Real-Time Recognition of Planar Targets on Mobile Devices

A Framework for Fast and Robust Homography Estimation

by

Hamid Bazargani

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Faculty of Engineering
University of Ottawa

Abstract

The present thesis is concerned with the problem of robust pose estimation for planar targets in the context of real-time mobile vision. As a consequence of this research, individual developments made in isolation by earlier researchers are here considered together. Several adaptations to the existing algorithms are undertaken yielding a unified framework for robust pose estimation. This framework is specifically designed to meet the growing demand for fast and robust estimation on power-constrained platforms.

For robust recognition of targets at very low computational costs, we employ feature-based methods which are based on local binary descriptors allowing fast feature matching at run-time. The matching set is then fed to a robust parameter estimation algorithm in order to obtain a reliable homography. On the basis of our experimental results, it can be concluded that reliable homography estimates can be obtained using a device-friendly implementation of the Gaussian Elimination algorithm. We also show in this thesis that our simplified approach can significantly improve the homography estimation step in a hypothesize-and-verify scheme. The author's attention is focused not only on developing fast algorithms for the recognition framework but also on the optimized implementation of such algorithms. Any other recognition framework would similarly benefit from our optimized implementation.

Acknowledgements

Foremost, I would like to express my heartfelt gratitude toward Prof. Robert Laganière for giving me the opportunity to work under his supervision. His continuous support, motivation and immense knowledge helped me in all the time of research and writing of this thesis. I would also like to thank Olexa Bilaniuk for his help and valuable discussions throughout this research.

I thank my past and present fellow labmates at VIVA Lab, especially Mehdi Arezoomand, Ehsan Fazl Ersi, Mina Rafi Nazari and Navid Tadayon for the friendly ambiance during the last 17 months. My special thanks also goes to M. Esmaeel Mousa Pasandi for the sleepless nights we were working together before deadlines.

Last but not the least, I would like to thank my family for the spiritual support, constant encouragement and unconditional love they have been providing throughout my life.

Dedication

This thesis is dedicated to my father who gave me the courage to go after my dreams and to my mother who taught me not to quit.

Contents

Nomenclature **xiii**

1 Introduction **1**
1.1 Motivation . . . 2
1.2 Problem Statement . . . 4
1.3 Contributions . . . 5
1.4 Thesis Organization . . . 6

2 Feature Detection and Description **8**
2.1 Feature Detection . . . 9
2.1.1 Scale Invariant Feature Transform (SIFT) . . . 9
2.1.2 Speeded Up Robust Features (SURF) . . . 9
2.1.3 KAZE features . . . 10
2.1.4 FAST features . . . 11
2.2 Descriptor Extraction . . . 12
2.2.1 SIFT Descriptor . . . 13
2.2.2 SURF Descriptor . . . 14
2.2.3 KAZE Descriptor . . . 14
2.2.4 Local Binary Descriptors . . . 15
2.3 Conclusion . . . 19

3 Target Detection System Overview **21**
3.1 Classification-based Methods . . . 22
3.1.1 Matching as a Classification Problem . . . 23
3.1.2 Random Ferns Classification . . . 23
3.1.3 Histogrammed Intensity Patches . . . 25
3.1.4 Trained Binary Descriptors . . . 30

3.2 Conclusion 30

4 The Geometric Pose Estimation **31**
4.1 The Perspective Projection 32
4.1.1 Intrinsic Parameters 33
4.1.2 Extrinsic Parameters 33
4.1.3 Distortion Coefficients 34
4.2 The Homography Estimation 34
4.2.1 Direct Linear Transformation (DLT) 35
4.2.2 Perspective-n-Point (PnP) 36
4.3 Distance Evaluation 38
4.3.1 The Algebraic error 38
4.3.2 The Geometric error 38
4.3.3 The re-projection error 39
4.4 Conclusion 39

5 Robust Model Estimation **41**
5.1 Maximum Likelihood Estimation (MLE) 42
5.2 RANdom SAmpling Consensus (RANSAC) 43
5.2.1 PROSAC 44
5.2.2 Randomized RANSAC (R-RANSAC) 46
5.2.3 Preemptive RANSAC 47
5.2.4 Adaptive Real-time RANSAC (ARRSAC) 48
5.2.5 MLESAC 48
5.2.6 Guided MLESAC 49
5.2.7 Locally Optimized RANSAC (LO-RANSAC) 50
5.3 Conclusion 50

