random variables.

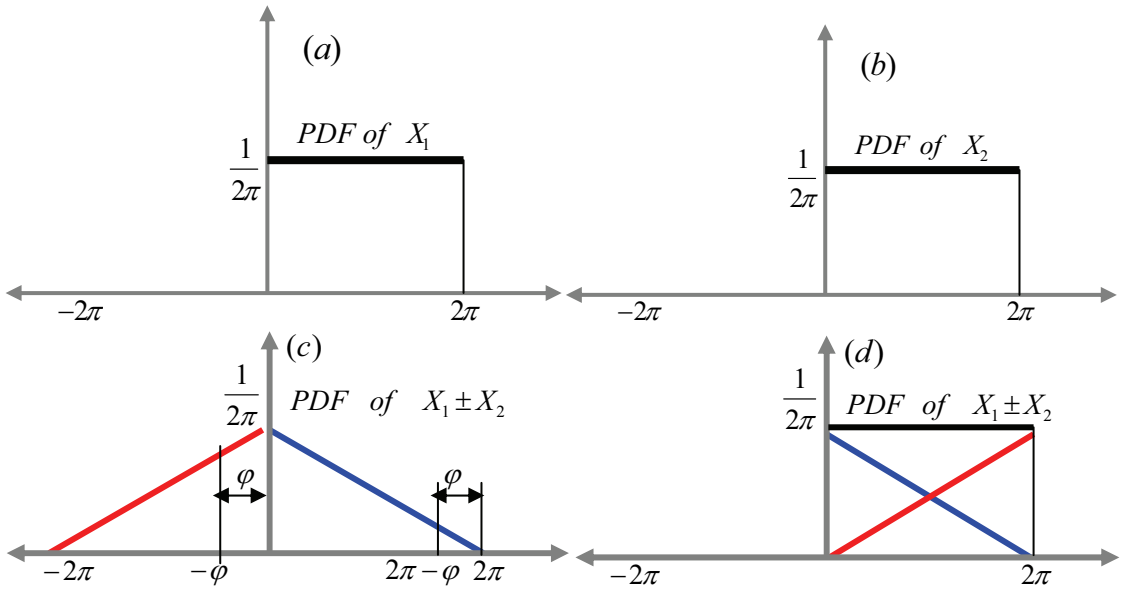


Figure 5.1: The circular probability density function of the sum of two independent uniformly distributed circular random variables.

5.2.2. PDF of Sum/Difference of ICRVs

The result proven in the above can be proven even only one of two independent circular random variables X_1 and X_2 is uniform.

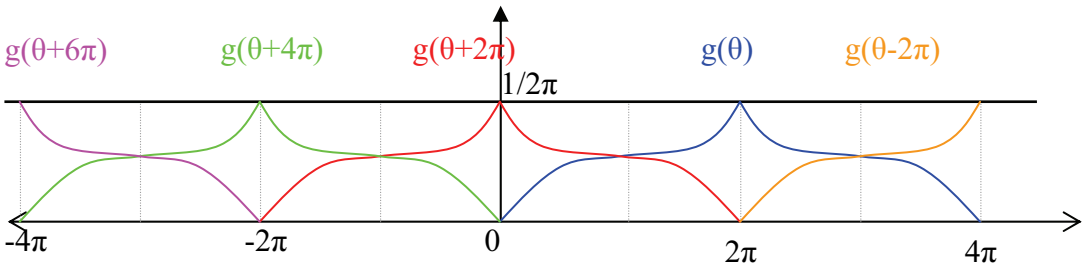


Figure 5.2: wrapping the $g(x)$ around the circumference of a circle of unit radius

For example, if only X_1 is uniformly distributed in the interval $[0,2\pi]$, whereas the other X_2 is arbitrary distributed in the same interval, then the sum/difference of these two random variables $X = X_1 \pm X_2$ is uniformly distributed in the same interval.

To prove this, it is assumed that the probability density functions of X_1 and X_2 on the real line are $g_1(x)$ and $g_2(x)$ respectively.

$$g_1(x)=\begin{cases}1/2\pi & 0\le x<2\pi\\ 0 & otherwise\end{cases}$$
$$g_2(x)=\begin{cases}h(x) & 0\le x<2\pi\\ 0 & otherwise\end{cases} \tag{5.4}$$

The probability density function of the sum is the convolution of the individual PDFs

$$g(x)=g_1(x)*g_2(x) \tag{5.5}$$

Transforming the equations (5.4) into the Laplace space yields:

$$g_1(x)\xrightarrow{s}=G_1(s)=\frac{1}{2\pi s}\left(1-e^{-2j\pi s}\right)$$
$$g_2(x)\xrightarrow{s}=G_2(s) \tag{5.6}$$

Because the convolution of two functions in real space is equivalent to the product of their Laplace transforms in Laplace space, the equation (5.5) in Laplace space is expressed as:

$$G(s)=G_1(s)\cdot G_2(s)=\frac{1}{2\pi s}\left(G_2(s)-G(s)e^{-2j\pi s}\right) \tag{5.7}$$

The PDF of the sum $X=X_1\pm X_2$ on the real line $g(x)$ is obtained by inverting the Laplace-space expression (5.7) back to real space:

$$G(s)\xrightarrow{x}g(x)=\frac{1}{2\pi}\left(\int_0^x g_2(\lambda)d\lambda-\int_0^x g_2(\lambda-2\pi)d\lambda\right)$$

$$g(x)=\begin{cases}\dfrac{1}{2\pi}\int g_2(\lambda)d\lambda & 0\le x<2\pi\\ \dfrac{1}{2\pi}\left(\displaystyle\int_0^{2\pi} g_2(\lambda)d\lambda-\int_{2\pi}^x g_2(\lambda-2\pi)d\lambda\right) & 2\pi\le x<4\pi\\ \dfrac{1}{2\pi}\left(\displaystyle\int_0^{2\pi} g_2(\lambda)d\lambda-\int_{2\pi}^{4\pi} g_2(\lambda-2\pi)d\lambda\right) & x\ge 4\pi\end{cases} \tag{5.8}$$

$$g(x)=\begin{cases}\dfrac{1}{2\pi}\int g_2(\lambda)d\lambda & 0\le x<2\pi\\ \dfrac{1}{2\pi}\left(1-\displaystyle\int_{2\pi}^x g_2(\lambda-2\pi)d\lambda\right) & 2\pi\le x<4\pi\\ 0 & x\ge 4\pi\end{cases}$$

The circular random variables Φ corresponding to X, is defined by.

$$\Phi \equiv X[\text{mod} \quad 2\pi] \tag{5.9}$$

The probability density function $f(\theta)$ of Φ is obtained by wrapping $g(x)$ around the circumference of a circle of unit radius [91]:

$$f(\theta) = \sum_{k=-\infty}^{+\infty} g(\theta + 2\pi k) \tag{5.10}$$

For the interval $[0,2\pi]$ $k = 0,+1$, hence

$$f(\theta) = g(\theta) + g(\theta + 2\pi) \tag{5.11}$$

From equation (5.8) yields:

$$g(\theta + 2\pi) = \begin{cases} \dfrac{1}{2\pi} \displaystyle\int_0^{\theta+2\pi} g_2(\lambda) d\lambda & 0 \le \theta + 2\pi < 2\pi \\ \dfrac{1}{2\pi}\left(1 - \displaystyle\int_{2\pi}^{\theta+2\pi} g_2(\lambda - 2\pi) d\lambda\right) & 2\pi \le \theta + 2\pi < 4\pi \\ 0 & \theta \ge 4\pi \end{cases}$$

$$g(\theta + 2\pi) = \begin{cases} \dfrac{1}{2\pi} \displaystyle\int_0^{\theta+2\pi} g_2(\lambda) d\lambda & -2\pi \le \theta < 0 \\ \dfrac{1}{2\pi}\left(1 - \displaystyle\int_{2\pi}^{\theta+2\pi} g_2(\lambda - 2\pi) d\lambda\right) & 0 \le \theta < 2\pi \\ 0 & \theta \ge 4\pi \end{cases} \tag{5.12}$$

By substituting (5.8) and (5.12) in (5.11), yields:

$$f(\theta) = \frac{1}{2\pi} \int_0^{\theta} g_2(\lambda) d\lambda + \frac{1}{2\pi}\left(1 - \int_{2\pi}^{\theta+2\pi} g_2(\lambda - 2\pi) d\lambda\right) = \frac{1}{2\pi} \quad 0 \le \theta < 2\pi \tag{5.13}$$

Equation (5.13) means that the probability density function of the sum/difference of two independent circular random variables, one of them is uniformly distributed in the interval $[0,2\pi]$, is uniformly distributed in the same interval. This result can be also generalized for any number of independent circular random variables at least one of them is uniformly-distributed.

5.3. Split SIFT Feature Matching

As said in Chapter 4, the SIFT feature locations are detected as the Extrema of the scale space. Extrema can be Minima or Maxima so that there are two types of SIFT features, Maxima and Minima SIFT features [63, 64] Through extraction of SIFT features from 600 different images of standard dataset [65], it was found that the number of Maxima is almost equal to the number of Minima SIFT features extracted from the same image. Therefore, when matching only Maxima with Maxima and Minima with Minima, the matching time is reduced by 50% with respect to the exhaustive search without losing any correct matches because no correct match can be expected between two features of different types. The claim that there are no correct matches between Minima and Maxima SIFT features is experimentally supported. Namely, it was found that the features of each correct match are always from the same type.

Figure 5.3 presents the Maxima and the Minima SIFT features extracted from the same image. It can be seen from Figure 5.3 that the Maxima SIFT feature locations are the centers of dark blobs on the light background and vice versa for the Minima locations.

Maxima SIFT features Minima SIFT features

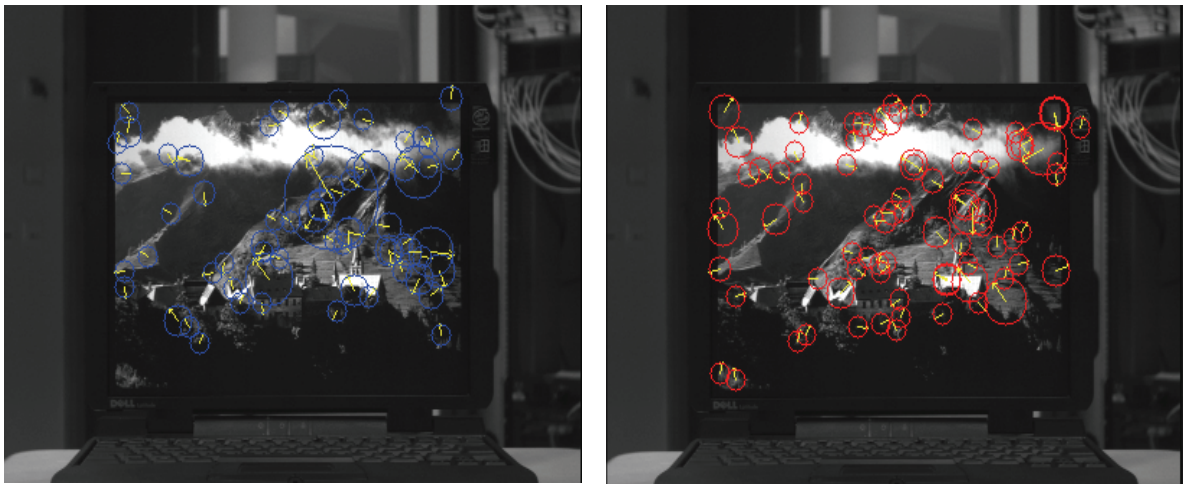


Figure 5.3: the Maxima and Minima SIFT features extracted from the same image.

To declare the matching time reduction by splitting the SIFT features; it is assumed that the number of features extracted from the right and the left image are expressed as:

$$\begin{aligned} r &= r_{\max} + r_{\min} \\ l &= l_{\max} + l_{\min} \end{aligned} \tag{5.14}$$

where $r_{\max}(l_{\max})$ and $r_{\min}(l_{\min})$ are the numbers of Maxima and Minima SIFT features respectively.

The matching time without regard to the type of features, also the time of exhaustive search is proportional to:

$$T_{exh} = r \cdot l \tag{5.15}$$

The matching time, in the case of comparison of only features of the same type, is proportional to the following sum:

$$T_{split} = r_{\max} \cdot l_{\max} + r_{\min} \cdot l_{\min} \tag{5.16}$$

Because the number of Minima SIFT features is almost equal to the number of Maxima SIFT features extracted from the same image:

$$\begin{aligned} r_{\max} &\cong r_{\min} \cong r/2 \\ l_{\max} &\cong l_{\min} \cong l/2 \end{aligned} \tag{5.17}$$

By substituting (5.17) in (5.16) one obtains:

$$T_{split} = \frac{r \cdot l}{2} = \frac{T_{exh}}{2} \tag{5.18}$$

which means that the matching time is decreased by 50% in respect to exhaustive search.

To get this matching time reduction, it is sufficient that at least one of the two feature sets meets the assumption that the number of Maxima is almost equal to the number of Minima. For example, if all SIFT features of set R are Maxima $r = r_{\max}$, then they are compared only with the Maxima-SIFT features of the set L. Hence the equation (5.18) becomes:

$$T_{split} = r \cdot l_{\max} \cong \frac{r \cdot l}{2} = \frac{T_{exh}}{2} \tag{5.19}$$

Therefore, in the case of matching a query image against a large database, there are no necessity to split SIFT features of the query image.

In order to examine this result experimentally, 200 pairs of stereo images are matched using SIFT method with and without splitting SIFT features. Some results are listed in Table 5.1

The test images are acquired from working environment of the robotic system FRIEND II with its stereo camera system (A Bumblebee 2 stereo camera with the resolution of. 1024X768 pixels)

Table 5.1: Comparison between Standard and Split SIFT Feature matching

Nr. of key-points		Standard SIFT Feature Matching		Split SIFT Feature Matching	
Left image	Right image	Matching time (sec)	Nr. of inliers	Matching time (sec)	Nr. of inliers
645	732	0,686	237	0,311	331
777	640	0,790	264	0,330	395
676	621	0,760	205	0,360	383

671	621	0,810	251	0,390	356

As evident from Table 5.1, by splitting SIFT features, not only the matching time is reduced to 50% but also the number of inliers (correct matches) is increased, which means that the matching quality is also enhanced by Split SIFT feature matching.

5.4. Extended SIFT Feature

Generally, if a scene is captured by two cameras or by one camera but from two different viewpoints, the corresponding points, which represent images of the same 3D point, will have different image coordinates, different scales, and different orientations, though, they must have almost similar descriptors that are used to match the images using a similarity measures. However, the high dimensionality of the SIFT descriptor V makes the feature matching very time-consuming.

In order to speed up the features matching, it is assumed that two independent orientations can be assigned to each feature so that the angle ϕ between them stays almost unchanged for all correct corresponding features even in the case of the images captured under different conditions such as viewing geometry and illumination changes. The idea of using an angle between two independent orientations is aimed to avoid comparing of a great portion of features that can not be matched in any way. This leads to a significant acceleration of the matching step. Hence, the reason for proposing SIFT feature angle ϕ is twofold. On the one hand, to filter the correct matches, so that a correct match M_{ij} can be established between two features F_i^1 and F_j^2, which belong respectively to images 1 and 2, if and only if the difference between their angles ϕ_i^1 and ϕ_j^2 is less than a preset threshold value ε:

$$\left|\Delta\phi\right| = \left|\phi_i^1 - \phi_j^2\right| \le \varepsilon \tag{5.20}$$

On the other hand, the reason for proposing SIFT feature angle ϕ is to accelerate the SIFT feature matching because there is no necessity to compare two features if the difference between their angles is larger than the preset threshold ε.

5.4.1. Matching Speeded-Up Factor

Assuming that two images to be matched whose feature angles $\{\phi_i^1\}$ and $\{\phi_j^2\}$ are considered as random variables Φ_1 and Φ_2 respectively. In the case of correct matches the random variables Φ_1 and Φ_2 are dependent on each other since the angle differences of correct matches are equal to zero which correspond to the ideal image matching case. In contrast, the random variables Φ_1 and Φ_2 are independent of each other for incorrect matches while the angle differences of incorrect matches are somehow distributed in the range $[-\pi,+\pi]$. Therefore, the difference $\Delta\Phi = \Phi_1 - \Phi_2$ for the incorrect matches has a probability density function (PDF) distributed over the whole angle range $[-\pi,+\pi]$, whereas the PDF of $\Delta\Phi$ for the correct matches is concentrated in the so-called range of correct matches, which is the narrow range about 0°. Generally, if the random variables Φ_1 and Φ_2 are independent and at least one of them is uniformly distributed in the range $[-\pi,+\pi]$, their difference $\Delta\Phi = \Phi_1 - \Phi_2$ has an uniform PDF as it has been proven in Section 5.2.2.

If a matching procedure, which compares only the features having angle differences $\Delta\Phi$ in the range of correct matches, is used in the case of uniform distribution of $\Delta\Phi$ for incorrect matches, then the matching process is accelerated by a speed-up factor SF which can be expressed as the ratio between the width of the whole angle range $w_{total} = 360°$ and the width of the range of correct matches w_{corr}.

$$SF = \frac{w_{total}}{w_{corr}} = \frac{360°}{w_{corr}} \tag{5.21}$$

5.4.2. SIFT Feature Angle

It is suggested here that a SIFT feature is extended with an angle that meets the following conditions:

1- The angle has to be invariant to the geometric and the photometric transformations (the invariance condition).

2- The angle has to be uniformly distributed in the range $[-\pi,+\pi]$ (the equally likely condition).

To assign an angle to the SIFT feature, two orientations are required. The invariance condition is guaranteed only if these orientations are different, whereas, as explained in above Section, the equally likely condition is guaranteed if the orientations are independent and at least one of them is uniformly distributed in the range $[-\pi,+\pi]$

As mentioned in Chapter 4, the original SIFT feature has already an orientation $\theta_{\max}$. Therefore, it is only necessary to define a different orientation independent from $\theta_{\max}$.

Firstly, the orientation θ_{sum} corresponding to the vector sum of all orientation histogram bins is considered and the difference between the suggested orientation and the original SIFT feature orientation $\phi_{sum} = \theta_{sum} - \theta_{\max}$ is assigned to the SIFT feature as the SIFT feature angle $\phi = \phi_{sum}$. Figure 5.4 presents geometrically the vector sum of an eight bins orientation histogram for the sake of simplicity, whereas the used orientation histogram has 36 bins for the case of the original SIFT. Hence, mathematically, the proposed orientation θ_{sum} is calculated according to the following equation:

$$\theta_{sum} = \arctan\left(\frac{\sum_{i=-17}^{18} mag(i)\cdot\sin(ori(i))}{\sum_{i=-17}^{18} mag(i)\cdot\cos(ori(i))}\right) \tag{5.22}$$

where $mag(i)$ and $ori(i)$ are the amplitude and the orientation of the i^{th} bin of the orientation histogram

Since θ_{sum} is different from $\theta_{\max}$ and both are calculated from the orientation histogram, then ϕ_{sum} meets the invariance condition.

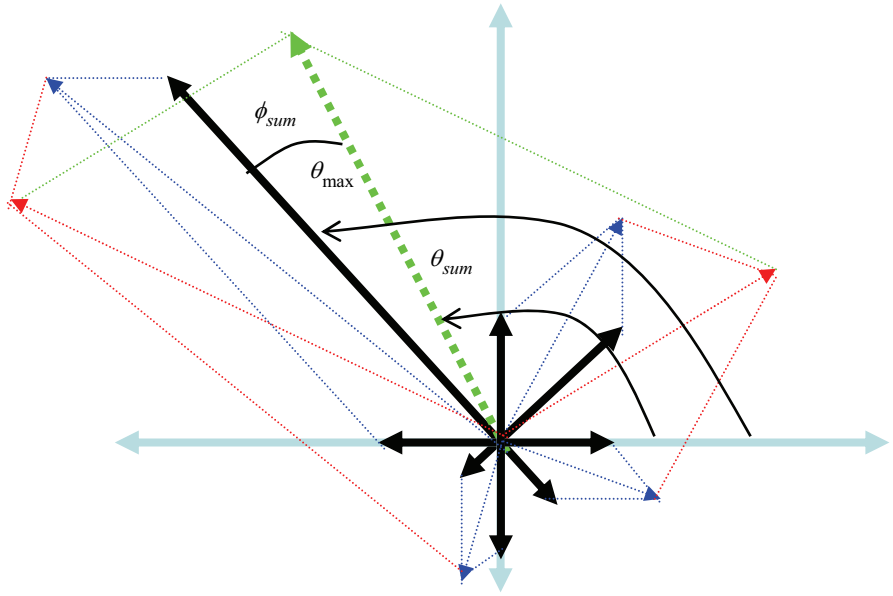


Figure 5.4: The vector sum of the bins of an eight orientation histogram.

To examine whether ϕ_{sum} meets the equally likely condition, it is considered as a random variable Φ_{sum}. The probability density function (PDF) of Φ_{sum} is estimated using 10^6 SIFT features extracted from 700 different images (500 benchmark images [65] and 200 stereo images from a real-world robotic application). Some examples of used images are given in Section 5.4.4.

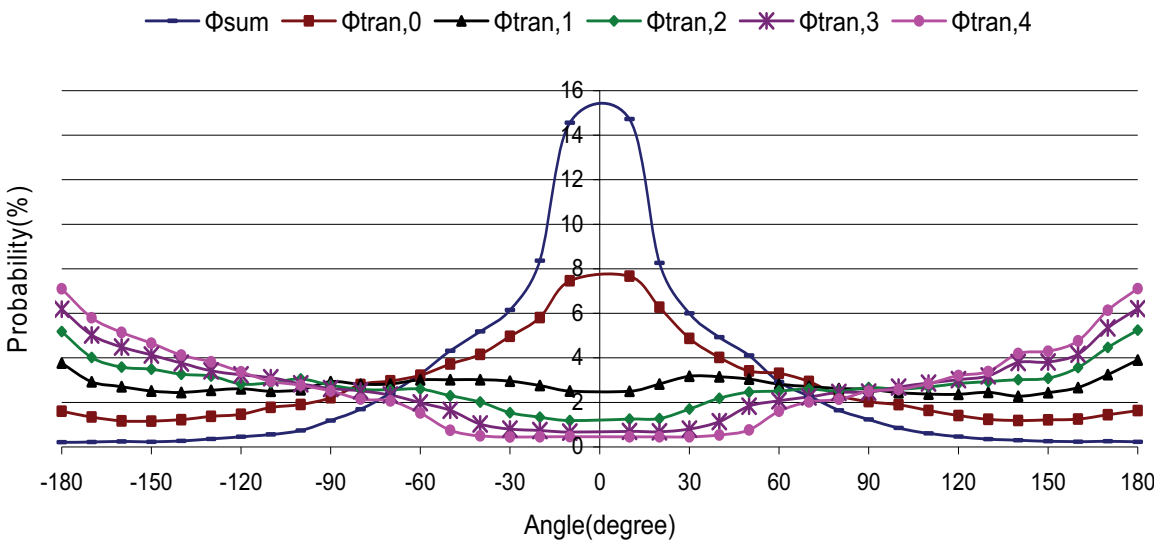


Figure 5.5: The experimental PDFs of Φ_{sum} and $\Phi_{tran,k}$ for SIFT features extracted from 700 test images.

The PDF of Φ_{sum} was computed by dividing the angle space [-180°,180°] into 36 sub-ranges, where each sub-range covers 10°, and by counting the numbers of SIFT features whose

angles ϕ_{sum} belong to each sub-range. For example, an estimate of the probability that a feature has an angle ϕ_{sum} in the sub-range [i°, i+10°) is :

$$p\left(\phi_{sum} \in \left[i°, i°+10°\right)\right) = \frac{100 \cdot N_{\phi_{sum} \in [i°, i°+10°)}}{N_{total}} \tag{5.23}$$

where $N_{\phi_{sum} \in [\alpha°, \alpha°+10°)}$ is the number of SIFT features having the angle in the considered sub-range and N_{total} is the total number of features 10^6 extracted from 700 test images in performed experiments.