6 Robust Target Matching Framework **52**
6.0.1 Homography estimation by Gaussian Elimination 53
6.1 Robust Parameter Estimation 54
6.1.1 Termination criterion 55
6.1.2 Model verification 57
6.1.3 The Degeneracy Test 59
6.2 The Model Refinement 60
6.2.1 Robust M-estimator 60

6.3 Conclusion 62

7 Experimental Results **63**
7.1 Experimentation 63
7.1.1 Homography Estimation Performance 63
7.1.2 Target Recognition Performance 69

8 Conclusion **78**
8.1 Future Works 79

A Homography Estimation by Gaussian Elimination **81**

References **85**

List of Tables

7.1 Performance result of Homography estimation as in [56]. (I) is the number of inliers found. (K) and (K_rej) are the number of samples drawn and the number of samples rejected by the degeneracy test. (models) is the number of total hypotheses, (VPM) the number of verification per model. The symmetrical reprojection (error) is measured w.r.t. the ground truth. (time) indicates the execution time per frame in ms. Note that all reported results are averaged over a total of 500 runs. 65
7.2 Average number of total matches, inliers, required iterations and recognition rate are shown for the four targets with both GE and SVD method . 67
7.3 Performance result of Homography estimation for different verification methods 73
7.4 Performance result of Homography estimation for different stopping criteria 74
7.5 Performance results of the four sequences with different stopping criteria. 75

List of Figures

1.1 Examples of Augmented Reality applications for smart-phones 3
1.2 State-of-the-art technologies designed for computer vision purposes. . . . 4
1.3 Overall system architecture for recognition on mobile phone 5

2.1 Left half shows discrete finite second derivatives of the Gaussian kernel with respect to x and xy directions in order. Right half shows boxed approximation of the same filters. 10
2.2 16 pixel locations on Bresenham circle of radius 3 used in FAST technique. 12
2.3 Extraction of the SIFT descriptor from image gradients. 13
2.4 The BRIEF pattern with Gaussian distributed pairs. 17
2.5 a) The BRISK sampling pattern. b) Shows 512 short-distance pairs with $\|p_i - p_j\| < 9.75s$. c) Shows 870 long-distance pairs with $\|p_i - p_j\| > 13.67s$. 18
2.6 The retinal FREAK sampling pattern and the most 256 discriminant pairs depicted by lines. 19

3.1 The left image is 16×16 pixel original interest point appearance. The middle shows 8×8 down-sampled extracted patch. The right shows aligned rotated patch. 26
3.2 The above image shows 8×8 pixels quantized patch. The below images show 5 levels of quantized patch. White pixels represent location of pixels of a particular color. 26

3.3 Taylor's method for building a HIP model. Patches from different viewpoints falling within an approximate location constitute a class. A sample histogram of a pixel is shown as a red box. Histograms of five intensity levels for each class of the 8×8 patches are similarly computed. After thresholding all histograms, samples with frequency less than 0.05 are considered as rare bits and stored with 1 in a binary stream. The computed HIP's binary stream is then used to compare with different patches. Correct matches (green) and wrong ones (red) are determined according to the number of bits that hit one of the HIP's rare bit. 27
3.4 Two rotation-aware hash patterns with 5 and 13 samples. 28
3.5 Shows three subdivision levels of a polyhedron and resulting viewpoint distribution with up to 40 degrees out-of-plane rotation 29

4.1 Perspective model of a pinhole camera. 32
4.2 Geometric view of P3P 37

5.1 Examples of RANSAC deficiency. a) Setting the distance threshold too low limits the number of inliers (support), thus prevents RANSAC from convergence to the solution. b) Setting the threshold too high results in a wrong fit with a strong support. c) A geometric degenerate configuration. Sampling from a dense cloud of points yields an infinite number of models consistent with the set. 45