As evident from Figure 5.5, about 60% of SIFT features have angles falling in the range [-30°,30°]. The reason of this outcome is the high dependency between $\theta_{\max}$ and θ_{sum} due to the fact that the θ_{sum} is defined as the vector sum of all orientation histogram bins including the bin which corresponds to $\theta_{\max}$. The $\theta_{\max}$ is the dominant orientation in the patch around the key-point so that it has dominant influence to the θ_{sum}. Due to the high dependency between $\theta_{\max}$ and θ_{sum}, ϕ_{sum} does not meet the equally likely condition, hene it can not provide the optimum speed up factor.

To define an appropriate SIFT feature angle, orientations $\theta_{tran,k}$ are further suggested to be considered as independent from $\theta_{\max}$. These orientations are computed as the vector sums of all orientation histogram bins excluding the maximum bin and k of its neighbor bins at the left and at the right side as follows:

$$\theta_{tran,0} = \arctan\left(\frac{\sum_{\substack{i=-17 \\ i\neq m}}^{18} mag(i)\cdot\sin(ori(i))}{\sum_{\substack{i=-17 \\ i\neq m}}^{18} mag(i)\cdot\cos(ori(i))}\right)$$

$$\theta_{tran,1} = \arctan\left(\frac{\sum_{\substack{i=-17 \\ i\notin[m-1,m+1]}}^{18} mag(i)\cdot\sin(ori(i))}{\sum_{\substack{i=-17 \\ i\notin[m-1,m+1]}}^{18} mag(i)\cdot\cos(ori(i))}\right) \tag{5.24}$$

.

.

.

$$\theta_{tran,\kappa} = \arctan\left(\frac{\sum_{\substack{i=-17 \\ i\notin[m-\kappa,m+\kappa]}}^{18} mag(i)\cdot\sin(ori(i))}{\sum_{\substack{i=-17 \\ i\notin[m-\kappa,m+\kappa]}}^{18} mag(i)\cdot\cos(ori(i))}\right)$$

where. $m = \arg\max(mag(i))$.

The PDFs of the random variables $\Phi_{tran,k}$ corresponding to angles $\phi_{tran,k} = \theta_{tran,k} - \theta_{\max}$ are estimated in the same manner as the PDF of Φ_{sum}, performing the experiments over 10^6 SIFT features extracted from 700 test images. The measured PDFs of $\Phi_{tran,k}$ (for $k = 0,1,2,3,4.$) are shown in Figure 5.5.

It is evident from Figure 5.5 that the $\Phi_{tran,1}$ has a PDF that is the closest match to the uniform distribution. Therefore, the angle $\phi_{tran,1}$ meets the both conditions, invariance and equally likely conditions, and it can be considered as a new attribute ϕ of the SIFT feature, that is $\phi = \phi_{tran,1}$. With this extension the SIFT feature becomes $F(x, y, \sigma, \theta_{\max}, V, \phi)$.

5.4.3. Extended SIFT Features Matching

Assuming that two sets of extended SIFT features $R = \{F_i^r : i = 1,2,...,r\}$ and $L = \{F_j^l : i = 1,2,...,l\}$, containing respectively r and l features, are given, The number of possible $M_{ij}(F_i^r, F_j^l)$ matches is equal to $r \cdot l$. Among these possible matches a small number of correct matches may exist, which are determined by Euclidian distance between feature descriptors followed by the RANSAC method [45] to keep only inliers.

A set of SIFT feature angle differences $\{\Delta\phi_{ij} = \phi_i^r - \phi_j^l : ij = 1,2,...,r \cdot l\}$ can be established from the angles $\{\phi_i^r : i = 1,2,...,r\}$ and $\{\phi_j^l : j = 1,2,...,l\}$ of the extended SIFT features of the given sets R and L.

Considering the angle differences $\Delta\phi_{ij}$ as a random variable $\Delta\Phi_{ij}$, the PDFs of $\Delta\Phi_{ij}$ for both correct and incorrect matches are measured in experiments over considered 700 images. The measured PDFs are shown in Figure 5.6.

It can be seen from Figure 5.6 that about 98 % of correct and only 12% of incorrect matches belong to [-20°,20°]. Therefore, in order to find correct matches it is needed to treat only 12% of possible matches which can speed up the features matching significantly.

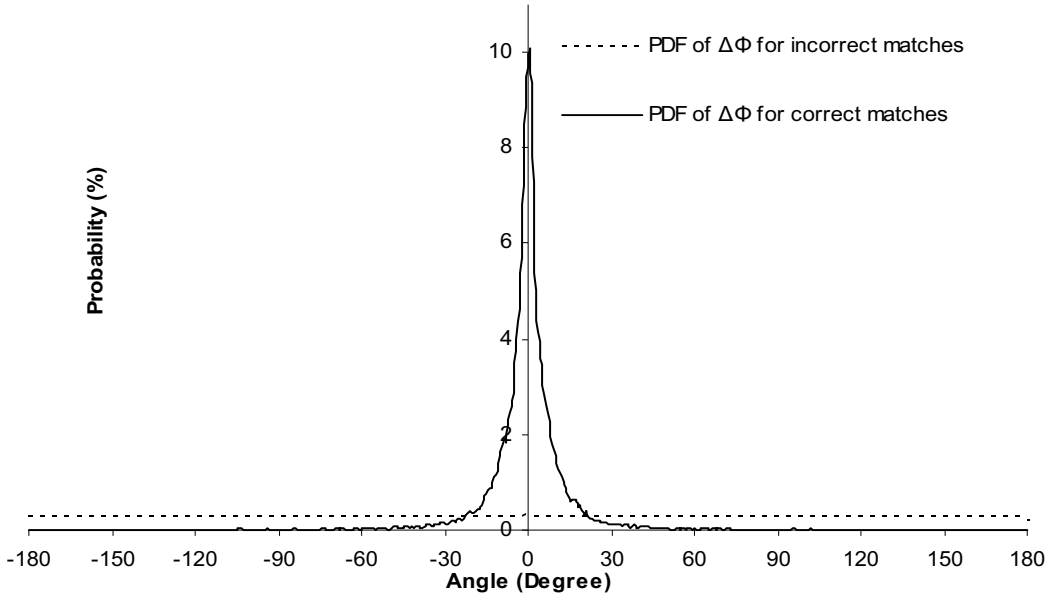


Figure 5.6: The experimental PDF of the angle difference $\Delta\Phi_{ij}$ for incorrect and correct matches.

To exploit this outcome, SIFT features are divided into several subsets based on their angles. The SIFT features of each subset are compared only with the features of some subsets, so that the resulting correspondences must have absolute differences of angles less than a pre-set threshold. Here a threshold of 20° is selected because almost all correct matches have angle differences in the range [-20°,20°] as illustrated in Figure 5.6 .

Consider that each of the sets of features R and L are divided into b subsets, so that the first subset contains only the SIFT features whose angles belong to $\left[-180°,-180°+360°/b\right]$ and the i^{th} subset contains features whose angles belong to $\left[-180°+\left(360°\cdot\left(i-1\right)\right)/b,-180°+\left(360°\cdot i\right)/b\right]$. Consequently, the b^{th} subset contains features whose angles belong to $\left[-180°+\left(360°\cdot\left(b-1\right)\right)/b,180°\right]$.

The number of features of both sets can be expressed as:

$$\begin{aligned} r &= r_0 + r_1 + ... + r_{b-1} \\ l &= l_0 + l_1 + ... + l_{b-1} \end{aligned} \tag{5.25}$$

Because of the evenly distribution of feature angles over the range of their angles [-180°,180°] as shown in Figure 5.5, the features are almost equally divided into several subsets. Therefore, it can be asserted that the feature numbers of each subset are almost equal to each other.

$$\begin{aligned} r_0 &\cong r_1 \cong ... \cong r_{b-1} \cong r/b \\ l_0 &\cong l_1 \cong ... \cong l_{b-1} \cong l/b \end{aligned} \tag{5.26}$$

To exclude matching of features that have differences of angles outside the range $[-a°, a°]$, each subset is matched to its corresponding one and to n neighboring subsets to the left and to the right side. In this case the matching time is proportional to the following term:

$$\begin{aligned} T_{extended} &= \sum_{i=0}^{b-1}\left(r_i \cdot \sum_{j=i-n}^{i+n} l_j \right) \cong \frac{r \cdot l}{b^2} \cdot \sum_{i=0}^{b-1}\left(\sum_{j=i-n}^{i+n} (1) \right) \\ T_{extended} &= \frac{r \cdot l}{b^2} \cdot b \cdot (2n+1) = \frac{r \cdot l \cdot (2n+1)}{b} \end{aligned} \tag{5.27}$$

Therefore, the achieved speedup factor with respect to exhaustive search is equal to:

$$SF_{extended} = \frac{b}{2n+1} \tag{5.28}$$

The relation between n, a and b is as follows:

$$(2n+1) \cdot \frac{360°}{b} = 2a \Rightarrow \left\lceil \left(\left(\frac{2ab}{360} - 1 \right) \Big/ 2 \right) \right\rceil \tag{5.29}$$

where $\lceil x \rceil$ represents the first integer value larger than or equal to x.

Substituting equation (5.28) into equation (5.27) yields:

$$SF_{extended} = \frac{360}{2a} \tag{5.30}$$

The matching procedure is illustrated in Figure 5.7 for the case of comparison of features with the angles from few ranges. For example, features with the angles in the range of $[0°, 360°/b]$, which are extracted from the first image, are compared only with the features extracted from the second image that have angles in the range of $[-n \cdot 360°/b, (n+1) \cdot 360°/b]$.

It is important to indicate that the achieving of the above speedup factor requires the uniform distribution of SIFT features based on their angles only for one of the feature sets. For example, if all SIFT features of the set R falls in the interval $[-180° + (360° \cdot (i-1))/b, -180° + (360° \cdot i)/b]$ the number of features is $r = r_i$.

In this case all SIFT features of set R are compared only with the SIFT features of the set L that fall in the corresponding interval and its specified neighbors. Hence the equation (5.27) becomes:

$$\begin{aligned} T_{extended} &= r \cdot \left(\sum_{j=i-n}^{i+n} l_j \right) = \frac{r \cdot l}{b} \cdot \sum_{j=i-n}^{i+n} (1) \\ T_{extended} &= \frac{r \cdot l \cdot (2n+1)}{b} \end{aligned} \tag{5.31}$$

Therefore, in the case of matching a query image against a large database, there are no necessity to split SIFT features of the query image based on their angles. In addition the assumption that the SIFT features of the database are uniformly distributed based on their angles in the range [-180°, 180°] is valid with a high probability.

$$\lim_{z\to\infty}(p_1)=\lim_{z\to\infty}(p_i)=\ldots=\lim_{z\to\infty}(p_b)=\frac{1}{b} \tag{5.32}$$

where z is the size of the database and p_i is the probability that a feature belongs to the i^{th} subset.

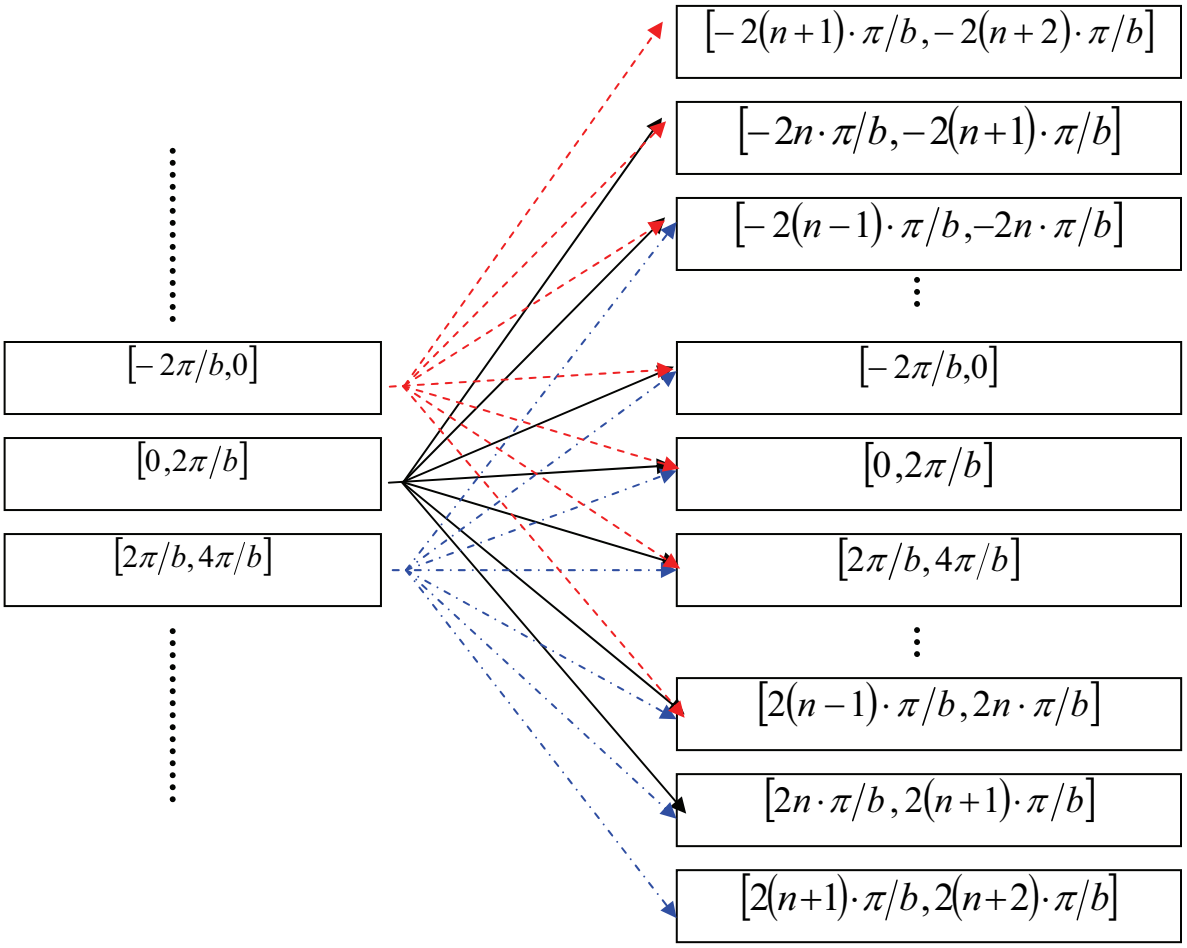


Figure 5.7: Extended SIFT feature matching procedure

The result (5.29) means that if it is aimed to exclude matching of features that have angle differences outside the range [-20°,20°], then the matching step is accelerated by a factor 9. When this modification of original SIFT feature matching is combined with the split SIFT features matching, the obtained speedup factor is 18 without losing a notable portion of correct matches. This is illustrated with the experimental results presented in the next Section.

Figure 5.8 presents the correspondence SIFT features extracted from two images of the same scene imaged from two different viewpoints. SIFT feature are represented by colored circles (blue for Maxima and red for Minima) with radius proportional to the feature scale. Feature

angle is represented by two directions. It can be seen from Figure 5.8 that correspondence SIFT features are always from the same type and have almost the same angles. .

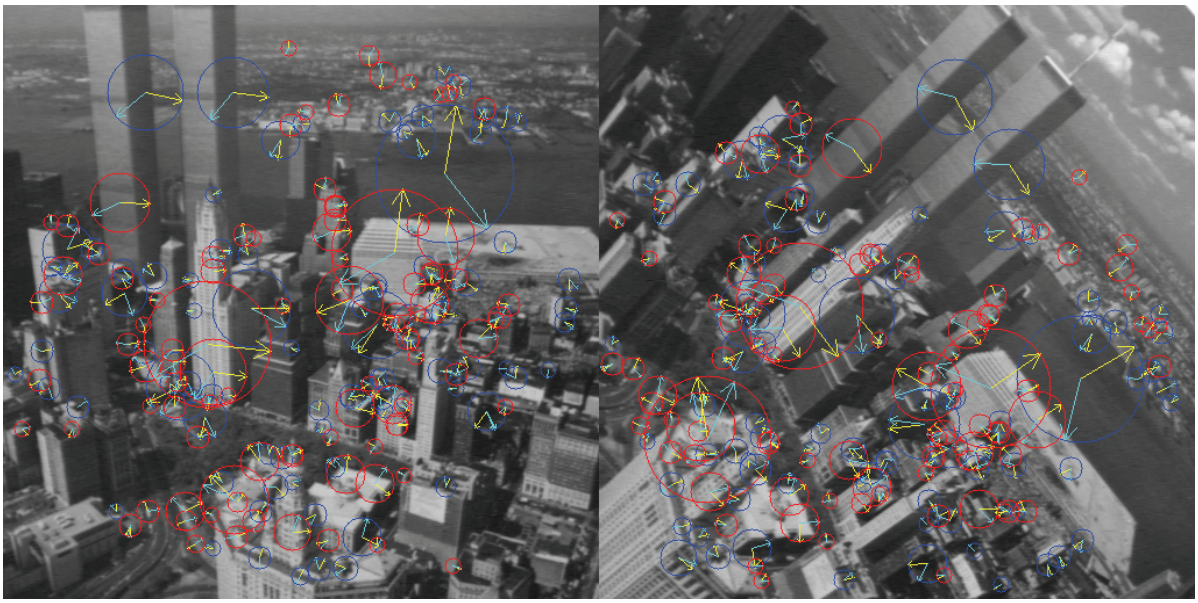


Figure 5.8: Matching result between two images of the same scene imaged from two different viewpoints.

5.4.4. Experimental Results

The proposed method for speeding up feature matching based on split and extended SIFT features was tested using both a standard image dataset, and real world stereo images.

(a) (b)

(c) (d)

Figure 5.9: Some of the standard dataset images of scenes captured under different conditions: (a) viewpoint, (b) light changes, (c) zoom, (d) rotation.

The used image dataset [65] consists of about 500 images of 34 different scenes. Each scene is represented with a number of images taken under different photometric and geometric conditions. Some examples of the images used in the experiments, whose results are presented here, are given in Figure 5.9.

Stereo images were grabbed by the stereo camera system of the rehabilitation robotic system FRIEND (Functional Robot arm with frIENdly interface for Disabled people) [9]. FRIEND is intended to support the user in daily life activities which demand object manipulation such as serving a drink and preparing and serving a meal. The crucial for autonomous object manipulation is precise 3D object localization. The key factor for reliable 3D reconstruction of object points is correct matching of correspondence points in stereo images. Hence, stereo robot vision is a typical application where fast and reliable feature matching is of utmost interest. Some examples of stereo images showing FRIEND environment in "serving a drink" robot working scenario are given in Figure 5.10.

Figure 5.10: Stereo images from a real-world robotic application used in the experiments.

In order to evaluate the effectiveness of the proposed method, its performance was compared with the performances of two algorithms for ANN (hierarchical k-means tree and randomized kd-trees) [59].

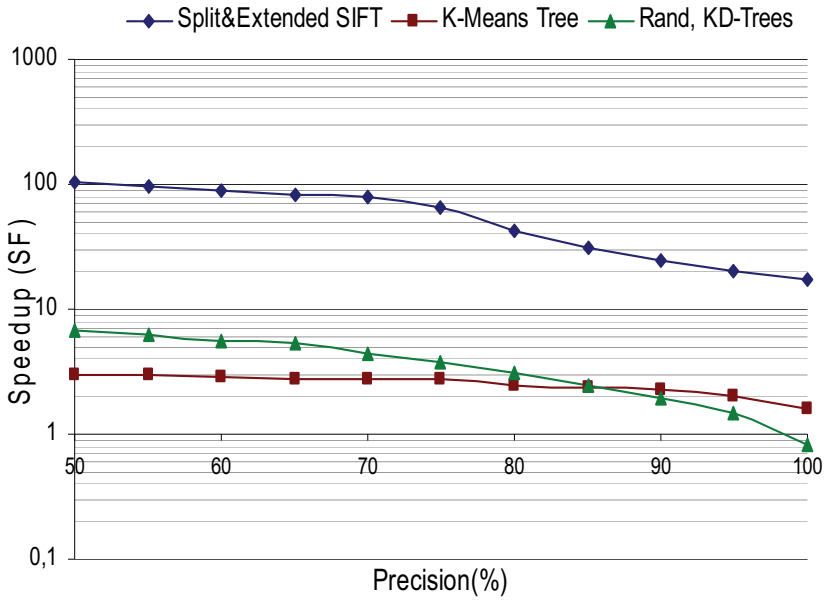


Figure 5.11: Trade-off between matching speedup and matching precision for real stereo image matching.

Comparisons were performed using the Fast Library for Approximate Nearest Neighbors (FLANN) [66], which is a library for performing fast approximate nearest neighbor searching in high dimensional spaces. For all experiments, the matching process is carried out under different precision degrees making trade off between matching speedup and matching accuracy.

The precision degree is defined as the ratio between the number of correct matches returned using the considered algorithm and the number of correct matches returned using exhaustive search, whereas the speedup factor is defined as the ratio between the exhaustive matching time and the matching time for the corresponding method.

For both ANN algorithms, hierarchical k-means trees and randomized kd-trees, the precision is adjusted by the number of nodes to be examined, whereas for the proposed "Split and Extended SIFT" method, the precision is determined by adjusting the width of the range of correct matches w_{cor} (explained in Section 5.4.1). The correct matches are determined using the nearest neighbor distance ratio (NNDR) matching strategy [4] with distance ratio equal to 0.6, followed by RANSAC algorithm [45] to keep only inliers.

Two experiments were run to evaluate proposed method, on real stereo images and on the images of the dataset [65]. In the first experiment, SIFT features are extracted from 200 stereo images. Each two corresponding images are matched using all three considered algorithms under different degrees of precision. The experimental results are shown in Figure 5.11.

As can be seen from Figure 5.11, the performance of the proposed method outperforms both ANN algorithms for all precisions. For precision around 99% level, the proposed method provides a speedup factor of about 20. For the lower precision degree speedup factor is much higher.

As evident from Figure 5.11 by using proposed "Split and extended SIFT" the speedup factor relative to exhaustive search can be increased to 80 times while still returning 70% of the correct matches.

The second experiment was carried out on the images of the dataset [65]. As said before, this dataset consists of about 500 images of various contents. These images represent images of 34 different scenes taken under different conditions such as rotation, zoom, light and viewpoint changes.

For the performed experiments the images of dataset are grouped according to these different conditions into viewpoint, zoom, rotation and light group. For each group, SIFT features are extracted from each image and pairs of two corresponding images are matched using hierarchical k-means tree, randomized kd-trees and proposed "Split and Extended SIFT", with different degrees of precision. The experimental results are shown in Figure 5.12.

As evident from Figure 5.12, proposed "Split and Extended SIFT" outperforms the both other considered ANN algorithms in speeding up of features matching for all precision degrees.

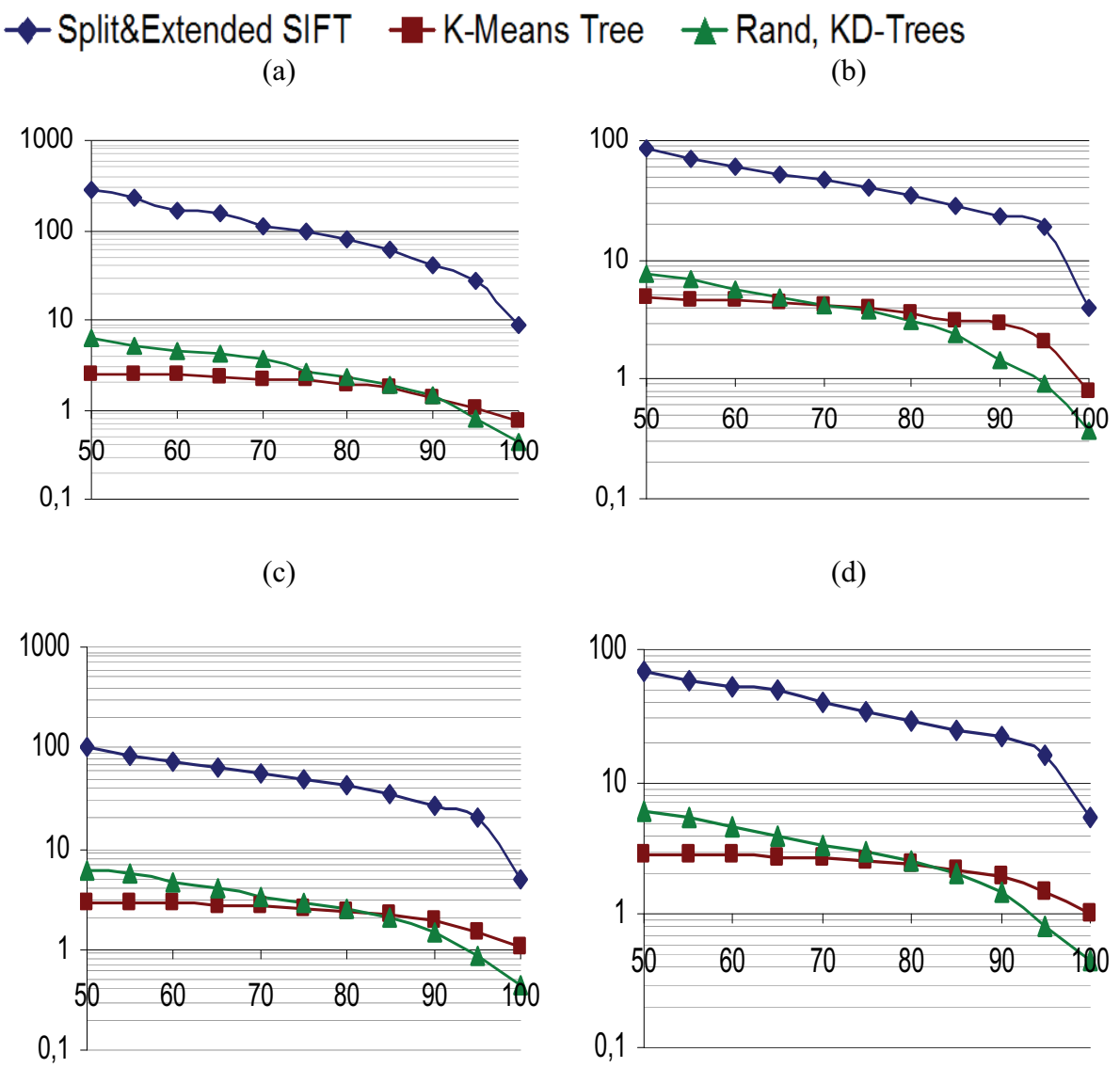


Figure 5.12: Trade-off between matching speedup (SF) and matching precision for image groups (a) light, (b) viewpoint, (c) rotation, (d) zoom changes.

5.5. Very Fast SIFT Feature

Generally, if a scene is captured by two cameras or by one camera but from two different viewpoints, the corresponding points in two resulted images will have different image coordinates, different scales, and different orientations. Nevertheless, they must have almost similar descriptors which are used to match the images using a similarity measure [4,67,68]. The high dimensionality of descriptor makes the feature matching very time-consuming.

In order to speed up the features matching, it is assumed that 4 pairwise independent angles can be assigned to each feature. These angles are invariant to viewing geometry and illumination changes. When these angles are used for feature matching together with SIFT-D, we can avoid the comparison of a great portion of features that can not be matched in any way. This leads to a significant speed up of the matching step as will be shown below.

5.5.1. SIFT Descriptor Based Feature Angles

In Section 5.4, a speeding up of SIFT feature matching by 18 times compared to exhaustive search was achieved by extending SIFT feature with one uniformly-distributed angle computed from the orientation histogram (OH) and by splitting features into Maxima and Minima SIFT features. In this Section the attempts to extend SIFT feature by few angles computed from SIFT descriptor (SIFT-D). As described in Chapter 4, for computation of SIFT-D, the interest region around key-point is subdivided in sub-regions in a rectangular grid. From each sub-region a sub-orientation histogram (SOH) is built.

Theoretically, it is possible to extend a SIFT feature by a number of angles equal to the number of SOHs as these angles are to be calculated from SOHs. In case of 4x4 grid, the number of angles is then 16. However, to reach the very high speed of SIFT matching, these angles should be components of a multivariate random variable that is uniformly distributed in the 16-dimensional space [-180°, 180°][16].

In order to meet this requirement, the following two conditions must be verified [69]:

- Each angle has to be uniformly distributed in [-180°, 180°] (equally likely condition).
- The angles have to be pair-wise independent (pair-wise independence condition).

In this section, the goal is to find a number of angles that are invariant to geometrical and photometrical transformations and that meet the above mentioned conditions. First, the angles between the orientations corresponding to the vector sum of all bins of each SOH and the horizontal orientation are suggested as the SIFT feature angles. Figure 5.13.b presents geometrically the vector sum of a SOH.

Mathematically, the proposed angles $\{\theta_{ij}; i, j = 1,..,4\}$ are calculated according to the following equation:

$$\theta_{ij} = \arctan\left(\frac{\sum_{k=0}^{7} mag_{ij}(k)\cdot\sin\left(ori_{ij}(k)\right)}{\sum_{k=0}^{7} mag_{ij}(k)\cdot\cos\left(ori_{ij}(k)\right)}\right) \tag{5.33}$$

where $mag_{ij}(k)$ and $ori_{ij}(k)$ are the amplitude and the angle of the k^{th} bin of the ij^{th} histogram respectively.

Now, these angles must be examined, whether they meet the equally likely and pair-wise conditions.

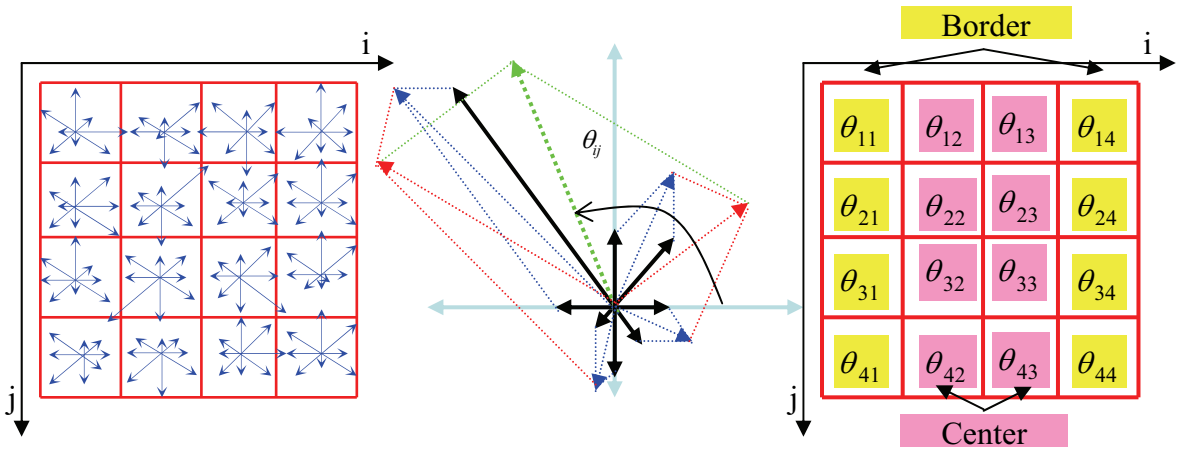
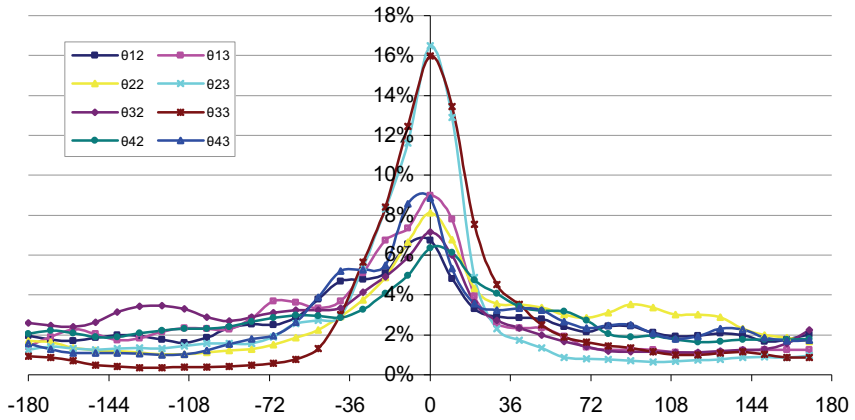


Figure 5.13: (a) SOHs ,(b):Vector sum of the bins of a SOH, (c) angles computed from SOHs

5.5.1.1 Equally Likely Condition

To examine whether the angles $\{\theta_{ij}; i, j = 1,4\}$ meet the equally likely condition, they are considered as random variables $\{\Theta_{ij}; i, j = 1,4\}$.

(a) PDFs of center SIFT descriptor angles.



(b) PDFs of border SIFT descriptor angles.

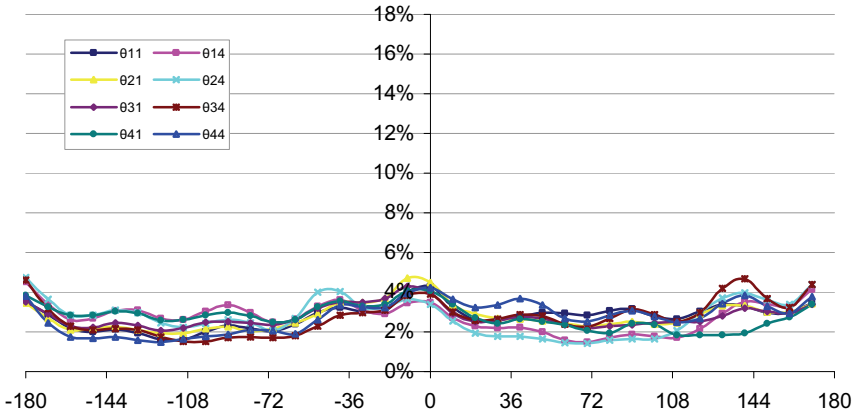


Figure 5.14: The PDFs of angles estimated from 106 SIFT features extracted from 700 images.

The probability density function (PDF) of each angle is estimated from 10^6 SIFT features extracted from 500 benchmark images [65] and 200 stereo images from a real-world robotic application.

The PDFs of Θ_{ij} was computed by dividing the angle space $[-180°,180°)$ into 36 sub-ranges, where each sub-range covers 10°, and by counting the numbers of SIFT features whose angles θ_{ij} belong to each sub-range.

As evident from Figure 5.14, it can be distinguish between two categories of angles based on their PDFs form, the angles computed from the sub-regions that are around the center of SIFT feature (center sub-regions) and the angles computed from the sub-regions that are lying on the SIFT descriptor region boundaries (border sub-regions).

The angles that are computed from the SOHs around the center of SIFT feature (called center angles), have distributions concentrated about 0°, whereas the angles that are calculated from the SOHs of the grid border (called border angles), tend to be equally likely distributed over the angle range.

The reason of this outcome can be interpreted as follows: On the one hand, the SOHs are computed from the interest region (where OH is computed) after its rotation as described in Chapter 4. Therefore the orientations of the maximum bin of each center SOH tend to be equal 0°. On the other hand, for each SOH, the orientation of the maximum bin and the orientation of the vector sum of all bins are strongly dependent since the vector sum includes the maximum bin that has the dominant influence to the vector sum [11]. In the contrary, the border SOHs and the OH do not share the same gradient data, therefore only border angles meet the equally likely condition. Figure 5.13 presents the border and the center angles.

5.5.1.2 Pair-wise Independence Condition

In order to examine whether suggested angles θ_{ij} meet the pair-wise independence condition, it is needed to measure the dependence between each two angles. The most familiar measure of dependence between two quantities is the Pearson product-moment correlation coefficient. It is obtained by dividing the covariance of the two variables by the product of their standard deviations. Assuming that two random variables are given X and Y with expected values μ_x and μ_y and standard deviations σ_x and σ_y then the Pearson product-moment correlation coefficient ρ_{xy} between them is defined as:

$$\rho_{xy} = \frac{E[(X-\mu_x)(Y-\mu_y)]}{\sigma_x \sigma_y} \tag{5.34}$$

where $E[\bullet]$ is the expected value operator.

The correlation coefficients between each two angles α and β are computed using 10^6 SIFT features extracted from the considered test images.

$$\rho_{\alpha\beta} = \frac{10^6 \cdot \sum_{i=1}^{10^6} \left((\alpha_i - \mu_\alpha)(\beta_i - \mu_\beta)\right)}{\left(\sqrt{\sum_{i=1}^{10^6} (\alpha_i - \mu_\alpha)^2} \cdot \sqrt{\sum_{i=1}^{10^6} (\beta_i - \mu_\beta)^2}\right)} \tag{5.35}$$

The estimated correlation coefficients are explained in Figure 5.15 As evident from Figure 5.15, angles that are computed from contiguous SOHs, are highly correlated, whereas there is no or very weak correlations between two angles that are computed from non-contiguous SOHs. The reason of this outcome is caused by the tri-linear interpolation that distributes the gradient samples over contiguous SOHs. In other words, each gradient sample is added to each SOH weighted by 1-d, where d is the distance of the sample from the center of the corresponding sub-region [4]. Hence from the 16 angles at most 4 angles can meet the pair-wise independence condition.

Therefore, only four angles can be pair-wise independent and only border angles can meet the equally likely condition, hence the best choice are the corner angles: $\phi_1 = \theta_{11}$, $\phi_2 = \theta_{14}$, $\phi_3 = \theta_{41}$, and $\phi_4 = \theta_{44}$.

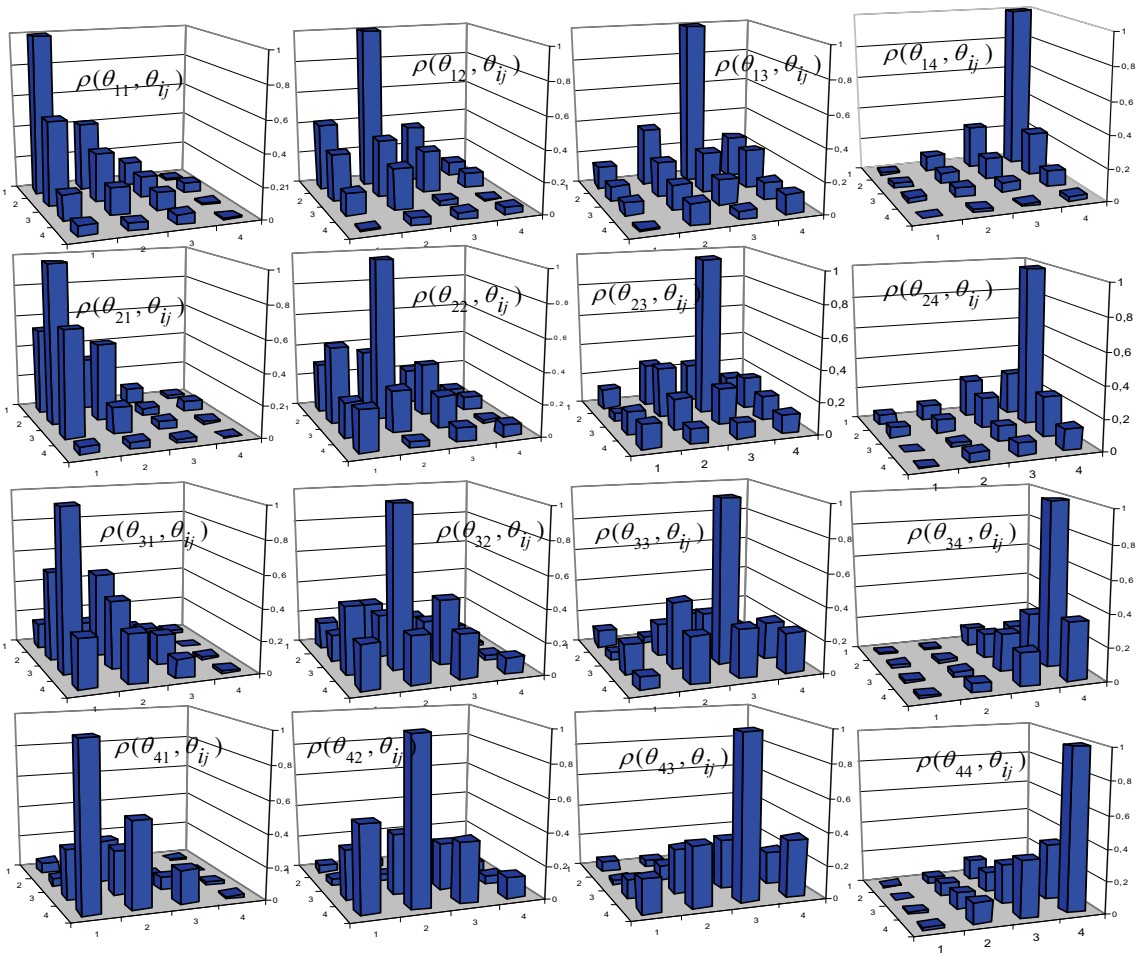


Figure 5.15: The correlation coefficients between angles of SIFT features. For example the top left diagram presents correlation coefficients between θ_{11} and all θ_{ij}. The x and y axes present indices i and j

respectively while z axis present correlation factor.

As evident from Figures 5.14 and 5.15, each two corner angles are independent from each other and uniformly-distributed in the angle range Therefore, the angles $\{\phi_i : i = 1,4\}$ meet both conditions, equally likely and pair-wise independence conditions, hence they can be considered as new attributes of the SIFT feature, that are exploited to accelerate SIFT feature matching.

$$F(x, y, \sigma, \theta_{\max}, V) \rightarrow F(x, y, \sigma, \theta_{\max}, V, \phi_1, \phi_2, \phi_3, \phi_4) \tag{5.36}$$

5.5.2. Very Fast SIFT Features Matching

Feature matching process is the most computationally expensive part of many computer vision algorithms. In this Section new idea is proposed to accelerate the matching process by comparison only features that share the same corresponding angles which may lead to correct matches.

Assuming that two sets of extended SIFT features $R = \{F_i^r; i = 1,2,...r\}$ and $L = \{F_j^l; j = 1,2,...l\}$, containing respectively r and l features, are given,. The number of possible $M_{ij}(F_i^r, F_j^l)$ matches is equal to $r \cdot l$. Among these possible matches a small number of correct matches may exist.

For each possible SIFT match, four different angle differences can be constructed:

$$\begin{aligned} \Delta\phi_{11} &= \phi_1^r - \phi_1^l \\ \Delta\phi_{22} &= \phi_2^r - \phi_2^l \\ \Delta\phi_{33} &= \phi_3^r - \phi_3^l \\ \Delta\phi_{44} &= \phi_4^r - \phi_4^l \end{aligned} \tag{5.37}$$

Considering the angle differences $\{\Delta\phi_{11}, \Delta\phi_{22}, \Delta\phi_{33}, \Delta\phi_{44}\}$ as random variables $\{\Delta\Phi_{11}, \Delta\Phi_{22}, \Delta\Phi_{33}, \Delta\Phi_{44}\}$.

The behaviors of these random variables vary differently according to the type of matches (correct and false matches)

For false match, its features are independent, therefore each two corresponding angles are independent, which lead to the fact that the four random variables are uniformly distributed (according to the lemma proven in Section 5.2) and are pair wise independent.

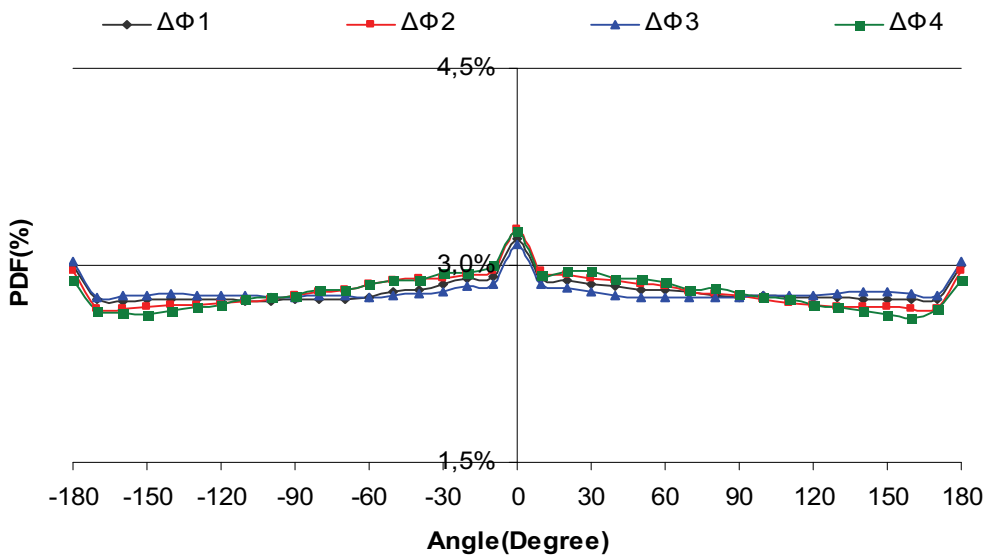
On the other hand, for correct match, each two corresponding angles tend to be equal, since the features of correct matches tend to have same SIFT descriptors. Therefore the four random variables tend to concentrate in narrow range around 0°.

The PDFs of $\Delta\Phi_{ij}$ for both correct and false matches are measured in experiments over considered 700 test images. The estimated PDFs are shown in Figure 5.16.

It can be seen from Figure 5.16 that for each angle separately about 99 % of correct and only 20% of false matches belong to the range $[-36°,36]^4$. Because the possible matches are

uniformly distrusted in the 4 dimensional angle space $[-180°,180]^4$ then the portion of possible matches in the range $[-36°,36]^4$ is equal to $\left(\frac{72}{360}\right)^4 \cdot 100\%$. Therefore, in order to find correct matches it is needed to treat only 0,16% of possible matches which can speed up the feature matching significantly.

(a): PDFs of false matches



(b): PDFs of correct matches

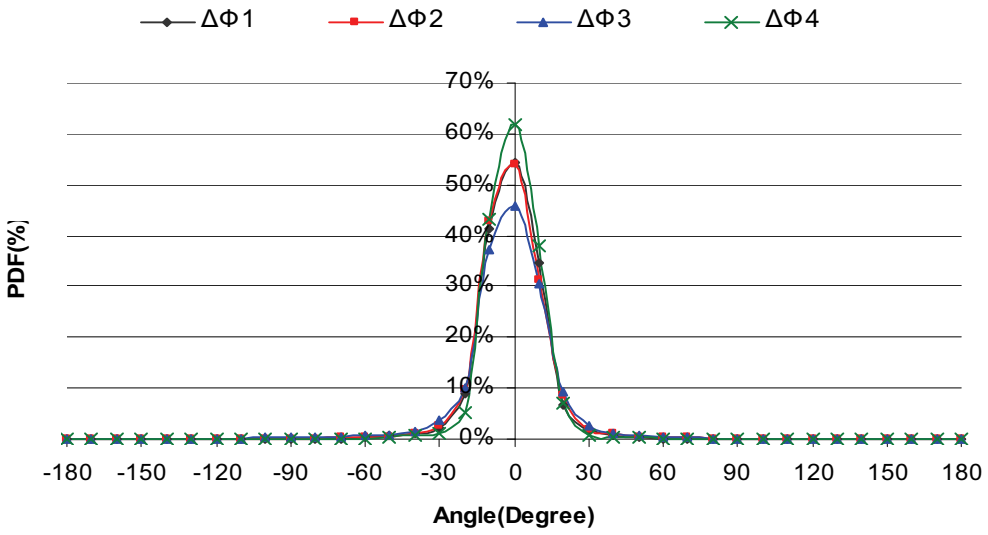


Figure 5.16: The experimental PDFs of the angle difference $\Delta\Phi_{ij}$ for the possible (a) and the correct matches (b).

To exploit this outcome, SIFT features are hashed into 4 dimensional table based on their angles. The SIFT features of each cell are compared only with the features of some cells, so that the correspondences must have absolute differences of angles less than a pre-set

threshold. Here a threshold of 36° is selected because almost all correct matches have angle differences in the range [-36°,36°] as illustrated in Figure 5.16b.