6.1 Robust homography estimation framework 53
6.2 Left shows χ^2 distribution with 1 degree of freedom. Right shows Binomial PDF and normal approximation for $n = 30$ and $\beta = 0.1$. The red area under the curves indicate p-value $= 0.05$ under H_0 57
6.3 a)Huber and Tukey's robust cost functions. b) Huber and Tukey's robust weight functions. 61

7.1 AR application based on our optimized homography estimation framework. 64
7.2 Homography estimation for one frame of each of our test videos. 66
7.3 Recognition rate for each test sequence, reported as the percentages of frames with maximal positional error less than 10 pixels. 68
7.4 Maximum positional error of the four sequences reported every 5 frames. 69
7.5 Per-frame average instruction fetch count for each H-estimator 70

7.6 Result of homography estimates (shown in red) for the sequence of images as used in [46]. 71
7.7 Shows number of extracted matches, inlier's proportion(Total inliers vs. total matches) and homography estimation time(ms) for each frame related to Figure (7.6) experiment. 72
7.8 Maximal positional error(px) for the homography estimation with different stopping criteria 76
7.9 Execution time for the homography estimation with different stopping criteria. 77

Nomenclature

Abbreviations

AR	Augmented Reality
AVX	Advanced Vector eXtension
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CCD	Charged Couple Device
CenSure	Center Surround Extremas
DEGSVD	Verifications Per Model
DLT	Direct Linear Transformation
DOF	Degree Of Freedom
DoG	Difference of Gaussian
FAST	Features from Accelerated Segment Test
FREAK	Fast Retina Keypoint
GE	Gaussian Elimination
HIP	Histogrammed Intensity Patches
HMD	Head-mounted Device
IMU	Inertial Measurement Unit
JNI	Java Native Interface
K-NN	K-Nearest Neighbor
LAPAC	Linear Algebra Package
LBD	Local Binary Descriptor
LO-RANSAC	Locally Optimized RANSAC
LS	Least Square
M-SURF	Modified Speeded Up Robust Features

MEMS	MicroelEctroMechanical System
MLE	Maximum Likelihood Estimation
MLESAC	Maximum Likelihood Estimation SAmple Consensus
NCC	Normalized Cross Correlation
NDK	Native Development Kit
ORB	ORiented Fast and Rotated BRIEF
PCA	Principal Component Analysis
PnP	Perspective-n-Point
POSIT	POSe with ITeration
PROSAC	PROgressive SAmpling Consensus
RANSAC	RANdom SAmple Consensus
SAD	Sum of Absolute Differences
SIFT	Scale Invariant Feature Transform
SIMD	Single Instruction Multiple Data
SLAM	Simultaneous Localization And Mapping
SOP	Scaled Orthographic Projection
SPRT	Sequential Probability and Ratio Test
SURF	Speed Up Robust Feature
SVD	Singular Value Decomposition
USAC	Universal SAmple Consensus
VPM	Verifications Per Model

Chapter 1

Introduction

Computer vision is a discipline of science that allows computing systems to better perceive and comprehend their environment. As a consequence of the rapid growth of the technology, computer vision algorithms have received a special attention in many applications including robotics, video analysis, image registration, visual odometry and augmented reality. One prominent example of the impact of these recent advances in computer vision is the field of Augmented Reality (AR) that has brought a new dimension to human life by enhancing perception and interaction with of the surrounding world.

Object recognition and visual tracking are now indispensable parts of computer vision applications. The scope of this research concerns the development of novel approaches for fast and reliable recognition of objects. The goal of object recognition is identifying the presence of a specific object or a class of objects in an image or a video sequence. Object recognition in a cluttered background and under a wide variety of viewpoints, shapes or lighting conditions is a challenging problem. In order to solve this problem, several matching algorithms such as block matching, gradient matching, feature matching and color matching algorithms have been developed. In this work, we will mainly study methods that rely on feature extraction to represent a specific target. The objective of feature extraction is to describe an image's local appearance. The local appearance of an object, which is extracted at multiple distinctive locations, describes the object through a set of representative signatures known as feature descriptors. These feature descriptors are then used to match and establish point correspondences between a target image and a live camera image. The methods for feature description and matching can be grouped in two major classes; namely *descriptor-based* and *classification-based* methods. In the former, local regions around distinctive features are detected and described by

highly invariant, but computationally expensive descriptors. In the latter, which employs simpler and faster descriptors, a great amount of computations is transferred to an offline training stage. When hand-held devices with limited computational power are used, the classification-based approaches are strongly preferred for real-time applications.