Consider that one of the sets of features R or L (for example R) is hashed into b^4 buckets, so that the $ijfg^{th}$ buckets S_{ijfg} contains only the SIFT features that meet the following conditions:

$$\left.\begin{array}{l}\phi_1 \in \left[-180° + (i-1)\cdot 360°/b, -180° + i\cdot 360°/b\right) \wedge \\ \phi_2 \in \left[-180° + (j-1)\cdot 360°/b, -180° + j\cdot 360°/b\right) \wedge \\ \phi_3 \in \left[-180° + (f-1)\cdot 360°/b, -180° + f\cdot 360°/b\right) \wedge \\ \phi_4 \in \left[-180° + (g-1)\cdot 360°/b, -180° + g\cdot 360°/b\right)\end{array}\right\} \Leftrightarrow S_{ijfg} \subset F(\phi_1,\phi_2,\phi_3,\phi_4) \quad (5.38)$$

The number of features of the set R can be expressed as:

$$r = \sum_{i=1}^{b}\sum_{j=1}^{b}\sum_{f=1}^{b}\sum_{g=1}^{b} r_{ijfg} \quad (5.39)$$

Because of the evenly distribution of feature angles over the range of their angles [-180°,180°] as shown in Figure 5.16, the features are almost equally divided into b^4 buckets. Therefore, it can be asserted that the feature numbers of the buckets are almost equal to each other.

$$\forall i,j,f,g \in \{1,2,...b\} : r_{ijfg} \cong r/b^4 \quad (5.40)$$

To exclude matching of features that have angle differences outside the range $\left[-a°, a°\right]$, each bucket is matched to its corresponding one and to n neighboring buckets to the left and to the right side. In this case the matching time is proportional to the following term:

$$T_{extended} = l \cdot \sum_{o=i-n}^{i+n}\sum_{p=j-n}^{j+n}\sum_{s=f-n}^{f+n}\sum_{t=g-n}^{g+n} r_{opst} = \frac{l\cdot r}{b^4}\cdot \sum_{o=i-n}^{i+n}\sum_{p=j-n}^{j+n}\sum_{s=f-n}^{f+n}\sum_{t=g-n}^{g+n}(1) = \frac{(2n+1)^4\cdot r\cdot l}{b^4} \quad (5.41)$$

Therefore, the achieved speedup factor with respect to exhaustive search is equal to:

$$SF_{extended} = \left(\frac{b}{2n+1}\right)^4 \quad (5.42)$$

The relation between n, a and b is as follows:

$$(2n+1)\cdot\frac{360°}{b} = 2a \Rightarrow n = \left\lceil \left(\left(\frac{2ab}{360°} - 1 \right) \Big/ 2 \right) \right\rceil \quad (5.43)$$

where $\lceil x \rceil$ represents the first integer value larger than or equal to x.

Substituting of (5.42) into (5.41) yields:

$$SF_{extended} = \left(\frac{360}{2a}\right)^4 \tag{5.44}$$

The result (5.43) means that if it is aimed to exclude matching of features that have angle differences outside the range [-36°,36°], then the matching step is accelerated by a factor of 625. When this modification of original SIFT feature matching is combined with the split SIFT features matching, the obtained speedup factor is 1250 without losing a notable portion of correct matches. This is illustrated with the experimental results presented in the next section.

5.5.3. Experimental Results

The proposed method Very Fast SIFT matching (VF-SIFT) was tested using a standard image dataset [65] and real-world stereo images. The used image dataset consists of about 500 images of 34 different scenes (some examples are shown in Figure 5.9). Real-world stereo images was captured using robotic vision system (A Bumblebee 2 stereo camera with the resolution of. 1024X768 pixels), some examples are shown in Figure 5.10.

In order to evaluate the effectiveness of the proposed method, its performance was compared with the performances of two algorithms for ANN (Hierarchical K-Means Tree (HKMT) and Randomized KD-Trees (RKDTs)) [59]. Comparisons were performed using the Fast Library for Approximate Nearest Neighbors (FLANN) [66].

For all algorithms, the matching process is run under different precision degrees making trade off between matching speedup and matching accuracy. The precision degree is defined as the ratio between the number of correct matches returned using the considered algorithm and using the exhaustive search, whereas the speedup factor is defined as the ratio between the exhaustive matching time and the matching time for the corresponding algorithm.

For both ANN algorithms, the precision is adjusted by the number of nodes to be examined [66], whereas for the proposed VF-SIFT method, the precision is determined by adjusting the width of the range of correct matches w_{corr}.

To evaluate the proposed method two experiments were run. In the first experiment, image to image matching was studied. SIFT features were extracted from 100 stereo image pairs and then each two corresponding images were matched using HKMT, RKDTs and VF-SIFT, under different degrees of precision. The experimental results are shown in Figure 5.17a.

The second experiment was carried out on the images of the dataset [65] to study matching image against a database of images. SIFT features extracted from 10 query images are matched against database of 10^5 SIFT features using all three considered algorithms, with different degrees of precision. The experimental results are shown in Figure 5.17b.

As can be seen from Figure 5.17, VF-SIFT extremely outperforms the two other considered algorithms in speeding up of feature matching for all precision degrees. For precision around 95%, VF-SIFT gets a speedup factor of about 1250. For the lower precision degrees speedup factor is much higher.

Through comparison between Figures 5.17a and 5.17b, it can be seen that the proposed method performs similarly for both cases of image matching (image to image and image

against database of images), whereas ANN algorithms are more suitable for matching image against database of images [66].

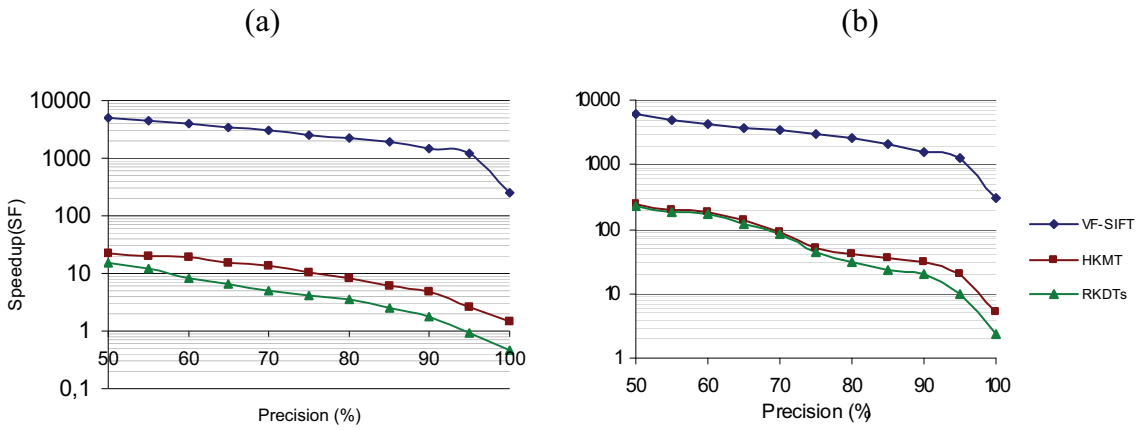


Figure 5.17: Trade-off between matching speedup (SF) and matching precision.

Matching result with illumination changes

Matching result with rotation changes

Figure 5.18: Correct SIFT feature correspondences between two images of the same scene captured under two different conditions.

Figure 5.18 presents two examples of image matching under rotation and illumination changes. It is easy to educe that the correspondence SIFT features are always from the same type (maxima or Minima) and have almost the same corresponding angles.

5.6. Conclusion

In this Chapter, a new method for fast SIFT feature matching is proposed. The idea behind is to extend a SIFT feature by 4 pair-wise independent angles, which are invariant to rotation, scale and illumination changes and uniformly-distributed in the angle range. During extraction phase, SIFT features are classified based on their angles into different clusters. Thus in matching phase, only SIFT features that belong to clusters where correct matches may be expected are compared. The proposed method was tested on real-world stereo images from a robotic application and standard dataset images. The proposed method was compared with two algorithms for ANN searching, hierarchical k-means and randomized kd-trees. The presented experimental results show that the performance of the proposed method outperforms two other considered algorithms. Also, the presented results show that the feature matching can be speeded up by 1250 times with respect to exhaustive search without losing a noticeable portion of correct matches.

6. Robust SIFT Feature Matching

6.1. Introduction

The matching of images in order to establish a measure of their similarity is a key problem in many computer vision tasks. Robot localization and navigation, object recognition, building panoramas and image registration represent just a small sample among a large number of possible applications. In this paper, the emphasis is on object recognition.

In general the existing object recognition algorithms can be classified into two categories: global and local features based algorithms. Global features based algorithms aim at recognizing an object as a whole. To achieve this, after the acquisition, the test object image is sequentially pre-processed and segmented. Then, the global features are extracted and finally statistical features classification techniques are used. This class of algorithm is particularly suitable for recognition of homogeneous (textureless) objects, which can be easily segmented from the image background. Features such as Hu moments [35] or the eigenvectors of the covariance matrix of the segmented object [70] can be used as global features. Global features based algorithms are simple and fast, but there are limitations in the reliability of object recognition under changes in illumination and object pose. In contrast to this, local features based algorithms are more suitable for textured objects and are more robust with respect to variations in pose and illumination. In [71] the advantages of local over global features are demonstrated.

Local features based algorithms focus mainly on the so-called key-points. In this context, the general scheme for object recognition usually involves three important stages: The first one is the extraction of salient feature points (for example corners) from both test and model object images. The second stage is the construction of regions around the salient points using mechanisms that aim to keep the regions characteristics insensitive to viewpoint and illumination changes. The final stage is the matching between test and model images based on extracted features.

The development of image matching by using a set of local key-points can be traced back to the work of Moravec [72]. He defined the concept of "points of interest" as being distinct regions in images that can be used to find matching regions in consecutive image frames. The Moravec operator was further developed by C. Harris and M. Stephens [23] who made it more repeatable under small image variations and near edges. Schmid and Mohr [73] used Harris corners to show that invariant local features matching could be extended to the general image recognition problem. They used a rotationally invariant descriptor for the local image regions in order to allow feature matching under arbitrary orientation variations. Although it is rotational invariant, the Harris corner detector is however very sensitive to changes in image scale so it does not provide a good basis for matching images of different sizes. Lowe [4, 67, 68] overcome such problems by detecting the points of interest over the image and its scales through the location of the local Extrema in a pyramidal Difference of Gaussians (DOG). The Lowe's descriptor, which is based on selecting stable features in the scale space, is named the Scale Invariant Feature Transform (SIFT). Mikolajczyk and Schmid [8] experimentally compared the performances of several currently used local descriptors and they found that the SIFT descriptors to be the most effective, as they yielded the best

matching results. SIFT improving techniques developed recently targeted minimization of the computational time [5][6][7][74], while limited research aiming at improving the accuracy has been done. The work presented in this paper demonstrates increased matching process performance robustness with no additional time costs. Special cases, similar scaled features, consume even less time.

The high effectiveness of the SIFT descriptor is the motivation to use it for object recognition in service robotics applications [75]. Through the performed experiments it was found that SIFT key-points features are highly distinctive and invariant to image scale and rotation providing correct matching in images subject to noise, viewpoint and illumination changes. However, it was also found that sometimes the number of correct matches is insufficient for object recognition, particularly when the target object, or part of it, appears very small in the test image with respect to its appearance in model image. In this chapter, a new strategy to enhance the number of correct matches is proposed. The main idea is to determine the scale factor of the target object in the test image using a suitable mechanism and to perform the matching process under the constraint introduced by the scale factor, as described in Section 6.2.1.

6.2. Improved SIFT Features Matching

From the SIFT algorithm description given in Chapter 4 it is evident that in general, the SIFT-algorithm can be understood as a local image operator which takes an input image and transforms it into a collection of local features. To use the SIFT operator for object recognition purposes, it is applied on two object images, a model and a test image, as shown in Figure 6.1 for the case of a food package. As shown, the model object image is an image of the object alone taken in predefined conditions, while the test image is an image of the object together with its environment.

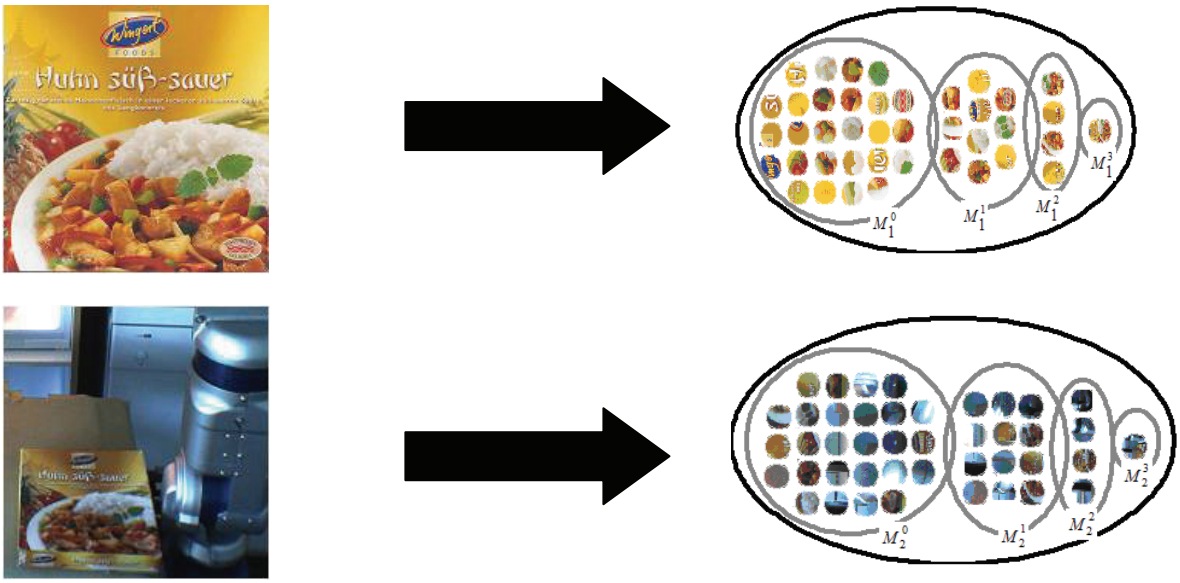


Figure 6.1: Transformation of both model and test image into two collections of SIFT features; division of the features sets into subsets according to the octave of each feature.

To find corresponding features between the two images, which will lead to object recognition, different feature matching approaches can be used. According to the Nearest Neighborhood procedure for each F_1^i feature in the model image feature set the corresponding feature F_2^j must be looked for in the test image feature set. The corresponding feature is one with the smallest Euclidean distance to the feature F_1^i. A pair of corresponding features $\left(F_1^i, F_2^j\right)$ is called a match $M\left(F_1^i, F_2^j\right)$.

To determine whether this match is positive or negative, a threshold can be used.

If the Euclidean distance between the two features F_1^i and F_2^j is below a certain threshold, the match $M\left(F_1^i, F_2^j\right)$ is labelled as positive. Because of the change in the projection of the target object from scene to scene, the global threshold for the distance to the next feature is not useful. Lowe [67] proposed the using of the ratio between the Euclidean distance to the nearest and the second nearest neighbors as a threshold τ.

Under the condition that the object does not contain repeating patterns, one suitable match is expected and the Euclidean distance to the nearest neighbor is significantly smaller than the Euclidean distance to the second nearest neighbor. If no match is correct, all distances have a similar, small difference from each other. A match is selected as positive only if the distance to the nearest neighbor is 0.8 times larger than that from the second nearest one. Among positive and negative matches, correct as well as false matches can be found. Lowe claims [4] that the threshold of 0.8 provides 95% of correct matches as positive and 90% of false matches as negative. The total amount of the correct positive matches must be large enough to provide reliable object recognition.

In the following an improvement to the feature matching robustness of the SIFT algorithm with respect to the number of correct positive matches is presented.

As mentioned above, the target object in the test image is part of a cluttered scene. In a real-world application the appearance of the target object in the test image, its position, scale and orientation, are not known a priori. Assuming that the target object is not deformed, all features of the target image can be considered as being affected with constant scaling and rotational factors. This can be used to optimize the SIFT-feature matching phase where the outliers' rejection stage of the original SIFT-method is integrated into the SIFT-feature matching stage.

6.2.1. Scaling Factor Calculation

As mentioned above, using the SIFT-operator, the two object images (model and test) are transformed into two SIFT-image feature sets. These two feature sets are divided into subsets according to the octaves in which the feature arise. Hence, there is a separate subset for each image octave as shown in Figure 6.1.

To carry out the proposed new strategy of SIFT-features matching, the features subsets obtained are arranged so that a subset of the model image feature set is aligned with an appropriate subset of the test image feature set. The process of alignment of the model image subsets with the test image subsets is indicated with arrows in Figure 6.2. The alignment process is performed through the $n+m-1$ steps, where n and m are the total number of octaves (subsets) corresponding to the model and test image respectively.

For each step all pairs of aligned subsets must have the same ratio ν defined as:

$$\nu = 2^{o_1} / 2^{o_2} \tag{6.1}$$

where o_1 and o_2 are the octaves of the model image subset and the test image subset respectively.

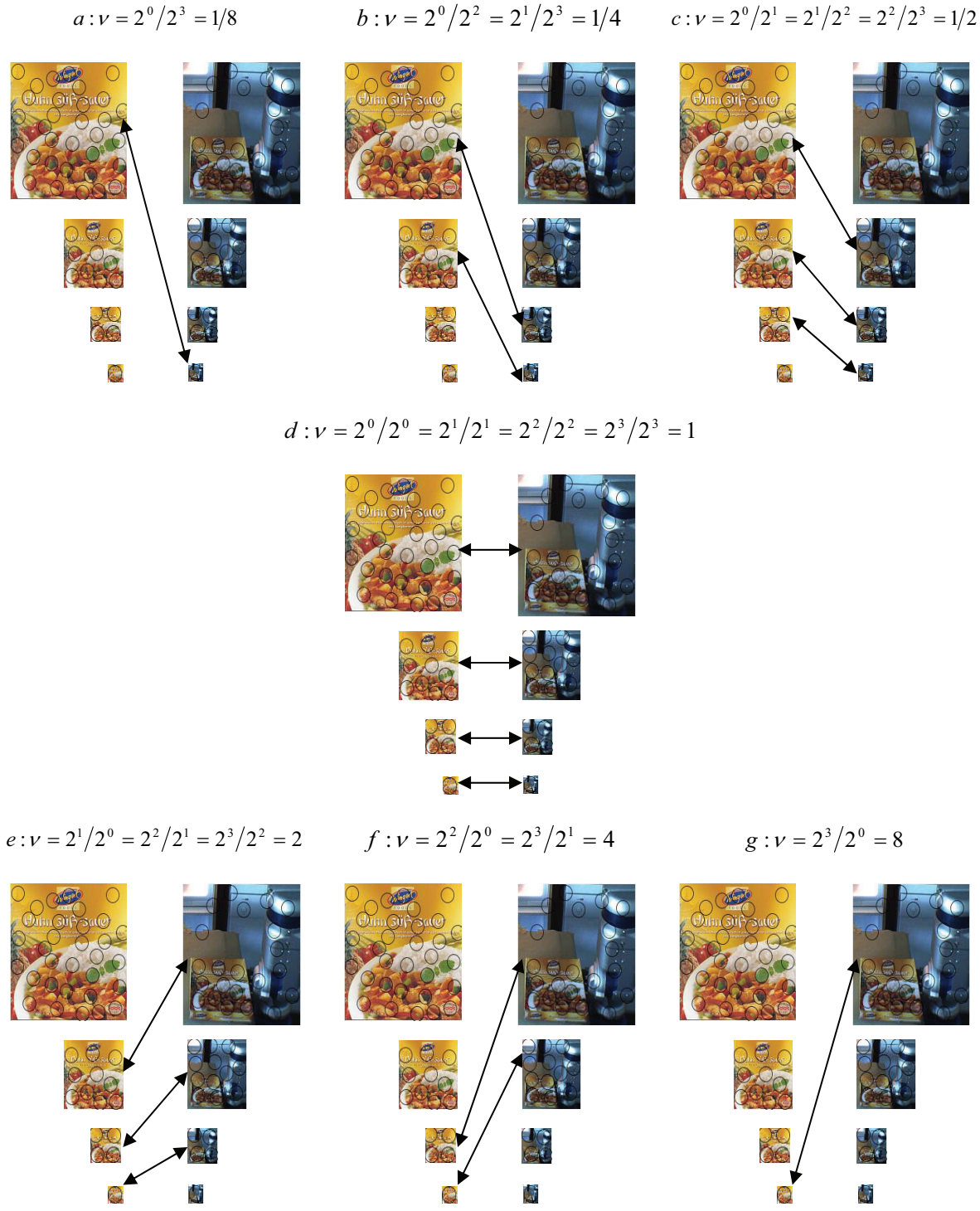


Figure 6.2: Steps of the procedure for scale factor calculation.

For example at the first step a, only SIFT features of model image extracted from the octave $o_1 = 0$ are compared with the SIFT features of the test image extracted from the

octave $o_2 = 3$. In this case we can grantee that all possible matches have the scale ratio $\nu = 2^0/2^3 = 1/8$.

In the step b, only the model SIFT features of the octaves $o_1 = 0$ and $o_1 = 1$ are compared with the test SIFT features of the octaves $o_2 = 2$ and $o_2 = 3$ respectively. In both cases, possible matches have scale ratio of $\nu = 2^0/2^2 = 2^1/2^3 = 1/4$, and so on for the other steps.

At every step, the total number of positive matches is determined for each aligned subsets pair. The total number of positive matches within each step is indexed using the appropriate shift index

$$k = o_1 - o_2 \tag{6.2}$$

Shift index can be negative (Figures 6.2a, 6.2b and 6.2c), positive (6.2e, 6.2f and 6.2g) or equal to zero (Figure 6.2d). The highest number of positive matches achieved determines the optimal shift index k_{opt} and consequently the scale factor:

$$S = 2^{k_{opt}} \tag{6.3}$$

In order to realize the proposed procedure mathematically, a scale ratio histogram (SRH) $F(x)$ is defined as:

$$F(x) = \begin{cases} \sum_{j=0}^{x} \Re\left(M_1^{n-1-x+j}, M_2^{j}\right) & if \quad x < n \\ \sum_{i=0}^{j=n-1} \Re\left(M_1^{x-n-1+j}, M_2^{j}\right) & if \quad n \le x < m \\ \sum_{i=0}^{j=m+n-2-x} \Re\left(M_1^{x-n+j}, M_2^{x-m+1+j}\right) & if \quad x \ge m \end{cases} \tag{6.4}$$

where $\Re\left(M_1^i, M_2^j\right)$ is the number of positive matches between the i^{th} subset of the model image feature set M_1^i and the j^{th} subset of the test image feature set M_2^j, and x is the modified shift index introduced for the sake of simplicity of equation 6.4.

$$x = \text{int}\left(k + \left(\frac{n+m-1}{2}\right)\right) \tag{6.5}$$

The diagram showing the distribution of $F(k)$ over the range of the shift index k for the example shown in Figure 2 is presented in Figure 6.2.

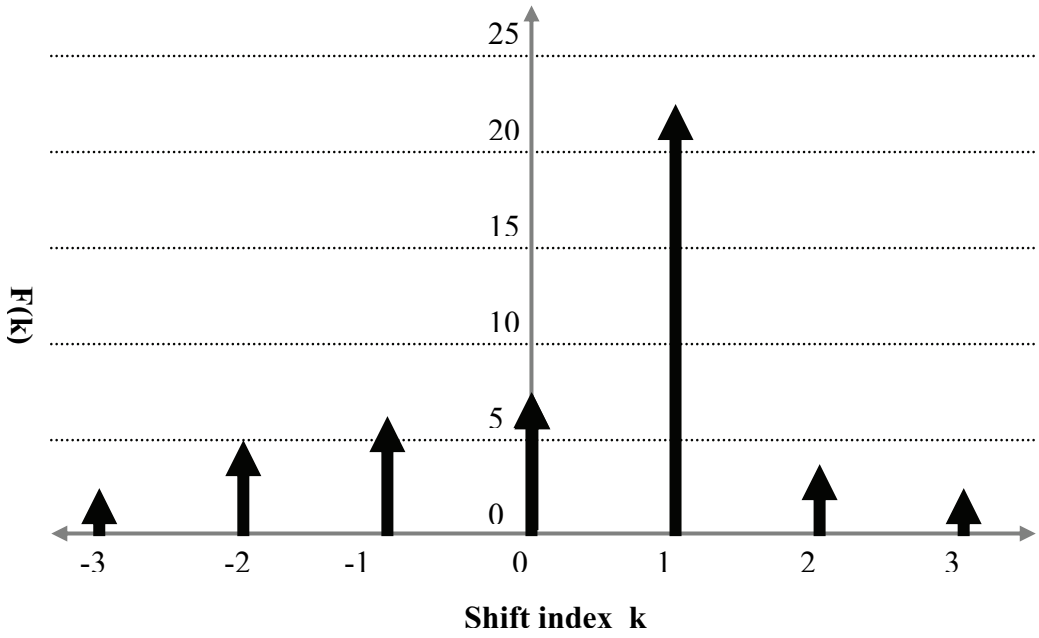


Figure 6.3: The scale ratio histogram $F(k)$.