In addition to the recognition of the object, retrieving its precise location and orientation is a crucial part of many AR applications. Therefore, a *robust pose estimation* algorithm is needed to recover 3D position and orientation of the detected target in some world coordinates. Planar targets are particularly interesting for recognition because, in this case, the different views are related by a 2D homographic transformation. Real-time recognition in video is generally achieved using a matching framework in which keypoints detected in each frame are matched with the ones associated with a reference view of the planar target. The resulting set of putative correspondences is then validated through a robust *hypothesize-and-verify* scheme that aims at identifying a plausible homography, mapping the current target view to its reference image.

Since the developed framework concerns with target recognition for mobile phone based augmented reality applications, achieving faster and higher efficiency on low-powered devices is more preferable than a small gain in accuracy. Thus we are ready to lose some accuracy in favor of lower computational cost.

1.1 Motivation

The tremendous growth in popularity of mobile devices has made Augmented Reality one of the most innovative technologies of the decade. Augmented reality brings an interactive environment to users by merging the physical world as seen through a smartphone with virtual elements. In the recent times, AR applications on hand-held devices have been gaining a great interest by pushing the boundaries of e-commerce, automotive industry, tourism, education, entertainment industry, etc.

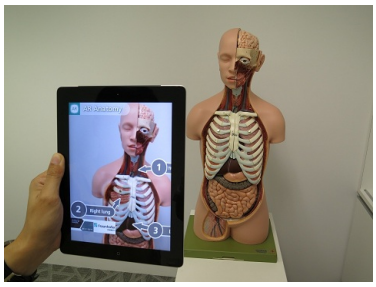
> *Annual revenues from mobile augmented reality (AR) services and applications will reach $1.2 billion by 2015, up from just over $180 million last year.*
>
> *Juniper Research- February 4th, 2014*

For instance, AR apps can help marketing companies to increase sales by providing customers with extra details and reviews of their products. AR is also used to im-

(a) ARDefender game

(b) Qualcomm AR demo

(c) Marketing AR app



(d) for education by Fraunhofer IDM@NTU

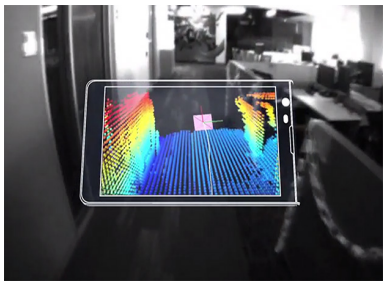
(e) Wikitude AR browser

(f) VW up! 3D AR by VolksWagen

Figure 1.1: Examples of Augmented Reality applications for smart-phones

prove educational performance and amend advanced learning technology. Furthermore, AR applications featured with visual and geolocational sensory inputs, allow the tourism industry to enhance users' experience. This can be achieved by bringing instant explanatory information pertaining to a historical place or a historic site. Figure 1.1 represents a few examples of augmented reality applications.

The technology behind several of these thriving AR applications is based on computer vision and object recognition algorithms. Indeed, object recognition and pose estimation provide the geometric information that is needed for augmenting the reality with a virtual object. For this reason, computer vision researchers have devoted considerable effort on developing efficient algorithms suitable for power-constrained platforms. Additionally, tech-player companies like Google, Nokia, Honda, Sony and Qualcomm have participated to this race. They either have developed efficient algorithms or designed advanced technologies such as MEMS and visual sensors, powerful processors, hand-held devices, Head-mounted displays (HMD), eye-wear devices or even contact lenses. Some of these state-of-the-art technologies are shown in Figure 1.2.



(a) Project Tango by Google

(b) Google Glass project

(c) Sony Head-mounted display

Figure 1.2: State-of-the-art technologies designed for computer vision purposes.

1.2 Problem Statement

The goal of this research is to develop a real-time system for hand-held devices that reliably detects planar targets in live video. The 3D position and orientation of the target should also be simultaneously recovered with respect to the camera coordinates. Although, the overall recognition framework should be able to locate 3D rigid objects, the focus of this thesis is on recognition and tracking of planar targets, because in this case, different views of the planar target are related by a 2D homographic transformation. The camera pose that is described by homography transformation, is then used to overlay a simple label or a virtual object on user's view as seen through the camera.

In order to estimate the camera pose parameters, a set of point correspondences between a target image and a query image has to be established. This is done by using a feature-based matching technique to produce and match top-level descriptors that describe the images with their local appearance. In feature-based techniques, the most salient feature points across the image are detected. These feature points are usually the most stable corners under a wide range of viewpoints. Once features are detected, a corresponding descriptor is then extracted that describes local region around that feature.

To improve robustness and speed of the matching process, multiple views of the reference target are generally synthesized during an offline process by warping it with different random perspective transforms. As we will discuss, matching algorithms are subject to produce some erroneous mismatches depending on the quality of the descriptors. To overcome the adverse effect of these wrong matches, a robust estimation scheme should be effectively used. In this work, we also study the hypothesize-and-verify scheme that constitutes the backbone of robust parameter estimation. Figure 1.3 demonstrates the

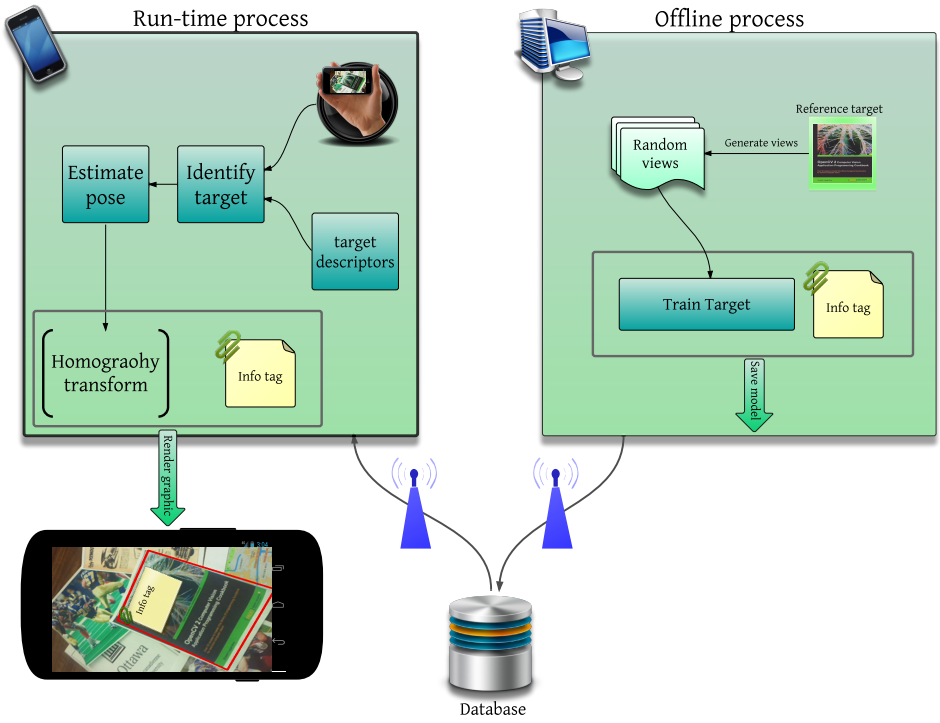


Figure 1.3: Overall system architecture for recognition on mobile phone

overall architecture of a planar recognition system based on offline training.

1.3 Contributions

The contribution of this research resides in the run-time process of the target recognition task. We comprehensively analyzed the problem of parameter estimation for fast and robust target matching through an optimal hypothesize-and-verify scheme. We have investigated possible limitations of RANSAC-style approaches to optimize our framework at both algorithmic level and implementation level. The optimized framework is specifically designed to be efficiently run on smart-phones and tablets. We also have developed a fast C++ software package by leveraging the state-of-the-art algorithms that have been studied over the years. Through experimentation we demonstrate that the frame rate of the recognition process runs at around 40 fps on a smart-phone with Quad-core 1.4 GHz Cortex-A9 processor and at 80 fps on a machine equipped with a 2.26 GHz CPU.