As evident from Figure 6.3, the scale ratio histogram $F(k)$ reaches its maximum at the shift index:

$$k_{opt} = arc\max(F(k)) = 1 \tag{6.6}$$

which corresponds to the scale factor

$$S = 2^{k_{opt}} = 2 \tag{6.7}$$

The optimal shift index defines a "domain of correct matches". All matches outside this domain, including positive matches, are excluded. The positive matches from the domain of correct matches are used to determine the affine transformation (rotation matrix, and translation vector) between the two feature sets, using RANSAC method [45]. Once the transformation is calculated, every match, either positive or negative, within the domain of correct matches is examined whether it meets the already calculated transformation. If the match fulfils the transformation, it is labelled as a correct, otherwise as a false match.

6.2.2. Retrieval of The Correct Matches

Among all found matches it can happen that a lot of correct matches exceed Lowe's threshold τ.

In order to retrieve these correct matches, the ratio between the Euclidean distance to the nearest and the second nearest feature neighbor must be reduced. This can be done either by reducing the smallest distance $d_1\left(F_1^i, F_2^{j_0}\right)$ or by increasing the next smallest distance $d_2\left(F_1^i, F_2^{j_1}\right)$. In practice, the first alternative is impossible while the enlargement of next smallest distance can be achieved by limiting the search area for both the nearest and next nearest feature to the feature F_1^i within a specified domain. For a better explanation of this

idea, suppose that a feature F_1^i from the model image feature set is correctly assigned to the feature $F_2^{j_0}$ from the test image feature set. Also, suppose that $F_2^{j_1}$ is the second nearest feature to the F_1^i while $F_2^{j_2}$ is the second nearest feature to it when the search is limited only to the octave in which the $F_2^{j_0}$ is found.

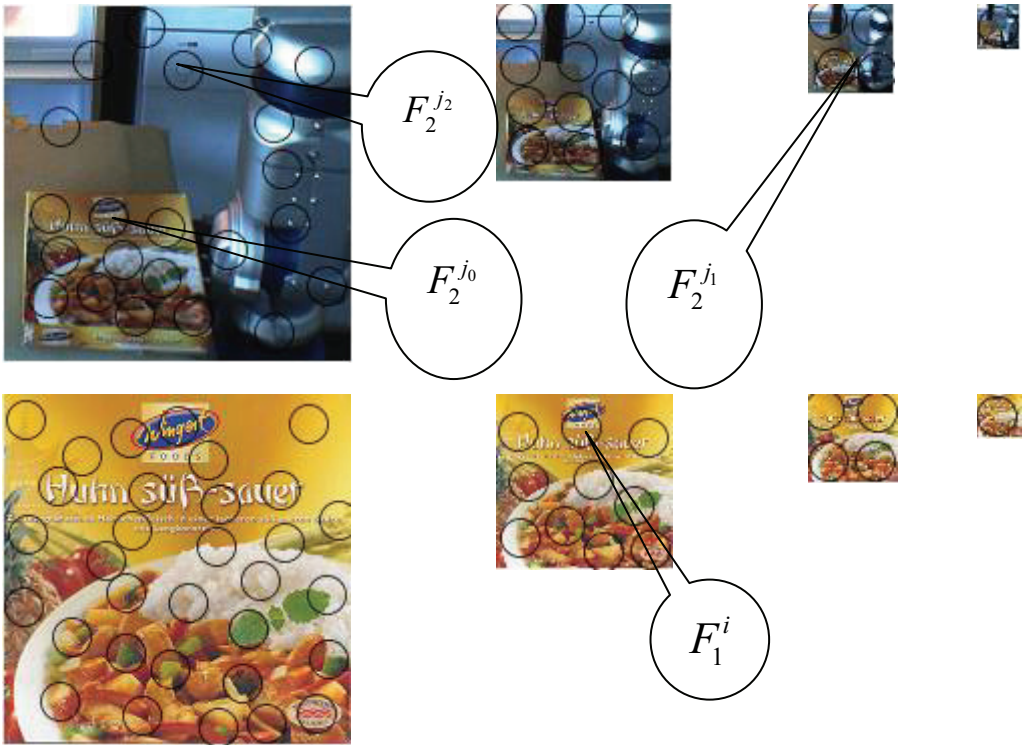


Figure 6.4: Saving the correct matches that may exceed Lowe's threshold.

Since $d_2\left(F_1^i, F_2^{j_1}\right) \leq d_3\left(F_1^i, F_2^{j_2}\right)$ always holds the following is obtained:

$$d_1\left(F_1^i, F_2^{j_0}\right) / d_2\left(F_1^i, F_2^{j_1}\right) \geq d_1\left(F_1^i, F_2^{j_0}\right) / d_3\left(F_1^i, F_2^{j_2}\right) \tag{6.8}$$

Thus, by reducing the search area it is possible to decrease the ratio related to the feature F_1^i and make it less than threshold τ.

In this way the number of correct matches is increased.

6.2.3. Complexity and Cost of Time

An additional result of the research presented in this chapter is consideration of the improvement of the original SIFT algorithm with respect to the processing time.

As first, it can be shown that the original SIFT procedure and the procedure developed in this work complete the matching procedure in the same time.

Assuming that the number of features in the model object image and in the test image is:

$$h = h_0 + h_1 + \ldots\ldots\ldots\ldots + h_{n-1} = \sum_{i=0}^{n-1} h_i$$
$$l = l_0 + l_1 + \ldots\ldots\ldots\ldots\ldots + l_{m-1} = \sum_{j=0}^{m-1} l_j \tag{6.9}$$

where n and m are the total number of octaves corresponding to the model and test image respectively.

Thus, the complexity of original SIFT-matching procedure is proportional to the product

$$P_1 = l \cdot h \tag{6.10}$$

The complexity of the proposed approach, which can be seen from Figure 6.2, is proportional to the following sum of the products:

$$\begin{aligned}
P_2 &= l_{m-1} \cdot h_0 \\
&\quad + l_{m-1} \cdot h_1 + l_{m-2} \cdot h_0 \\
&\vdots \\
&\quad + l_{m-1} \cdot h_{n-1} + l_{m-2} \cdot h_{n-2} + l_{m-3} \cdot h_{n-3} + \ldots\ldots\ldots + l_1 \cdot h_1 + l_0 . h_0 \\
&\qquad + l_{m-2} \cdot h_{n-1} + l_{m-3} \cdot h_{n-2} + \ldots\ldots\ldots + l_1 \cdot h_2 + l_0 . h_1 \\
&\vdots \\
&\qquad + l_1 \cdot h_{n-1} + l_0 \cdot h_{n-2} \\
&\qquad + l_0 \cdot h_{n-1} = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} h_i \cdot l_j
\end{aligned} \tag{6.11}$$

Substituting equation (6.9) in equation (6.11) one obtains:

$$P_2 = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} h_i \cdot l_j = \sum_{i=0}^{n-1} h_i \cdot \sum_{j=0}^{m-1} h_i = l \cdot h \tag{6.12}$$

which is equal to the product P_1 corresponding to the complexity of the original SIFT matching procedure.

The above condition represents the complexity of the proposed matching procedure when no a-priory information about the scaling factor of corresponding features is available, that is when the procedure consists of all $n + m - 1$ steps as explained in Section 6.2.1.

However, in some applications the complexity is reduced. For example, if the two images to be matched are images of stereo camera system with small baseline, all corresponding features should have the same scale. Hence the proposed matching procedure is carried out with only one step corresponding to the shift index $k = 0$.

In this case, the complexity of the proposed procedures is reduced, since it is proportional to the sum of the following products:

$$P_3 = l_0 \cdot h_0 + l_1 \cdot h_1 + ... + l_{n-1} \cdot h_{n-1} \tag{6.13}$$

In order to determine the amount of reduced processing time in comparison to original SIFT procedure, it is assumed that the number of extracted features in the lower octave with respect to the higher octave is decreased 4 times due to the down-sampling by the factor of 2 in both image directions. Hence, it is assumed that:

$$\begin{aligned} l_{i-1} &\approx 4 \cdot l_i \\ h_{i-1} &\approx 4 \cdot h_i \end{aligned} \tag{6.14}$$

Substituting equation (6.14) in both products P_2 and P_3, defined with (6.12) and (6.13) respectively, one obtains:

$$\begin{aligned} P_3 &= l_0 \cdot h_0 + (1/4)^2 \cdot l_0 \cdot h_0 + (1/4)^4 \cdot l_0 \cdot h_0 + ... + (1/4)^{2(n-1)} \cdot l_0 \cdot h_0 \\ P_3 &= l_0 \cdot h_0 \cdot \sum_{i=0}^{n-1} (1/4)^{2i} \cdot \end{aligned} \tag{6.15}$$

and

$$\begin{aligned} P_2 &= l \cdot h = (l_0 + l_1 + ... + l_{n-1}) \cdot (h_0 + h_1 + ... + h_{n-1}) \\ P_2 &= \left(l_0 + (1/4) \cdot l_0 + ... + (1/4)^{n-1} \cdot l_0\right) \cdot \left(h_0 + (1/4) \cdot h_0 + ... + (1/4)^{n-1} \cdot h_0\right) \cdot \\ P_2 &= l_0 \cdot h_0 \cdot \left(\sum_{i=0}^{n-1} (1/4)^i \right)^2 \end{aligned} \tag{6.16}$$

From equations (6.15) and (6.16) the ratio P_2 / P_3 is given as:

$$\frac{P_2}{P_3} = \frac{l_0 \cdot h_0 \cdot \left(\sum_{i=0}^{n-1} (1/4)^i \right)^2}{l_0 \cdot h_0 \cdot \sum_{i=0}^{n-1} (1/4)^{2i}} \approx \frac{\left(\sum_{i=0}^{\infty} (1/4)^i \right)^2}{\sum_{i=0}^{\infty} (1/4)^{2i}} \tag{6.17}$$

It is known that

$$\sum_{i=0}^{\infty} x^i = \frac{1}{1-x} \qquad if \qquad |x| < 1 \tag{6.18}$$

Substituting (6.18), the ratio (6.17) becomes:

$$\frac{P_2}{P_3} \approx \frac{\left(\sum_{i=0}^{\infty}(1/4)^i\right)^2}{\sum_{i=0}^{\infty}(1/4)^{2i}} = \frac{\left(\frac{1}{1-(1/4)}\right)^2}{\left(\frac{1}{1-(1/4)^2}\right)} = \frac{15}{9} = 1{,}67 \tag{6.19}$$

Hence, the matching time cost in the case of matching stereo images is reduced 1.67 times in comparison to the original SIFT method.

6.3. Experimental Results

In this section a performance evaluation of the proposed improvement of the Lowe's SIFT feature matching algorithm is presented. Since the goal is to achieve a trade-off between the increasing the number of correct matches and minimizing the number of false matches for an object image pair consisting of test and model object images, the performance of the proposed method is evaluated using the popular Recall-Precision metric [76].

As mentioned in Section 6.2, two SIFT features F_1^i and F_2^j are matched when the SIFT descriptor of the feature F_2^j has the smallest distance to the descriptor of feature F_1^i among distances corresponding to all other extracted features. If the ratio between the Euclidian distances to the nearest neighbor and to the second nearest neighbor is below a threshold τ, the match is labeled as positive, otherwise as negative..

Among positive and negative labeled matches, correct as well as false matches can be found. Thus there are four different possible combinations through the following confusion matrix:

Table 6.1: The confusion Matrix

	Actual positive	Actual negative
Predicted positive	TP	FP
Predicted negative	FN	TN

During the matching of an image pair the elements of the confusion matrix are counted. The value of τ is varied to obtain the Recall versus 1-Precision curve, with which the result are presented.

Recall and 1- Precision are calculated based on the following definitions [67]:

$$\begin{aligned} \mathrm{Re}\,call &= \frac{TP}{(TP+FN)} \\ 1-\mathrm{Pr}\,ecision &= \frac{FP}{(TP+FN)} \end{aligned} \tag{6.20}$$

The algorithms were tested by matching real images of the scenes from working scenarios of the robotic system FRIEND II containing different target objects to be recognized (bottles, packages, and etc), acquired with the stereo camera system of FRIEND II robot.

Two main types of experiments were run to discuss the difference between the original SIFT and the proposed optimized SIFT matching algorithm. In the first experiment, the model images of two different objects, a bottle of the "mezzo mix" drink and a coffee filters package, were matched with the corresponding test object images using the original and proposed improved SIFT matching algorithm.

The experimental results are illustrated in Figure 6.6. As evident, the appearance of the target objects in the test images is different from their appearance in model images due to different conditions such as illumination during the image acquisition, viewpoint, partial occlusion etc. the advantage of the proposed matching technique over the original SIFT matching technique is evident from Figure 6.6.

Beside the examination of the results illustration in Figure 6.6, performance evaluation can be done by examination of the recall versus 1-precision curve shown in Figure 6.5. The curves are obtained by varying the threshold from 0.5 till 1.0.

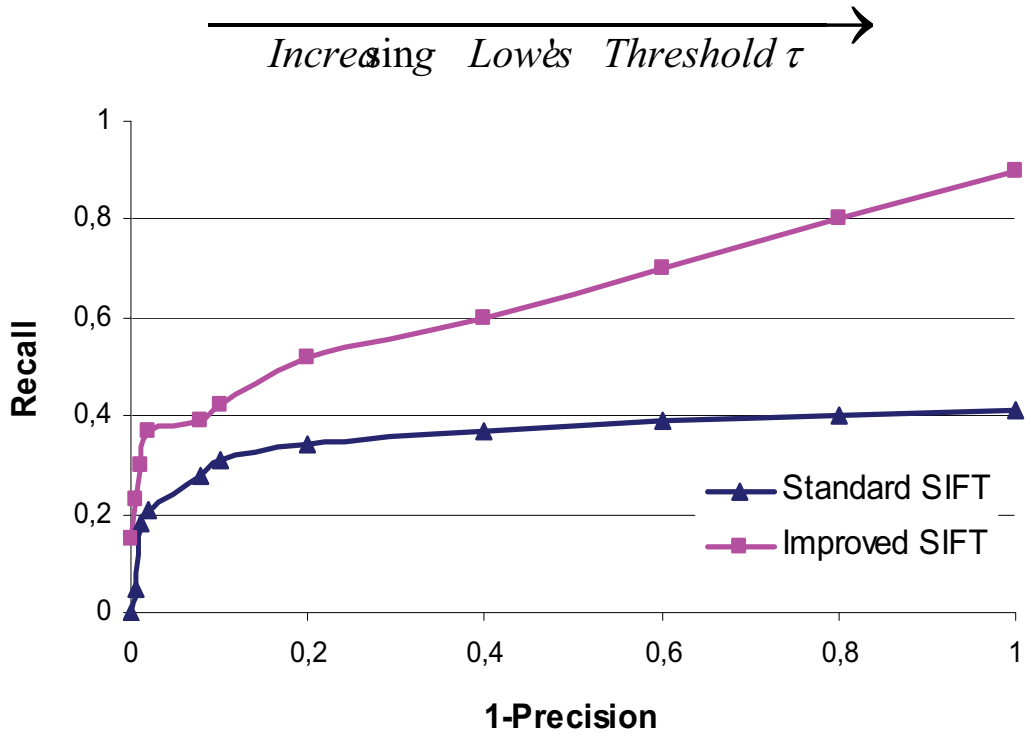


Figure 6.5: Recall versus 1-Precision curves for the original and optimized SIFT matching methods.

In the second experiment images of a scene from the robot FRIEND II environment, captured by the robot stereo camera system, were matched to evaluate the optimizing of the computational matching time of the proposed approach with respect to the original SIFT. The experimental results are given in the Table 6.2. The experimentally obtained ratios of the processing time of original SIFT and processing time of proposed technique slightly differ from the ratio derived in section 6.2 because the assumption assumed the proof does not necessarily hold. The matching process was carried out using a Pentium IV 1GH processor with, images of size 1024X768 pixels.

Table 6.2: Comparison of the stereo images matching time.

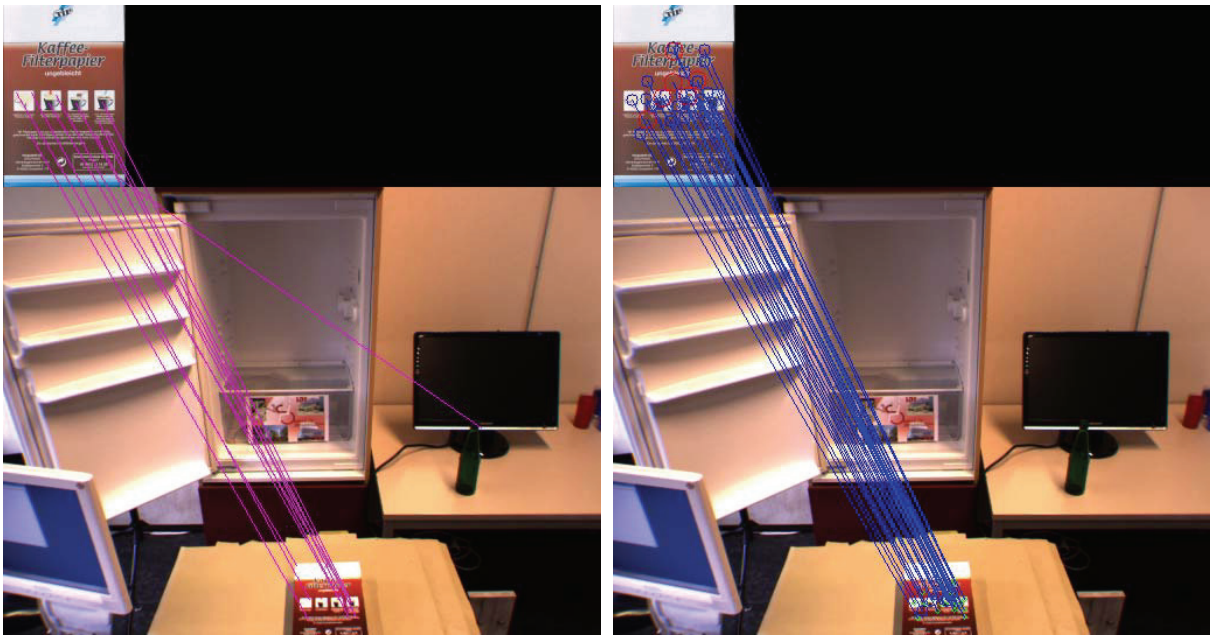
Key-points number in stereo images		Original SIFT matching		Improved SIFT matching	
right	left	Matching time (sec)	Number of inliers	Matching time (sec)	Number of inliers
217	229	0.140	111	0.025	133
777	640	0.790	284	0.230	325
3014	2233	10.760	605	4.950	683
6871	6376	69.810	751	47.790	856

6.4. Conclusions

In this chapter an improvement of the original SIFT-algorithm developed by Lowe was proposed. This improvement corresponds to enhancement of feature matching robustness, so the number of correct SIFT features matches is significantly increased while nearly all outliers are discarded. The idea is based on the determination of the scale factor between images to be matched and limiting the matching process to feature pairs that fit this scale factor. In order to determine the scale factor, the feature sets are divided into subsets according to the octaves in which the feature arise. After that the feature matching is performed in stepwise fashion so that with each step only the SIFT features of the same scale ratio is matched. The step with the highest number of positive matches determines the approximate scale factor between the images being matched. When no pre-information about scale factor are available then both matching procedures, the standard SIFT and the procedure developed in this work complete the matching process in the same time.

The new proposed approach was tested using real images acquired with the stereo camera system of FRIEND II/III robotic system. The experimental results showed that the number of correct matches was increased and, at the same time, the number of outliers was decreased in comparison with the original SIFT algorithm. Compared with the original SIFT algorithm, a 40% reduction in processing time was achieved for the matching of the stereo images, since the scale factor in case of stereo image matching is equal to 1.

Matching result for coffee filter package



Matching result for bottle of the "mezzo mix" drink

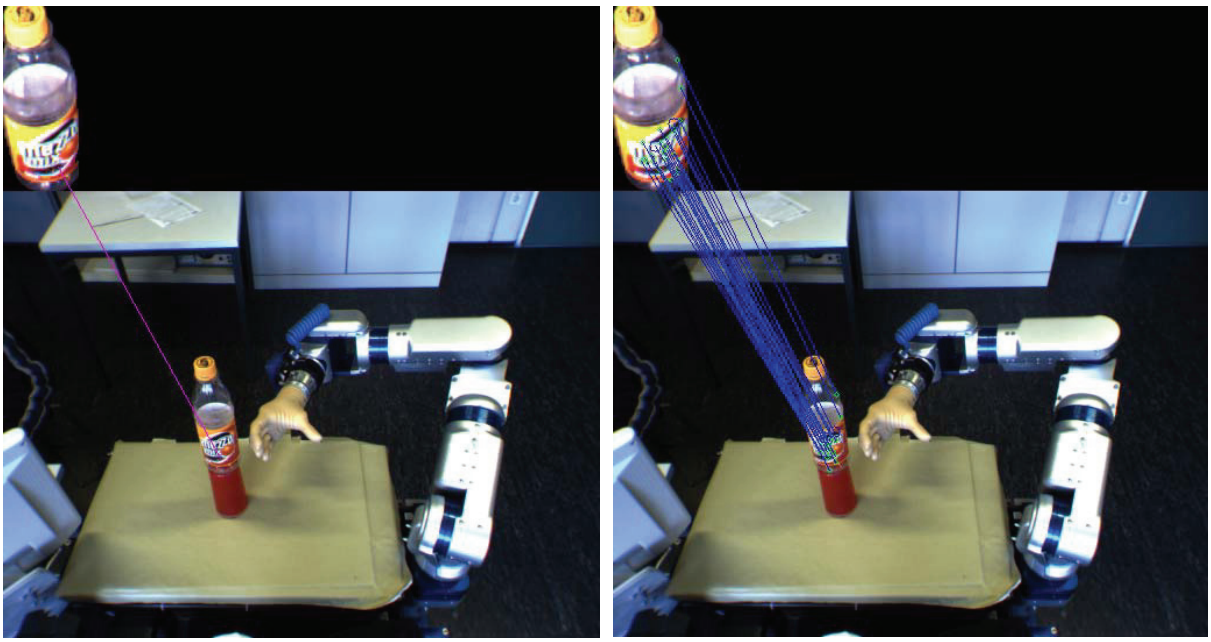


Figure 6.6: (left column) matching result with original SIFT, (right column) matching result with improved SIFT.

7. Fuzzy Based Closed Loop Control System for Object Recognition

7.1. Introduction

One of the most widely researched and an important area in computer vision is object recognition. In general, the object recognition systems can be classified into two major categories: the global and the local feature-based systems. The global feature-based systems aim at recognizing the object in its whole. To this end, the query image is acquired, pre-processed, segmented, and then global features are extracted. Finally, statistical classification techniques are used. This class of algorithms is especially suitable for homogeneous objects, which can be easily segmented. Features such as the Hu moments [85], the eigenvectors of the covariance matrix [86], centroids, perimeters, areas, and colors [87][88] can be used as global features. The global feature-based algorithms are simple and fast, but there are limitations in recognition under illumination and pose changes, Figure 7.1 presents the flow diagram of the global feature-based object recognition systems. Local feature-based systems on the other hand are more suitable for textured objects and more robust with respect to viewpoint and illumination changes.