Additionally, We show that the robust estimation of a homography can be significantly improved by using a computationally efficient implementation of the well-known Gaussian Elimination method to solve the underlying set of equations. We show that in the context of planar target detection, the solution found provides a reliable estimate of the homography with an accuracy comparable to solutions based on the more commonly used Singular Value Decomposition methods.

Finally, the following paper has been accepted to the Mobile Vision Workshop at the 2014 CVPR conference.

- Hamid Bazargani, Olexa Bilaniuk, Robert Laganière: **Fast Target Recognition on Mobile Devices: Revisiting Gaussian Elimination for the the Estimation of Planar Homographies**. In Computer Vision and Pattern Recognition Workshops, 2014. The Fourth IEEE International Workshop on Mobile Vision.

The homography estimation method based on Gaussian Elimination has been designed and implemented by O.Bilianuk. The robust target recognition framework and all experimentations have been realized by the author.

1.4 Thesis Organization

From the following chapter onwards, we explore crucial steps toward a real-time object recognition framework. Our objective in this thesis aims at the development of efficient algorithms that are suitable for mobile devices.

In Chapter 2, we survey previous research in feature detection and description as a required background needed for readers who are not familiar with the principles of feature detection and/or descriptor extraction techniques.

In Chapter 3, feature matching algorithms from descriptor-based and classification-based categories are reviewed. For each class, popular methods are analyzed in the context of real-time target recognition. This chapter also proposes a classification-based method that works with FAST-9 features and BRIEF descriptors.

Chapter 4 explores the problem of camera pose estimation that relates world coordinates to the image plane. More specific explanation is then given in relation with the pose estimation in the context of planar targets.

In Chapter 5, we describe one of the key steps in robust target recognition. Different robust parameter estimation algorithms are explored and their limitations are discussed.

In addition to conventional regression methods, robust algorithms based on RANSAC strategy are presented.

Chapter 6 presents a unified and efficient framework for robust homography estimation. In this chapter we will also propose a fast homography estimation scheme by revisiting the Gaussian Elimination algorithm. Any other recognition framework would similarly benefit from our optimized implementation.

In Chapter 7, experimental results are presented. We benchmark our unified framework against the different implementations and we show that our framework outperforms the state-of-the-art algorithms. We also demonstrate through a comprehensive experimentation that reliable homography estimates can be obtained using a device-friendly implementation of the Gaussian Elimination algorithm. We show in this chapter that our optimized approach can improve by a factor of 20 the homography estimation step in a hypothesize-and-verify scheme.

In Chapter 8, we conclude the thesis and state the main findings of our research. Also potential works for future research are summarized.

Chapter 2

Feature Detection and Description

Feature detection and description have been playing a pivotal role in many Computer Vision applications including Augmented Reality, Robotic mapping (SLAM) , 3D scene reconstruction, motion tracking, etc. As an example of Augmented Reality application, a specific target may need to be recognized and tracked using only local features in an image or a video stream. These features have to be detected and described from the visual characteristics of the image. Features can then be used to identify and localize a target using a hand-held device.

Many different feature detection methods have been studied in the past decade to extract an abridged representation of image information in the form of so-called feature descriptors. Features can be either in the form of independent or connected points, edges or blobs. SIFT [41], SURF [7], Harris [25], FAST [61], BRISK [38], KAZE [4], CenSure [1] are some well known feature detection algorithms among which some provide invariance to different scales, affine transformations and rotation.

From the viewpoint of object recognition, an important aspect of feature detection algorithms, is that the detected features should be robustly detectable under possible changes of object appearance and in the presence of noise. Moreover, since mobile devices are nowadays the host of such algorithms, they have to be computationally inexpensive, yet efficient in terms of detecting the object of interest on power-constrained platforms such as smart-phones and tablets.

2.1 Feature Detection

As edges and corners are usually the most detectable and best features to track, many feature detectors try to locate these edges or corners in an image by performing specific operations on a surrounding region of each pixel. These operations include computing and assigning a score to each pixel representing its rate of cornerness. Thus, the algorithm controls the number of feature points by adjusting a threshold parameter for accepting or rejecting a point as a feature point. In the following, we go over some of the most popular and efficient feature detection algorithms.