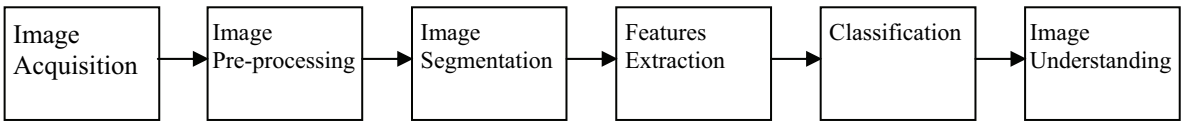


Figure 7.1: Global feature-based object recognition system

The local feature-based systems are based on the idea of representing an object by a collection of local invariant patches. This idea can be traced back to Schmid and Mohr [89][90], where the centers of patches are located at points of interest and are invariant to rotation. Lowe [67][68] developed an efficient object recognition approach based on scale invariant features (SIFT).

Generally, the structure of the local feature-based object recognition system mainly involves four major steps, as shown in Figure 7.2:

- Features detection: Extraction of salient points (typically corners or blob-like shapes), from images to be matched (query and model images).
- Features description: Construction of descriptors from regions around the salient key-point uses mechanisms that aim to keep the characteristics of these regions insensitive to viewpoint, illumination changes and invariant to rotation, scaling and affine transformation.
- Features matching: Computing the correspondence points between the query and the model image based on extracted features. Out of the matched points an affine transformation between query image and model image can be computed using a fitting method (such as Least of Squares or RANSAC method). The matching process is then iteratively refined by removing those correspondence points which do not fit this affine transformation.

- Pose estimation: Estimation of the (x, y, z)-translation components and (α, β, δ)-rotation angles of the object with respect to the camera coordinate system using the correspondence points, the target object geometry and the intrinsic camera parameters.

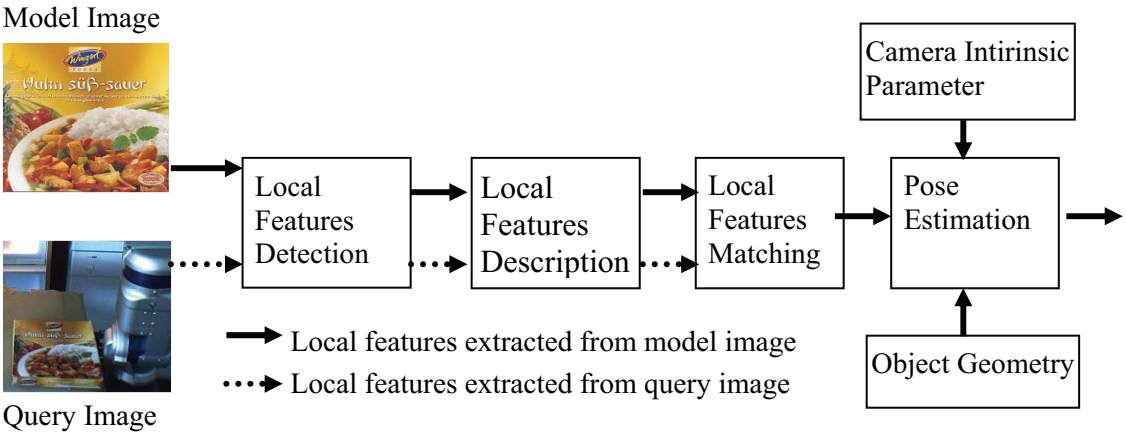


Figure 7.2: Local feature-based object recognition system

It can be easily noticed that both object recognition systems presented above are open-loop, which means that the result of each step depends on the result of the previous one, therefore the errors are accumulated over the entire recognition system and propagated to the final step. Hence the system final result tends to be error prone and unreliable. This problem is usually solved using closed-loop control techniques.

In the literature, there are few publications dealing with the usage of closed loop control strategies for object recognition and image processing. For example, in [80] and [81] reinforcement learning has been used to induce a mapping from input images to corresponding segmentation parameters by using the confidence level of model matching as a reinforcement. In [82] control strategies have been used at low, intermediate, and high levels of analysis for improving on established-single-pass hypothesis generation and verification approaches in object recognition. In [83] and [84] the feedback control of image quality at different levels of image processing chain, aiming at global feature-based object recognition is introduced to improve the image quality for successful image segmentation and feature extraction.

The above-mentioned methods commonly concentrate on the global feature-based object recognition systems through optimizing of the segmentation stage.

In this chapter, we propose a closed-loop control system for object recognition, pose estimation and camera calibration based on SIFT features [4]. Our work concentrates on using the benefits of closed loop structure to increase the invariance to affinity, therefore to increase the quality and the quantity of the matching process and to refine pose estimation, which is essential for service robotics for autonomous object manipulation. The idea is to extract two independent parallel feature streams (Maxima and Minima SIFT features) from both the model and the query image, and then matching between features belong to corresponding streams to estimate two independent affine transformations. The dissimilarity between these transformations is used as a feedback variable that serves to observe and control the matching process. If this variable is more than a certain threshold, one of the transformations is selected using fuzzy controller to warp the model image. The procedure is repeated until the two

transformations become similar or one of them converges to the identity matrix. The system has been verified through experiments on several real-world images. The obtained results are shown in Section 7.3.

7.2. Closed Loop Control System for Object Recognition

A typical local feature-based object recognition system as shown in Figure 7.2 is used to identify and locate an object of interest captured by camera system in a scene .The input of the system is a model image of the object of interest and a query image. The model image is used to examine the presence of the object in the corresponding query image and to estimate its pose with respect to the camera coordinate system. At first, key points are extracted from the model and the query images and described by SIFT descriptor [4][67]. These SIFT features are then provided as input to image matching process. In general, image matching is defined as a process in which the correspondences between subset of points in two images are determined. From correspondences an affine transformation (rotation, scaling, and translation changes) is estimated that maps the two images. Once the correspondence points are established, and the intrinsic camera parameters and the geometry of the object are known, the pose of object can be estimated [92].

The accuracy and the reliability of the estimated pose depend strongly on the outcome of the image matching process. Hence the matching result plays a crucial role in the reliability of the whole system.

The system illustrated in Figure 7.2 totally ignores the effects of mismatches on the performance of the pose estimation method. This problem is similar to that occurs in the open loop systems, which are affected by noise. In control theory, feedback loops have been used to solve these problems. In this chapter, we try to use a similar principle for improving the quality and the quantity of the matching process result, which leads to enhance the efficiency of the object detection and to refine the 3D pose of the target object.

To close the loop, we need to define a quantitative measurement that describes how good the matching result is, and to modify the input of image matching for improving its output when the matching result is not accepted.

The definition of this quantitative measurement is based on the fact that the SIFT feature locations are efficiently detected by identifying Maxima and Minima of the Difference-of Gaussian (DoG) scale space as explained in chapter 4.

Each set of the SIFT features of the query image GF^{query} and of the model image GF^{model} are divided into two subsets, one for the Maxima and the other for the Minima SIFT features.

$$\begin{aligned} GF^{model} &= GF^{model}_{\min} \cup GF^{model}_{\max} \\ GF^{query} &= GF^{query}_{\min} \cup GF^{query}_{\max} \end{aligned} \tag{7.1}$$

By matching Maxima SIFT features with Maxima and Minima with Minima, two independent sets of positive matches $GM_{\max}$ and $GM_{\min}$ are obtained.

$$\begin{aligned} GM_{\min} &= match\left(GF^{model}_{\min}, GF^{query}_{\min}\right) \\ GM_{\max} &= match\left(GF^{model}_{\max}, GF^{query}_{\max}\right) \end{aligned} \tag{7.2}$$

From these sets of positive matches, two independent affine transformations (Maxima and Minima affine transformations) can be estimated using RANSAC algorithm [45].

$$\begin{aligned} T_{\min} &= RANSAC(GM_{\min}) \\ T_{\max} &= RANSAC(GM_{\max}) \end{aligned} \tag{7.3}$$

Since both affine transformations are estimated through two different channels affected by different noise reasons (Maxima- and Minima-mismatches), the degree of the dissimilarity between them $Dis(T_{\max}, T_{\min})$ reflects the degree of goodness of the matching outcome.

Generally, when affine transformations are computed, it can be distinguished between two cases: at least one of the transformations is correct or both are wrong

In the first case, if the dissimilarity $Dis(T_{\max}, T_{\min})$ is less than a pre-defined threshold δ, which means both transformations are correctly estimated since two independent streams of matches return the same information, hence the object is well-detected and its pose is estimated with a sufficient degree of accuracy. Otherwise both transformations are given as a feedback to a fuzzy controller to select the correct transformation to warp the model image. The SIFT features are then extracted from the new produced model image and matched to the query image, hence two new affine transformations are estimated and their dissimilarity is computed. The process is repeated until the dissimilarity between the current transformations or the dissimilarity between one of them and the identity matrix is less than a certain threshold. The last termination condition is due to the feedback loop is designed to make the target object appearance in model image as similar as possible to in the query image.

The second case can be distinguished when the output of the closed loop does not converge to the identity matrix, which means that the query image does not involve the target object or it is very difficult to be detected.

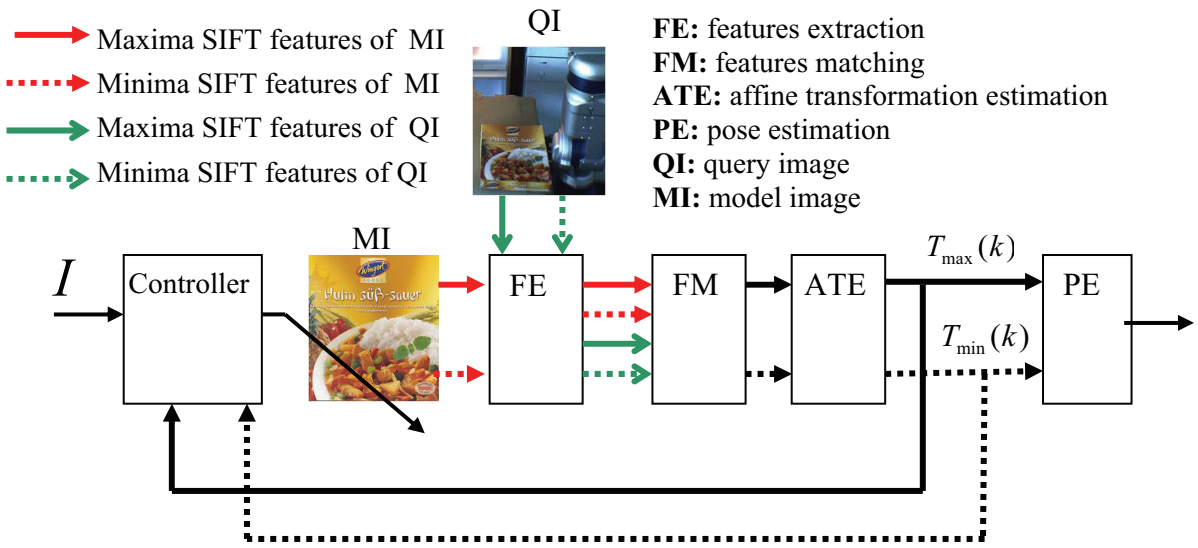


Figure 7.3: proposed closed loop object recognition system.

7.3. Dissimilarity between Two Affine Transformations

In general, because at least three non-collinear corresponding points between two images are required to determine the affine transformation, it is also needed at least three non-collinear points to compute the dissimilarity between two affine transformations T_1 and T_2:

Assuming that $p_1(a,a), p_2(a,-a)$ and $p_3(-a,a)$ are three non-collinear points at the plane xy, where a is arbitrary value.

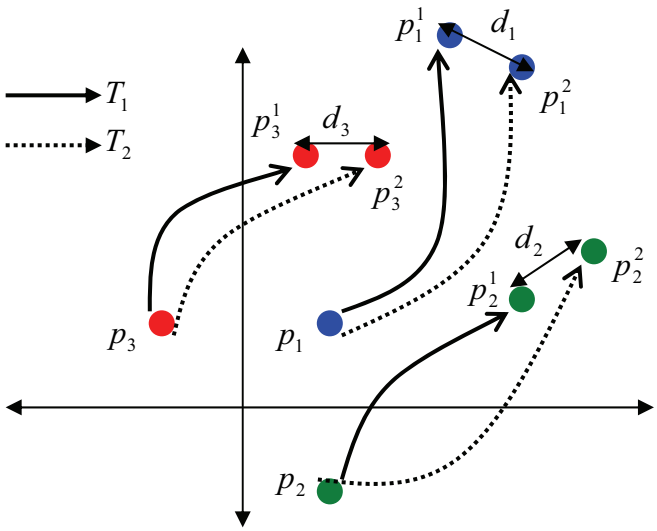


Figure 7.4: Dissimilarity between two affine transformations.

Each one of these points is mapped by each affine transformation.

$$p_i^1 = T_1 \cdot p_i$$
$$p_i^2 = T_2 \cdot p_i \qquad (7.4)$$

where $i = 1,2,3$

Hence the dissimilarity $Dis(T_1, T_2)$ is defined as:

$$Dis(T_1, T_2) = \frac{1}{3}\sum_{i=1}^{3}\left(d\left(p_i^1, p_i^2\right)\right) \tag{7.5}$$

where $d(p_1, p_2)$ is the Euclidian distance between two points $p_1(x_1, y_1)$ and $p_2(x_2, y_2)$ computed as follows:

$$d(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{7.6}$$

7.4. Fuzzy Controller

Generally, a fuzzy knowledge-based system is composed of two modules, a knowledge base represented by a set of conditional rules, and an inference engine, which makes the rules work in response to the system inputs.

An important application of fuzzy knowledge-based system is the control of complex, nonlinear systems [93]. Control algorithms with fuzzy controllers offer better response and efficiency in case of complex nonlinear systems when compared to conventional controllers.

The basic difference between fuzzy and conventional controllers is that the latter are designed using a mathematical model of the process being controlled. On the contrary, fuzzy controllers are based on the synthesis of the knowledge which is provided by human expertise to construct a set of rules (in the form of IF–THEN statements) [94].

Depending on the structure of the rules, two types of fuzzy controller can be distinguished: fuzzy relational and fuzzy functional models [95]. In the functional fuzzy controller proposed by Takagi and Sugeno [96], the rule consequents are crisp functions of the linguistic input variables calculated using a weighting method, whereas by relational fuzzy controllers, the mapping from the input to the output linguistic variables is represented by a fuzzy relation. The most widely used relational fuzzy model is the Mamdani model [97] illustrated in figure 7.5.

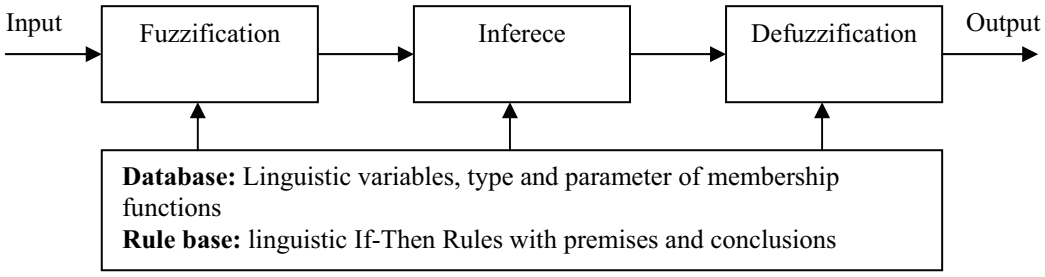


Figure 7.5: Structure of relational fuzzy controller.

Because no mathematical model for the open loop object recognition system is available, which is necessary for classical control methods, the system is controlled by a fuzzy model designed to select one of the feedbacked transformations. The selected transformation is used to produce new model image for matching operation in the next iteration.

For each channel (Maxima and Minima) of the object recognition system, the error $e_{\max/\min}$ (dissimilarity between the transformation and the identity matrix computed according equation 7.5) and the derivation of the error $\Delta e_{\max/\min}$ are chosen as inputs:

$$\begin{aligned} e_{\max/\min} &= Dis\left(T_{\max/\min}, I\right) \\ \Delta e_{\max/\min} &= e_{\max/\min}\left(k-1\right) - e_{\max/\min}\left(k\right) \end{aligned} \tag{7.7}$$

where I is the identity transformation given by.

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{7.8}$$

The output is defined as a quality index, which is a real value in the range [0, 1] representing how correct the corresponding affine transformation is estimated.

Once the quality index has been computed for both channels, they are compared and the transformation corresponded to the highest quality index is selected for the next matching iteration as long as the termination criteria are not met. Figure 7.6 presents the block diagram of the proposed fuzzy controller.

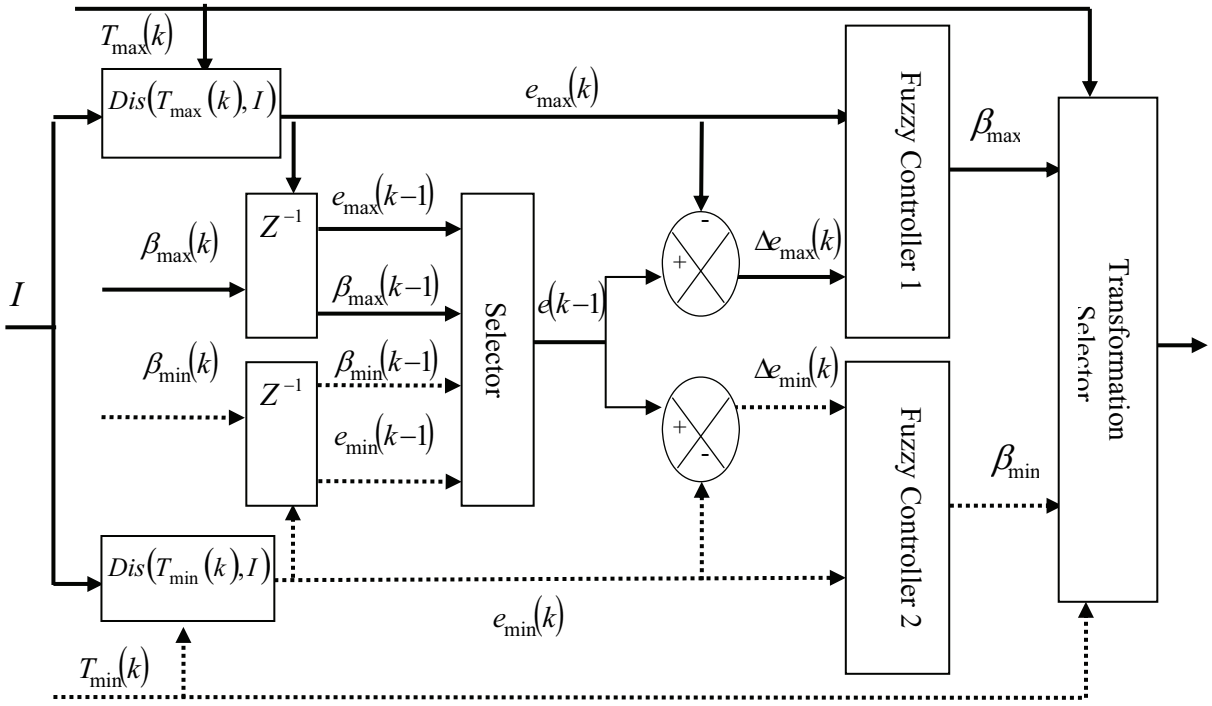


Figure 7.6: Fuzzy- based system for affine transformation selection.

Generally, the fuzzy controller consists of three main stages: the formation of membership functions (fuzzification), the definition and the evaluation of fuzzy rules (Inference) and selecting defuzzification method (defuzzification).

7.4.1. Fuzzification

In fuzzification, the crisp inputs are converted into fuzzy inputs using the corresponding membership functions in the knowledge base. The selection of membership functions depends on many aspects. For example, the use of Gaussian membership functions for specifying fuzzy sets is desired in many applications because they exhibit properties that are continuously, which facilitates sensitivity analysis over the obtained fuzzy inference system [98]. If the goal is to obtain simple linear interpolations and simple numerical evaluations, the triangular and trapezoid membership functions are preferred. Figure 7.7 illustrates three different types of membership functions used for fuzzification.

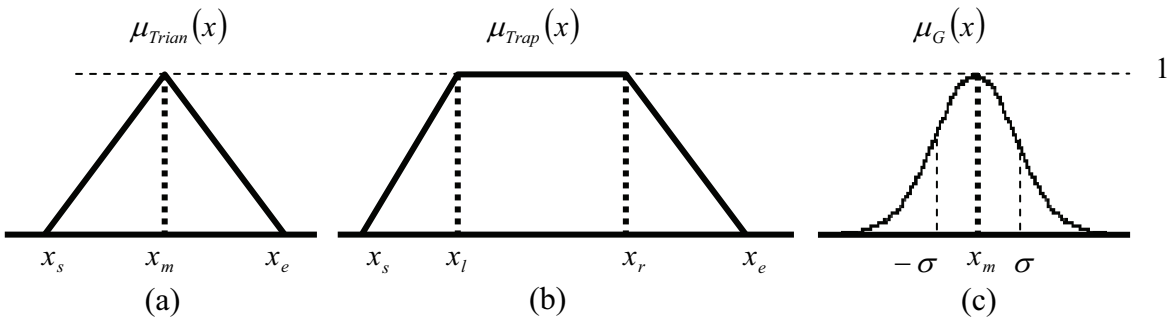


Figure 7.7: Three types of widely used membership functions: (a) triangular, (b) trapezoid, and (c) Gaussian type membership functions.

The mathematical formulas of triangular, trapezoidal and Gaussian memberships are given by the following equations:

$$\mu_{Trian}(x) = \begin{cases} 0 & x < x_s \\ \dfrac{x - x_s}{x_m - x_s} & x_s \le x \le x_l \\ \dfrac{x_e - x}{x_e - x_m} & x_r \le x \le x_e \\ 0 & x > x_e \end{cases}$$

$$\mu_{Trap}(x) = \begin{cases} 0 & x < x_s \\ \dfrac{x - x_s}{x_l - x_s} & x_s \le x \le x_l \\ 1 & x_l \le x \le x_r \\ \dfrac{x_e - x}{x_e - x_r} & x_r \le x \le x_e \\ 0 & x > x_e \end{cases} \tag{7.9}$$

$$\mu_{Trian}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-x_m)^2}{2\sigma^2}}$$

In the proposed model, triangular shape is selected as the main membership function. But a few trapezoidal membership functions are used at the marginal ranges.

The membership functions used for fuzzification and their ranges of each input and output are presented in Figure. 7.8. The range values are determined experimentally. For each inputs, three linguistic variables are used: (S: small, M: medium, and L: large for the error $e_{\max/\min}$) and (Z: zero, N: negative and P: positive for the error derivation $\Delta e_{\max/\min}$), while for the output five linguistic variables are defined as: very small (VS), small (S), medium (M), large (L) and very large(VL).

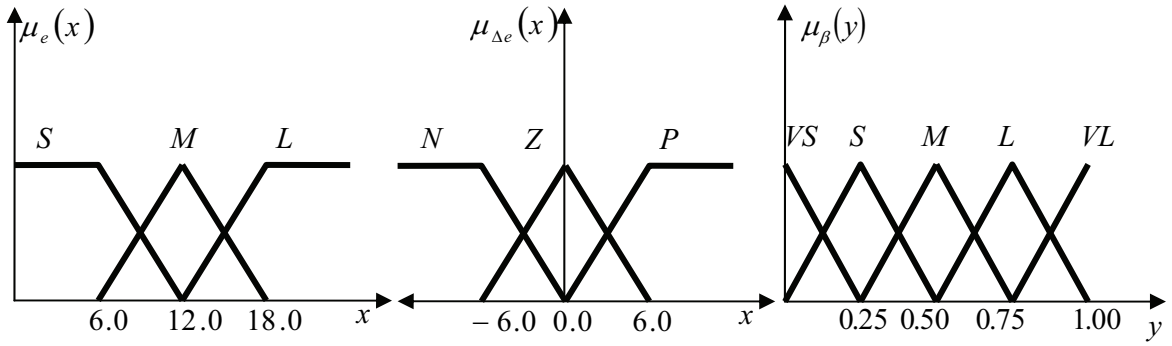


Figure 7.8: Input and output membership functions and their ranges.

The membership functions for each variable considered in developed system and the default limit values corresponding to 100% and 0% of certainty for each membership function are stored in the linguistic database as illustrated in Table 7.1.

Table 7.1: The database of linguistic variables.

μ_e	**Triangular**			**Trapezoidal**			
	x_s	x_m	x_e	x_s	x_l	x_r	x_e
S				**-∞**	**-∞**	**4**	**8**
M	**4**	**8**	**12**				
L				**8**	**12**	∞	∞
$\mu_{\Delta e}$	**Triangular**			**Trapezoidal**			
	x_s	x_m	x_e	x_s	x_l	x_r	x_e
N				**-∞**	**-∞**	**-6**	**0**
Z	**-6**	**0**	**6**				
P				**0**	**6**	∞	∞
μ_β	**Triangular**			**Trapezoidal**			
	x_s	x_m	x_e	x_s	x_l	x_r	x_e
VS				**-∞**	**-∞**	**0**	**0,25**

S	**0**	**0,25**	**0,5**				
M	**0,25**	**0,5**	**0,75**				
L	**0,5**	**0,75**	**1**				
VL				**0,75**	**1**	∞	∞

7.4.2.Inference

The relationship between the inputs and the outputs in a fuzzy model is characterized by a set of linguistic statements called as fuzzy rules [99]. They are defined based on the human expert knowledge and observations from experimental work. The number of fuzzy rules in a fuzzy system is related to the number of inputs and the number of fuzzy sets for each input variable. In this study for each channel, there are three input variables, each of which is classified into three linguistic variables. Therefore, the number of rules for this model is set to 9. The experimental and expert knowledge of the model is described in the table 7.2.

Table 7.2: Rule base of proposed fuzzy controller.

$\beta_{max/min}$		$e_{max/min}$		
		S	M	L
$\Delta e_{max/min}$	N	M	S	VS
	Z	L	M	S
	P	VL	L	M

The fuzzy rules are used in fuzzy control in order to define the map from the fuzzified inputs of the fuzzy controller to its fuzzy outputs [101]. In this model, knowledge is interpreted IF–THEN rules and multiple statements are joined by AND connective. The fuzzy rules in linguistic form are shown in Table 7.3.

Table 7.3: Fuzzy-expert rules in linguistic form

Rule 1	**IF** (N is L) **AND** (e is S) **AND** (Δe is N)	**THEN**(β is M)
Rule 2	**IF** (N is L) **AND** (e is S) **AND** (Δe is Z)	**THEN**(β is L)
Rule 3	**IF** (N is L) **AND** (e is S) **AND** (Δe is P)	**THEN**(β is VL)
Rule 4	**IF** (N is L) **AND** (e is M) **AND** (Δe is N)	**THEN**(β is S)
Rule 5	**IF** (N is L) **AND** (e is M) **AND** (Δe is Z)	**THEN**(β is M)
Rule 6	**IF** (N is L) **AND** (e is M) **AND** (Δe is P)	**THEN**(β is L)
Rule 7	**IF** (N is L) **AND** (e is L) **AND** (Δe is N)	**THEN**(β is VS)

Rule 8	**IF** (N is L) **AND** (e is L) **AND** (Δe is Z)	**THEN**(β is S)
Rule 9	**IF** (N is L) **AND** (e is L) **AND** (Δe is P)	**THEN**(β is M)

The rules that have exactly the same consequences must be combined into a single rule with OR-operators so that for each output linguistic variable, there is exactly one consequent for each possible antecedent in the rule base:

Table 7.4: Combined fuzzy-expert rules.

R_{VL}	**IF** (Rule3.)	**THEN**(β is VL)
R_L	**IF** (Rule 2 OR Rule 6)	**THEN**(β is L)
R_M	**IF** (Rule 1 OR Rule 5 OR Rule 9)	**THEN**(β is M)
R_S	**IF** (Rule 4 OR Rule 8)	**THEN**(β is S)
R_{VS}	**IF** (Rule 7)	**THEN**(β is VS)

The fuzzy inference method used is defined by a combination of two operators, the disjunctive and conjunctive operators. In literatures, many such operators are available [102]. The most common used disjunctive and conjunctive operators are the AND- and the OR-operators for the conjunction and the disjunction respectively.

Rules activation degrees are computed by evaluating the minimum between two membership degrees that are combined with AND-Operator and the maximum between membership degrees that are combined with OR-operator.

$$\begin{aligned} x \quad and \quad y &= MIN(x, y) \\ x \quad or \quad y &= MAX(x, y) \end{aligned} \tag{7.10}$$

The activated rules are aggregated into one fuzzy set for the output variable by evaluating the maximum between rules activation degrees.

7.4.3. Defuzzification

Output of a fuzzy process needs to be a single scalar quantity as opposed to a fuzzy set. Defuzzification is the conversion of a fuzzy quantity to a precise quantity. In literatures, many defuzzification methods have been proposed by investigators in recent years, such as: centroid method, weight average method, mean of max.-membership method, center of sums method, center of largest area method, first (or last) of maxima method. The selection of the defuzzification technique is critical and has a significant impact on the speed and accuracy of the fuzzy model.

In this model, centroid of area defuzzification method is used because it has been used generally and gives more reliable results than the others [100][101]. In this method, the resultant membership functions are developed by considering the union of the output of each rule, which means that the overlapping area of fuzzy output sets is counted only once, providing more results.

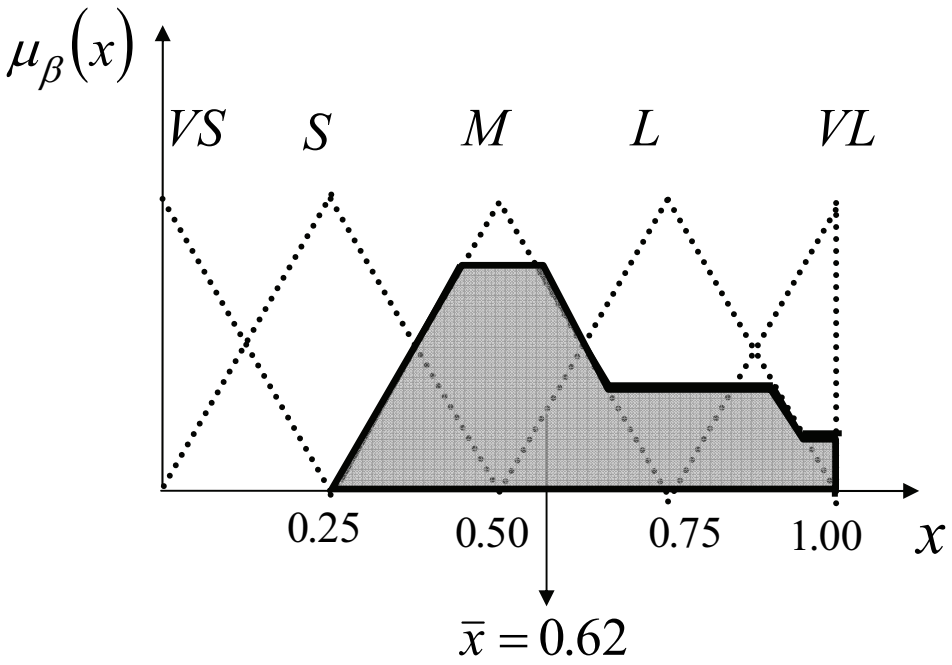


Figure 7.9: Graphical representation of centroid area method.

Figure 7.9 shows the basic graphical representation of center of area defuzzification method. In this Figure, the shape refers to the remaining area of active fuzzy sets that are controlled by the related fuzzy rules. The center of gravity of the shape is mathematically obtained by the following equation:

$$\bar{x} = \frac{\int_{x_s}^{x_e} x \cdot \mu_\beta(x) dx}{\int_{x_s}^{x_e} \mu_\beta(x) dx} \tag{7.11}$$

7.5. Experimental Results

To evaluate the performance of the proposed system, many experiments were conducted on different pairs of images (model and query images) of standard image database [103] and real world images. Each model image presents single target object, while the corresponding query image includes the target object captured in cluttered background under different conditions (illumination, viewpoint, partial occlusion). The system is evaluated on 100 image pairs of the database [103] (two examples are shown in Figure 7.10) and 100 real world image pairs from working scenarios of the robotic system FRIEND II acquired with its stereo camera system (an example is shown in Figure 7.11).

Figure 7.10: Two examples of the database images (left column) model images, (right column) query images.

Model image **query image**

Figure 7.11: An example of used real world images.

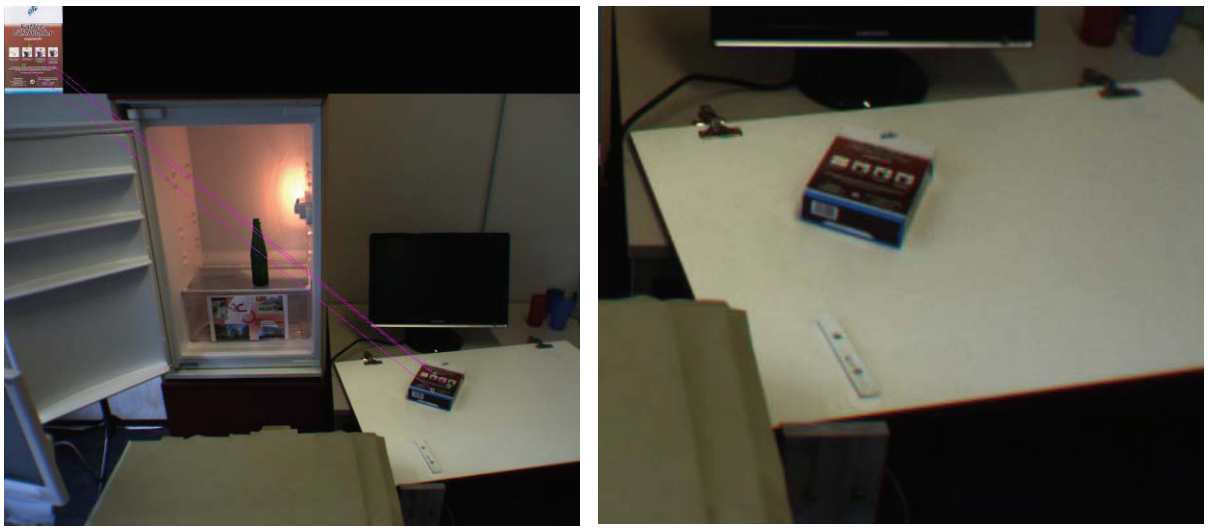
The pose estimation of the object represented by the model image in the scene represented by corresponding query images is done parallel from two independent matching channels (Maxima and Minima SIFT feature matching). The matching process is repeated until both estimated poses are nearly equal. Because both poses are provided from different independent information channels and each of them consists of 6 independent parameters, the equality of both poses means that both are correctly estimated with an error lower than their difference. The results of pose estimation for some examples are listed in table 7.5. As evident from Table 7.5, the positional errors (the error of the translations along x-axis and y-axis of the camera coordinate system) is less that 1 mm, while the error of the translation along optical axis is less than 3 mm. The angular errors, i.e. pitch, roll and yaw angle errors are less than 0,5 degree.

Table 7.5: Comparison between object poses estimated by Minima and Maxima SIFT matches.

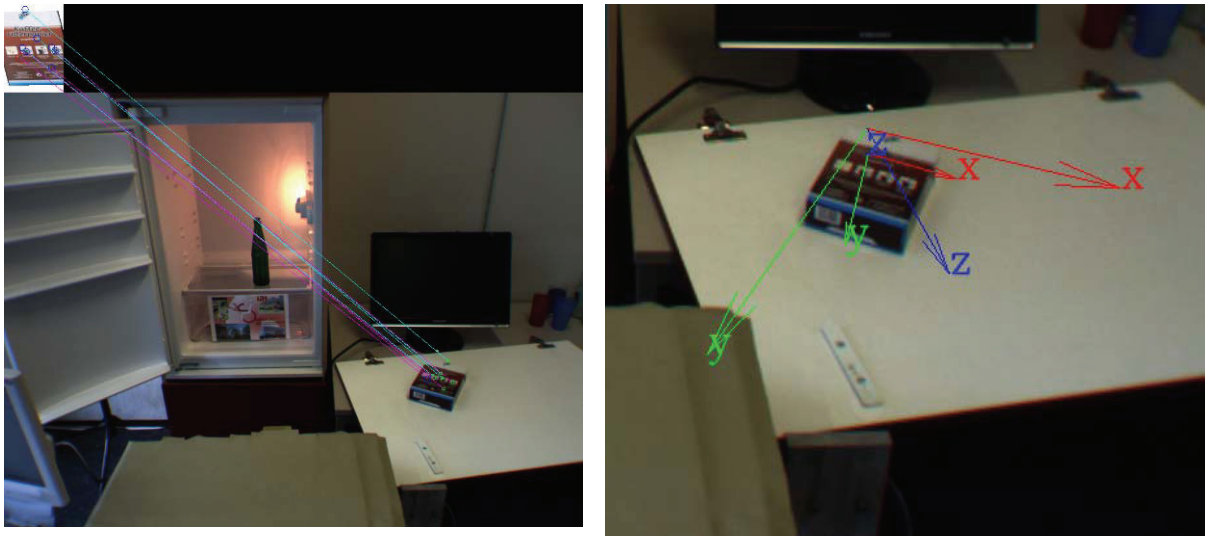
Pose estimated from Maxima correspondences						Pose estimated from Minima correspondences					
T_x	T_y	T_z	α	β	γ	T_x	T_y	T_z	A	β	γ
30,76	10,8	123,22	-50,77	20	27,57	31,1	11,07	124,57	-50,74	20,63	27,23
41,59	10,38	157,31	-60,77	18,03	26,41	41,57	10,66	157,35	-62,1	19,09	26,1
-24,87	-10,09	151,36	16,21	-22,56	-1,32	-25,46	-10,23	156,52	15,54	-21,78	-1,37
-19,09	18,79	120,35	-64,76	1,77	-1,25	-18,99	18,62	119,16	-65,5	1,79	-1,16
-16,08	27,69	97,17	-64,33	1,61	-0,71	-16,01	27,43	96,66	-63,64	0,83	-0,67
56,2	-2,51	140,94	26,4	42,1	10,62	57,12	-2,34	143,42	25,4	43,52	9,98
5,99	20,11	90,29	-1,92	-42,29	174,2	6,19	20,13	90,41	0,47	-41,94	173,58

An example for the progress of the image matching and pose estimation of target object (coffee filter package) is illustrated in Figure 7.6. One notes that the accuracy and convergence rate of estimated pose has been improved during iterations.

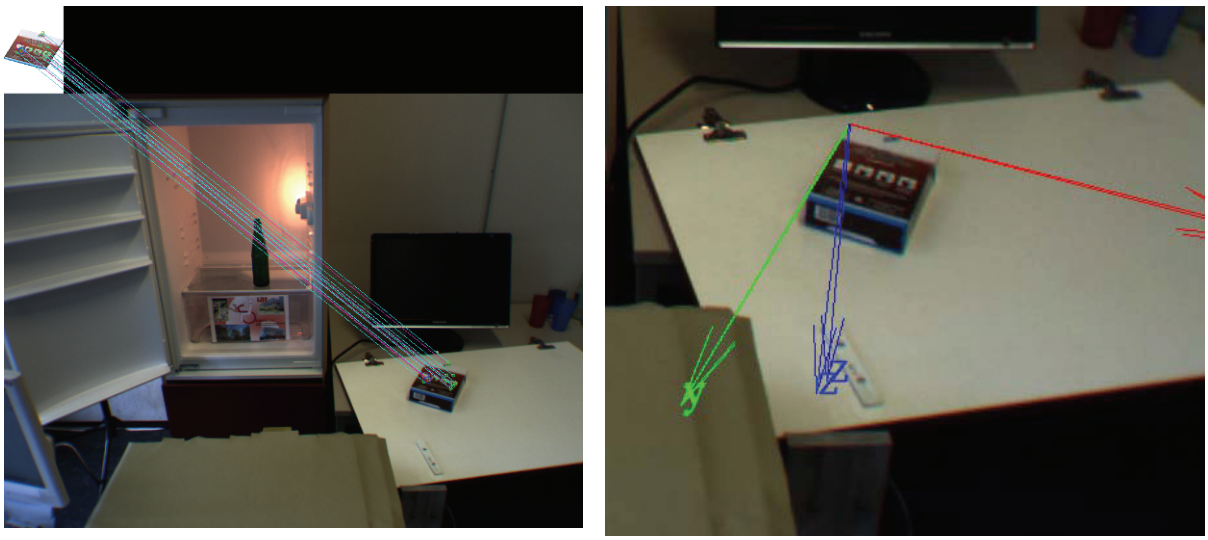
Iteration 1: no pose is estimated because only 3 matches are found, but one affine transformation is estimated, which is used to warp the model image in the next iteration.



Iteration 2: Ex=63.692, Ey=51.119, Ez=475.28, Eα=38.12, Eβ=49.01, Eγ=26.76.



Iteration 3: Ex=1.145, Ey=0.395, Ez=4.195, Eα=2.10, Eβ=1.84, Eγ=0.01



Iteration 4:Ex=0.329, Ey=0.152, Ez=0.91, Eα=0.59, Eβ=0.17, Eγ=0.50

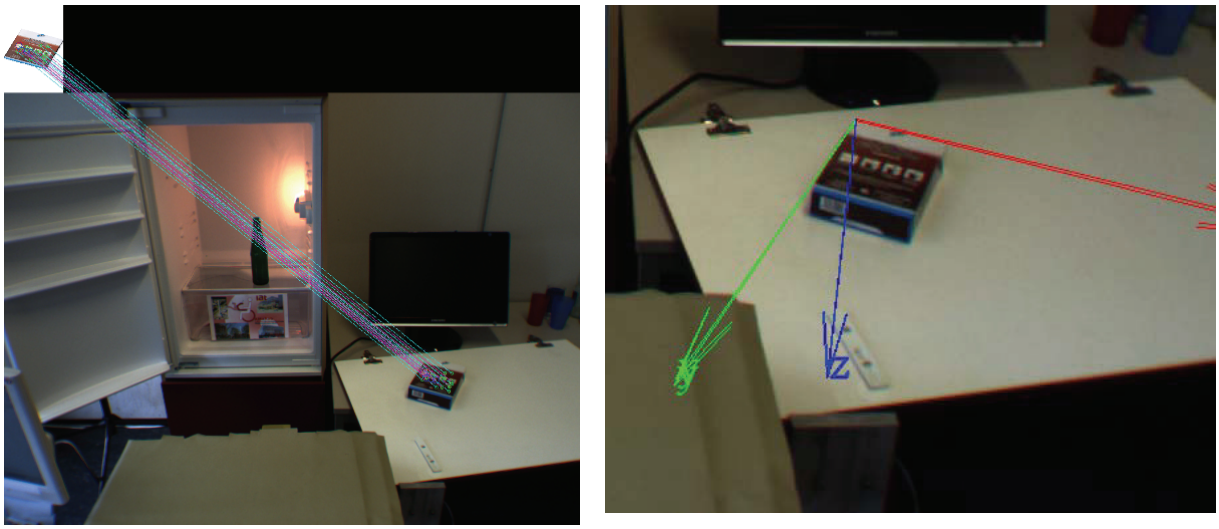


Figure 7.12: update of image matching and pose estimation results during time. Left image matching result and right its corresponding pose estimation result. In each iteration, the translation errors (Ex, Ey and Ez in mm) and rotation angle errors (Eα, Eβ and Eγ in degree) are listed. Note that the number of matches is increased, the difference of the both estimated poses is decreased and convergence to the pose of target object.

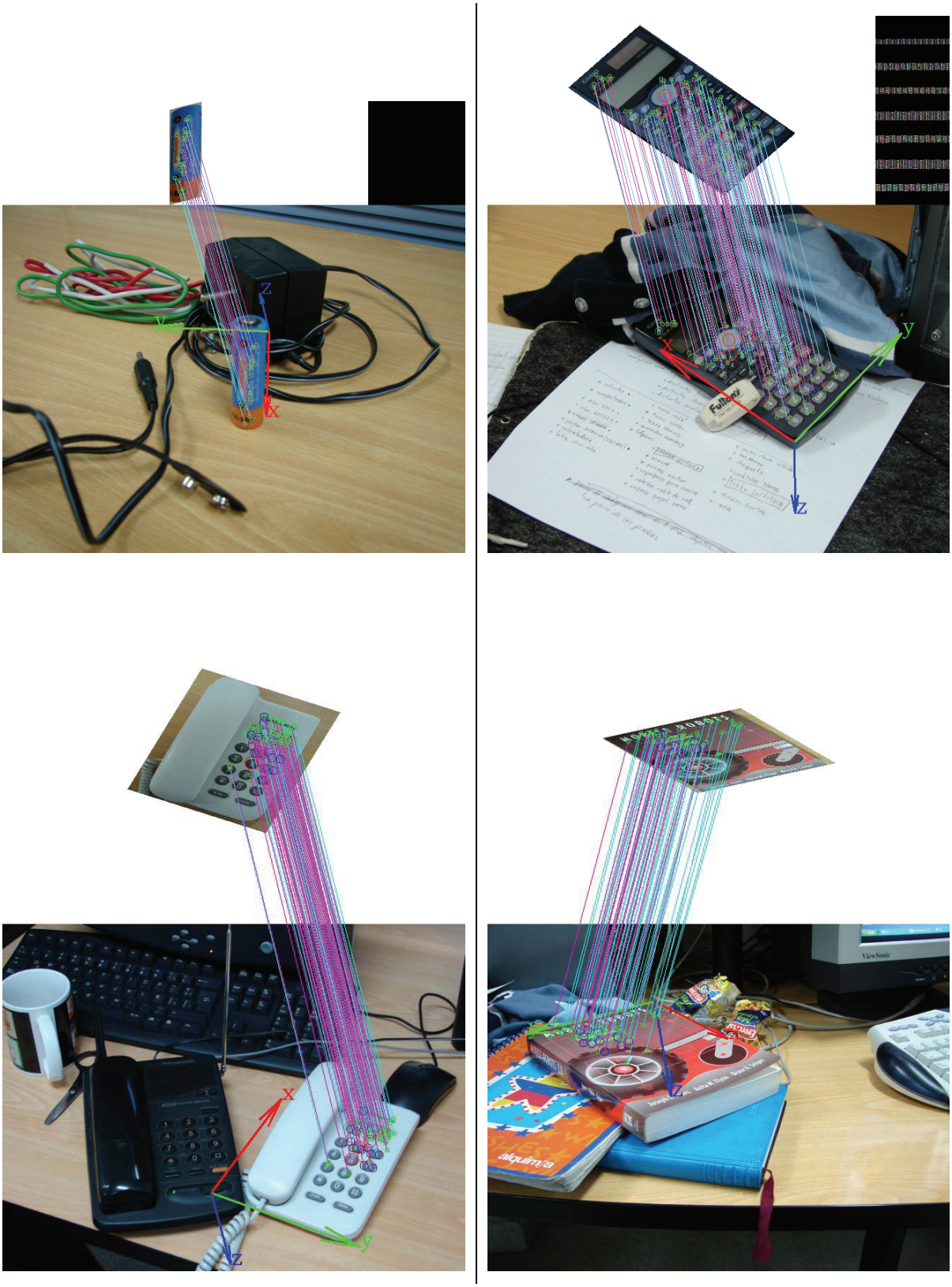


Figure 7.13: Matching and pose results of the final iteration for some model and query image pairs.

7.6. Conclusions

In this Chapter, we proposed an improvement to the currently used local features based object recognition systems.

This improvement corresponds to introduce a fuzzy based closed loop control system for object recognition, which increases significantly the robustness and the accuracy of estimated pose of the object to be recognized. This is achieved by extracting two kinds of features from the images to be matched, and taking into account the type of features while mapping these images, this operation allows us to define a controlled value. Because the system is non-linear and no mathematical model is available, a fuzzy controller is used. This controller system uses fuzzy-expert rules, triangular/trapezoid membership functions for fuzzification and centroid area method for defuzzification process. The new proposed approach was tested using real images, acquired with the camera system of FRIEND robotic system and images of a standard dataset The obtained results showed that the proposed approach is very promising.

8. Conclusion and Outlook

Image matching is the core task for many computer vision applications, such as object recognition, robot navigation, stereo vision, camera calibration and visual servo control. Although considerable research has been conducted in recent years on the development of image matching algorithms, there are still open research challenges in the context of the reliability, accuracy and processing time.

The modern used methods for image matching are based on local features of the image, such as SIFT, SURF and GLOH.

SIFT is the most widely used method for image matching, which has recently attracted much attention in the computer vision and photogrametry communities since SIFT features are highly distinctive, and invariant to scale, rotation, viewpoint and illumination changes. In addition, they are robust against noise, occlusion and background clutter and easy to extract and to match against a large database of features.

Generally, there are two main drawbacks of SIFT method, the former is that the computational complexity of the algorithm increases rapidly with the number of key-points, especially at the matching step due to the high dimensionality of the SIFT feature descriptor. The latter is that the SIFT features are not robust to large viewpoint changes. These drawbacks limit the reasonable use of SIFT algorithm for robot vision applications since they require often real-time performance and may need dealing with large viewpoint changes.

This dissertation has proposed three new approaches to address the constraints faced when using the SIFT features for robot vision applications: Speeded up SIFT feature matching, robust SIFT feature matching and the inclusion of the fuzzy based closed loop control structure for robust object recognition and pose estimation.

SIFT feature correspondences finding is the part of the matching algorithm that takes the most amount of processing time, especially when the numbers of features being compared are relatively large. Since most robot vision applications require real-time response, this thesis has proposed a new strategy to speed up feature matching. This strategy is based on hashing of SIFT features into several clusters during feature extraction phase using some new attributes that are computed from SIFT orientation histogram (SIFT-OH) or SIFT descriptor (SIFT-D). Thus, in the feature matching phase only features are compared that share nearly the same corresponding attributes. This strategy has speeded up image matching by a factor of about 1000 according to exhaustive search, and has also improved matching quality significantly.

Some robot vision applications, such as camera calibration and pose estimation require robust feature matching. Even though SIFT features are reasonably invariant; they can not accommodate large changes in viewpoint, which is the core problem of camera calibration and pose estimation. This problem is caused by either the absence of true positive correspondences or their portion is insufficient for fitting methods to work correctly. In this thesis, a new procedure has been proposed to determine the scale factor between images to be matched. This procedure divides SIFT features into different sub-sets based on their octaves. The matching process is done in prioritized order, so that only the features of the same scale ratio are compared on each step. At the same time a scale ratio histogram (SRH) is

constructed. Only matches of the step corresponding to the highest SRH bin are provided to the fitting method. This restriction decreases the portion of outliers among positive matches leading to improve the performance of the fitting methods, such as Random Sample Consensus (RANSAC) [45] or Least Median of Squares (LMS) methods.

Finally, a fuzzy based closed loop control system has been included to increase the accuracy of object recognition and pose estimation. The idea is to extract two different types of SIFT features, from model and query images. These features are matched separately providing two independent affine transformations. The dissimilarity between these transformations is used as signal indicates to the matching quality. The transformations themselves are feedbacked to a controller to improve the matching result. Because there is no mathematical model for the system is available, a fuzzy controller is used. The dissimilarities between the identity matrix and each of the affine transformations are delivered to fuzzy controller. The task of the controller is to select the best transformation to produce new model image used in next matching iteration as long as the termination criterion is not met. As termination criterion, the dissimilarity between the affine transformations or the dissimilarity between on of them and the identity matrix is used, if at least one of these dissimilarities is less than a certain threshold, the loop is broken down. The proposed controller is based on fuzzy-expert rules and uses triangular/trapezoid membership functions for fuzzification, max/min operators for inference, and centroid area method for defuzzification processes.

Finally, we suggest some possible directions for future work of the proposed methods as follows.

The proposed Fast and Robust SIFT feature matching methods are based on dividing features into several subsets, before they are matched with one another. Therefore The feature matching process can be parallelized so that it could run parallel on multi-core systems in order to achieve more speeding-up of the feature matching, which open the door widely to utilize the SIFT feature matching for applications require high real-time performance.

In Chapter 5, a new theorem in context of the probability density function of the sum/difference of circular random variables has been introduced and proven. This theorem can be used generally to speed up the nearest neighbor searching in high dimensional space. The optimum speed up factor can be obtained by uniformly mapping the sample points onto low dimensional space.

In Chapter 6, because the scale factor between images to be matched was computed by the ratio of octave pair between which the number of found positive matches is maximum and the downscaling of image size by factor 2 in both direction from octave to octave, the scale factor can be obtained only in the form of 2^k. In order to refine the obtained scale factor, we suggest to divide SIFT features according to the intervals they are extracted from and then running the feature matching in prioritized order to get more precise scale factor.

In Chapter 7, a fuzzy based closed loop control system has been proposed to use mainly for object recognition and pose estimation. Another possible application which can be significantly enhanced by using fuzzy based closed loop control system is the camera calibration.

Bibliography

[1] D. http://en.wikipedia.org/wiki/Time-of-flight_-_cite_ref-2Modarress, P. Svitek, K. Modarress and D. Wilson. Micro-optical sensors for boundary layer flow studies. *ASME Joint U.S.-European Fluids Engineering Summer Meeting*, 2006.

[2] M. Levoy, J. Ginsberg, J. Shade, D. Fulk, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson and J. Davis. The Digital Michelangelo Project: 3D Scanning of Large Statues (PDF). *Proceedings of the 27th annual conference on Computer graphics and interactive techniques,* pp.131–144, 2002

[3] G. Ni, Q. Liu. Analysis and prospect of multi-sources image registration techniques. *Opto-Electronic Engineering*, 31(9), pp.1-6, 2004.

[4] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), pp 91–110, 2004.

[5] Y. Ke and R. Sukthankar. PCA-sift: A more distinctive representation for local image descriptors. *International conference on Computer Vision and Pattern Recognition*, pp.506-513, 2004.

[6] H. Bay, T. Tuytelaars and L. Van Gool. SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, 110(3), pp.346-359, 2008.

[7] L. Ledwich and S. Williams. Reduced sift features for image retrieval and indoor localization. *In Australian Conference on Robotics and Automation*, 2004.

[8] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on pattern analysis and machine intelligence*, 27(10), 2005.

[9] C. Martens, O. Prenzel and A. Gräser. The Rehabilitation Robots FRIEND-I & II: Daily Life Independency through Semi-Autonomous Task-Execution. *I-Tech Education and Publishing (Vienna, Austria)*, pp.137–162. ISBN 978-3-902613-04-2, 2007.

[10] O. Ivlev, C. Martens and A. Gräser. Rehabilitation Robots FRIEND-I and FRIEND-II with the dexterous lightweight manipulator. *Restoration of Wheeled Mobility in SCI Rehabilitation* 17, 2005.

[11] I. Volosyak, O. Ivlev and A. Gräser. Rehabilitation robot FRIEND-II - the general concept and current implementation. *Proceedings of the. 9th International Conference on Rehabilitation Robotics(ICORR* 2005), pp.540-544, 2005.

[12] Y.I. Abdel-Aziz and H.M. Karara. Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. *In Proceedings of the Symposium on Close-Range Photogrammetry*, *Falls Church, VA: American Society of Photogrammetry*, pp.1-18, 1971.

[13] W. J. Wilson, C. C. W. Hulls, and G. S. Bell. Relative end-effector control using Cartesian position-based visual servoing. *IEEE Transactions Robot. Automat.*, vol. 12, pp.684 696, Oct. 1996.

[14] D. Dementhon and L. S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2), pp.123–141, June 1995.

[15] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions. Robot. Automat*, vol. 8, pp.313–326, June 1992.

[16] E. Malis, F. Chaumette and S. Boudet. 2-1/2d visual servoing. *IEEE Transactions on Robotics and Automation*, vol. 15, pp.238-250, Apr. 1999.

[17] G. Hager. Calibration-Free Visual Control Using Projective Invariance. In *proceedings of the 5th International Conference on Computer Vision,* 1995.

[18] K. Hashimoto, T. Ebine and H. Kimura. Visual servoing with hand–eye manipulator–optimal control approach. *IEEE Transactions on Robotics and Automation.* 12(5), pp.766–774, 1996.

[19] B. Espiau. Effect of camera calibration errors on visual servoing In robotics. *In proceedings of the 3rd International symposium on Experimental Robot*, Kyoto, Japan, Oct. 1993.

[20] J.F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 8(6), pp.679-698, Nov 1986.

[21] D. Marr and E.C. Hildreth. Theory of Edge Detection. *In Proceedings of the Royal Society of London*. 207, pp.187-217, 1980

[22] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. *Doctoral dissertation Technical Report, Robotics Institute, Carnegie Mellon University,* CMU-RI-TR-80-03, Sep. 1980.

[23] C. Harris and M.J. Stephens. A combined corner and edge detector. *In Proceedings of the 4th Alvey Vision Conference*, pp.147–152, 1988.

[24] T. Lindeberg. Discrete Scale-Space Theory and the Scale-Space Primal Sketch. *Doctoral dissertation, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden,* May 1991.

[25] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11(3), pp.283-318, 1993.

[26] S.W. Teng and G. Lu. Image indexing and retrieval based on vector quantization, *Journal Pattern Recognition*, 40(11), pp.3299–3316, 2007.

[27] G. Pass, R. Zabih and J. Miller. Comparing images using color coherence vectors. *In ACM 4th International Conference on Multimedia, Boston, Massachusetts, United States*, pp.65–73, 1996.

[28] Huang, J., 1997. Image indexing using color correlograms. *In IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Juan*, pp.762–768, 1997.

[29] A. Pikaz and A. Averbuch. An efficient topological characterization of gray-levels textures using a multi-resolution representation. *Graphical Models Image Process*, 59, pp.1–17, 1997.

[30] L. Chen, G. Lu and D. Zhang. Effects of Different Gabor Filter Parameters on Image Retrieval by Texture, *In proceedings of the 10th International Multimedia Modeling Conference*, pp.273-278, 2004.

[31] D. K. Park, C.S. Won and S.J. Park. Efficient use of mpeg-7 edge histogram descriptor. *ETRI Journal*, 24(2), pp.23–30, 2002.

[32] J.R. Carr and F.P. De Miranda. The semivariogram in comparison to the co -occurrence matrix for classification of image texture. *Geoscience and Remote Sensing*, 36(6), pp.1945-1952, 1998.

[33] H. Freeman and L.S. Davis. A Corner Finding Algorithm for Chain Coded Curves. *IEEE Transactions on Computers*, vol. 26, pp.297-303, 1977.

[34] E. Persoo and K. Fu. Shape Discrimination Using Fourier Descriptors, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, pp.170-179, 1977.

[35] M.K. Hu. Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8(2), pp. 179-187, Feb. 1962.

[36] M. Teague. Image analysis via the general theory of moments. *Journal of Optical Society of America*, 70(8), pp. 920-930, Aug. 1980.

[37] Y. Rubner, C. Tomasi and L.J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40 (2), pp.99–121, 2000.

[38] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), pp.509–517, September 1975.

[39] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. *In Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, 14(2), pp.47-57, June 1984.

[40] D. Comer The Ubiquitous B-Tree. *Computing Surveys*, 11(2), pp.123–137, June 1979

[41] A. Gionis, P. Indyk and R. Motwani. Similarity Search in High Dimensions via Hashing. *In proceedings of the 25th International Conference on Very Large Database (VLDB)*, pp.518–529, 1999

[42] T. Lindeberg. Scale-space theory: a basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21(2), pp.224--270, 1994.

[43] J.H. Firedman, J.L. Bentley and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions Mathematical Software*, pp.209-226, 1977.

[44] P.J. Rousseeuw and A.M. Leroy. Robust Regression and Outlier Detection. *New York: Wiley Series in Probability and Statistics*, 79(1984), pp.871–880 1987.

[45] M. Fischer and R. Bolles. Random sample consensus: A paradigm to model fitting with applications to image analysis and automated cartography, *Communications of the ACM*, 24(6), pp.381–395, 1981.

[46] K. Mikolajczyk and Schmid, Indexing based on scale invariant interest points. In *proceedings of the 8th International Conference on Computer Vision (ICCV)*, vol. 1, pp.525-531, 2001.

[47] T. Lindeberg. Feature detection with automatic scale selection, *International Journal of Computer Vision*, 30(2), pp.79-116, 1998.

[48] C. Valgren and A. Lilienthal. SIFT, SURF and Seasons: Long-term Outdoor Localization Using Local Features. *In proceedings o the European Conference on Mobile Robots (ECMR)*, pp.253-258, 2007.

[49] S.N. Sinha, J.M. Frahm, M. Pollefeys and Y. Genc. GPU-based video feature tracking and matching. *Technical report, Department of Computer Science, UNC Chapel Hill*, 2006.

[50] S. Heymann, K. Miller, A. Smolic, B. Froehlich, and T. Wiegand. SIFT implementation and optimization for general-purpose gpu. *In Proceedings of the 15th International Conference in Central Europe on Computer Graphics (WSCG)*, pp.317–322, January 2007.

[51] A. Chariot and R. Keriven. GPU-boosted online image matching. *In Proceedings of the 19th Conference on Pattern Recognition, Tampa, Florida, USA*. Dec 2008.

[52] S. Se, H. Ng, P. Jasiobedzki, T. Moyung. Vision based modeling and localization for planetary exploration rovers. *In Proceedings of the 55th International Astronautical Congress*, 2004.

[53] Q. Zhang, Y. Chen, Y. Zhang and Y. Xu. SIFT implementation and optimization for multi-core systems. *In Proceedings of the 10th Workshop on Advances on Parallel and Distributed Computing Models*, pp. 1-8, 2008.

[54] B. Leibe, K. Mikolajczyk and B. Schiele. Efficient clustering and matching for object class recognition. *In Proceedings of the 17th British Machine Vision Conference (BMVC)*, 2006.

[55] C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.1-8, 2008.

[56] H. Yang, Q. Wang and Z. He. Randomized sub-vectors hashing for high-dimensional image feature matching. *In Proceeding of the 16th ACM international on Multimedia*, pp.705-708, 2008.

[57] E. Valle, M. Cord and S. Philipp-Foliguet. High-dimensional descriptor indexing for large multimedia databases. *In Proceedings of the 17th Conference on Information and Knowledge Management (CIKM)*, pp.739-748, 2008.

[58] M. E. Houle and J. Sakuma. Fast Approximate Similarity Search in Extremely High-Dimensional Data Sets. *In Proceedings of the 21st International Conference on Data Engineering (ICDE)*, pp.619-630, 2005.

[59] M. Muja and D. G. Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration, *In Proceedings of the 4th International Conference on Computer Vision Theory and Applications (VISAPP)*, pp. 331-340, 2009.

[60] E. Batschelet. (1981). Circular Statistics in Biology. *Academic Press, London*. ISBN 0-12-081050-6,1981.

[61] S.R. Jammalamadaka, A. Sengupta. Topics in Circular Statistics. *World Scientific, River Edge, N.J*, ISBN 0-521-35018-2, 2001.

[62] M.K. Simon, M.M. Shihabi and T Moon. Optimum Detection of Tones Transmitted by a Spacecrft. *TDA Progress Report (TDA PR)*, pp.42-123, 69-98, November 1995.

[63] F. Alhwarin, D. Ristic Durant and A. Gräser. Speeded up image matching using split and extended SIFT features. *In Proceedings of the 5th International Conference. on Computer Vision Theory and Applications (VISAPP)*, pp.287-295, 2010.

[64] F. Alhwarin, D. Ristic Durant and A. Gräser. VF-SIFT: Very fast SIFT feature matchnig. *In Proceedings Annual Symposium German Association for Pattern Recognition (DAGM)*, pp 222-231, 2010.

[65] Image database, available at:

http://lear.inrialpes.fr/people/Mikolajczyk/Database/index.html

[66] Fast Library for Approximate Nearest Neighbors (FLANN):

http://people.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN

[67] D.G. Lowe. Object recognition from local scale-invariant features. *In Proceedings of the of 7th IEEE International Conference on Computer Vision (ICCV)*, pp.1150-1157, September 1999.

[68] D.G. Lowe. Local feature view clustering for 3D object recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.682-688, December 2001.

[69] K. Fleischer. Two tests of pseudo random number generators for independence and uniform distribution. *Journal of statistical computation and simulation*, vol. 52, pp.311–322, 1995.

[70] Y. Lee, K. Lee, and S. Pan. Local and Global Feature Extraction for Face Recognition, *Springer-Verlag Berlin Heidelberg*, 2005.

[71] Y. Ke, R. Suthankar and L. Huston. Efficient Near-Duplicate Detection and Sub image Retrieval. *In Proceedings of the ACM International Conference on Multimedia,* pp.869–876, 2004.

[72] H. P. Moravec. Towards Automatic Visual Obstacle Avoidance. *In Proceedings of the. 5th International Joint Conference on Artificial Intelligence.(IJCAI)*, pp.584, 1977.

[73] C. Schmid and R. Mohr. Local Greyvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5), pp.530-535, 1997.

[74] G. Michael, G. Helmut and B. Horst. Fast Approximated SIFT. *In Proceedings of the Asian Conference on Computer Vision*, pp.918-927, 2006.

[75] S.K. Vuppala, S.M. Grigorescu, D.Ristic Durant, and A. Gräser. Robust color Object Recognition for a Service robotic Task in the System FRIEND II. *In Proceedings of the* 10th International Conference on Rehabilitation Robotics (ICORR), 2007.

[76] J. Davis and M. Goadrich. The Relationship between Precision-Recall and ROC Curves, *In Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp.233-240, 2006.

[80] J. Peng and B. Bhanu. Closed-Loop Object Recognition Using Reinforcement Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*,1998.

[81] B. Bhanu and J. Peng. Adaptive Integrated Image Segmentation and Object Recognition. *IEEE Transactions on Systems Man and Cybernetics Part C*, 30(4), pp.427-441, 2000.

[82] M. Mirmehdi, P.L. Palmer, J. Kittler and H. Dabis. Feedback Control Strategies for Object Recognition, *IEEE Transactions on Image Processing*, 8(8), pp.1084-1101, August 1999.

[83] D. Ristic Durant, S.K. Vuppala and A. Gräser. Feedback Control for Improvement of Image Processing: An Application of Recognition of Characters on Metallic Surfaces. *In Proceedings of the IEEE International Conference on Computer Vision Systems (ICVS)*, pp.39, 2006.

[84] D. Ristic Durant and A. Gräser. Performance Measure as Feedback Variable in Image Processing. *EURASIP Journal on Applied Signal Processing,* Volume 2006.

[85] S.K. Vuppala, S.M. Grigorescu, D. Ristic Durant, and A. Gräser. Robust color Object Recognition for a Service robotic Task in the System FRIEND II. *In Proceedings of the 10th International Conference on Rehabilitation Robotics (ICORR)*, 2006.

[86] Y. Lee, K. Lee, and S. Pan. Local and Global Feature Extraction for Face Recognition, *Springer-Verlag Berlin Heidelberg*, 2005.

[87]A. Chachich, A. Pau, A. Barber, K. Kennedy, E. Olejniczak, J. Hackney, Q. Sun, and E. Mireles. Traffic sensor using a color vision method. *In Proceedings of the International.Society for Optical Engineering*, vol. 2902, pp.156-164,January 1997.

[88] B. Schiele. Model-free tracking of cars and people based on color regions. *In Proceedings of the IEEE International Workshop Performance Evaluation of Tracking and Surveillance (PETS), Grenoble, France*, pp.61–71, 2000.

[89] C. Schmid and R. Mohr. Matching by local Invariants. *Technical Report Institut National de Recherche en Informatique et en Automatique (INRIA)*, No 2644, August 1995.

[90] C. Schmid, R. Mohr. Local Grey value Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5), pp.530–4, 1996.

[91] N.I. Fisher. Statistical analysis of circular data, *Cambridge University Press, Cambridge, UK*, ISBN 0-521-35018-2, Oktober 1995.

[92] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), pp.1330-1334, 2000.

[93] T.J. Ross. Fuzzy Logic with Engineering Applications, *McGraw-Hill, USA*, ISBN: 0-470-86075-8, 1995.

[94] S.G. Tzafestas and G.G. Rigatos. Design and stability analysis of a new sliding-mode fuzzy logic controller of reduced complexity. Machine *Intelligence and Robotic Control*, 1(1), pp.27–41, 1999.

[95] R. Babuska and H.B. Verbruggen. An overview of fuzzy modeling for control, *Control Engineering Practice*, 4(11), pp.1593–1606, 1996.

[96] T. Takagi, M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transaction Systems, Man and Cybernetics*, 15(1)), pp.116–132, 1985.

[97] E.H. Mamdani, S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1), pp.1–13, 1975.

[98] K.M. Tay and C.P. Lim. Fuzzy FMEA with a guided rules reduction system for prioritization of failures. *International Journal of Quality&Reliability Management*, 23(8), pp.1047-1066, 2006

[99] O. Yilmaz, O. Eyercioglu and N.N.Z. Gindy. A user friendly fuzzy based system for the selection of electro discharge machining process parameters. *Journal of Materials Processing Technology*, 172(3), pp.363-371, 2006.

[100] K. Hashmi, I.D. Graham and B. Mills. Data selection for turning carbon steel using fuzzy logic. *Journal of Materials Processing Technology*,135, pp.44–58, 2003.

[101] M. Arghavani, M. Derenne and L. Marchand, Fuzzy logic application in gasket selection and sealing performance. *International Journal of Advanced Manufacturing Technology*, 18, pp.67–78, 2001.

[102] C. Lee. Fuzzy logic in control systems-parts 1 and 2. *IEEE Transactions on systems, Man and Cybernetics*, 10(2), pp.404-434, 1999,

[103] UCH100 image database, available at: http://vision.die.uchile.cl/.

List of Abbreviations

SIFT	Scale Invariant Feature Transform
ANN	Approximate Nearest Neighbor
BBF	Best-Bin-First
BCI	Brain computer Interface
BOB	Bounds Overlap Ball
BWB	Ball Within Bounds
CCV	Color Coherence Vector
CH	Color Histogram
CM	Color Moments
CMP	Chip Multiprocessor
DLT	Direct Linear Transformation
DoF	Degrees of Freedom
DoG	Defference of Gaussian
DoH	Determinant of Hessian
DoM	Difference of Means
EHD	Edge Histogram Descriptor
FA-SIFT	Fast Approximated SIFT
FPGA	Field Programmable Gate Array
FLANN	Fast Library for Approximate Nearest Neighbors
FoV	Field of View
FRIEND	Functional Robot arm with frIENdly interface for Disabled people
GLOH	Gradient Location and Orientation Histogram
GPU	Graphics Processing Unit
HKMT	Hierarchical K-Means Tree
HSV	Hue-Saturation-Value color space
HTD	Homogeneous Texture Descriptor
HVS	Hybrid Visual Servoing
IBVS	Image Based Visual Servoing
ICRVs	Independent Circular Random Variables
k-d	k-dimensional tree
LMS	Least Median of Squares
LoG	Laplacian of Gaussian
LSH	Locality Sensitive Hashing
MRKDTs	Multiple Randomized KD-Trees
NNDR	Nearest Neighbor Distance Ratio
NNS	Nearest Neighbor Search
PBVS	Postion Based Visual Servoing
PCA	Principal Components Analysis
PDF	Probability Density Function
QFD	Quadratic Form Distance
RANSAC	Random Sample Consensus

RGB	Red-Green-Blue color space
R-SIFT	Reduced SIFT
RSVH	Randomized Sub-Vector Hashing
SASH	Spatial Approximation Sample Hierarchy
SIFT-D	SIFT Descriptor
SIFT-OH	SIFT Orientation Histogram
SOHs	Sub-Orientation Histograms
SRH	Scale Ratio Histogram
SS	Scale Space
SURF	Speeded Up Robust Feature
SVD	Singular Value Decomposition
TNN	Thresholded Nearest Neighbor
VF-SIFT	Very Fast SIFT