2.1.1 Scale Invariant Feature Transform (SIFT)

Scale Invariant Feature Transform (SIFT) proposed by Lowe [41], is proven to robustly detect stable keypoints under fairly large level of affine transformations and noise. The idea is to search for a keypoint across the spatial domain in different scale-space of the image. From [33] and [40], Lowe chose the Gaussian kernel as a scale-space kernel and proposed to search for local exterema in the convolved image with difference of Gaussian functions:

$$DoG(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y), \tag{2.1}$$

where k is a scale factor, I is the image matrix and G is a Gaussian function with variance of σ^2,

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

After constructing the space octaves with different DoG images, stable keypoints are located by looking for local exterema of each DoG image. The SIFT algorithm includes comparing each pixel with 8 neighbors within its own scale along with its 18 neighboring pixels in the two adjacent octaves. Finally, a 3D quadratic surface is fitted to the detected exterema for providing sub-pixel precision.

2.1.2 Speeded Up Robust Features (SURF)

There have been huge efforts in the recent years to reduce the computational complexity of SIFT, while preserving its scale and rotation invariance property. Bay et al. in [7] proposed Hessian matrix to locate exterma both in scale space and location. $\mathcal{H}(x, \sigma)$ is denoted as follows:

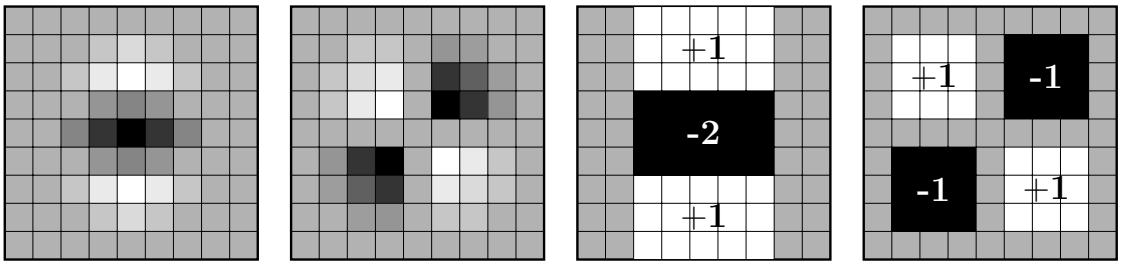


Figure 2.1: Left half shows discrete finite second derivatives of the Gaussian kernel with respect to x and xy directions in order. Right half shows boxed approximation of the same filters.

$$\mathcal{H}(x,\sigma) = \begin{bmatrix} \frac{\partial^2}{\partial x^2}\mathcal{G}(x,y,\sigma) * I(x,y) & \frac{\partial}{\partial x}\frac{\partial}{\partial y}\mathcal{G}(x,y,\sigma) * I(x,y) \\ \frac{\partial}{\partial x}\frac{\partial}{\partial y}\mathcal{G}(x,y,\sigma) * I(x,y) & \frac{\partial^2}{\partial y^2}\mathcal{G}(x,y,\sigma) * I(x,y) \end{bmatrix} \tag{2.2}$$

The Hessian matrix is made of second order derivatives of Gaussian kernels $\mathbb{G}(x,y,\sigma)$ convolved with the image with respect to x and y axis. For efficient computing of the Hessian matrix as quickly as possible, Bay et al. further approximated discrete and finite Gaussian functions with boxed approximations shown in Figure 2.1. By computing image integrals $I_\Sigma(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j)$, these approximations can be quickly computed using only four additions.

Since the keypoint score is evaluated with Hessian determinant, a weight factor is applied to reduce the difference between the determinant approximation and its true value.

$$\det \mathcal{H}(x,y) \simeq D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{2.3}$$

Where D_{xx}, D_{yy} and D_{xy} represent filtered image with the boxed filters shown in Figure 2.1. The weight factor 0.9 was found to more accurately estimate determinant of the second derivatives of the Gaussian kernel.

2.1.3 KAZE features

KAZE feature detector, proposed by Fernandez et al. [4] is another 2D multi-scale feature detection technique like SIFT and SURF. It detects and describes features by searching for image exterema spatially and in different scale levels. In contrary to SIFT and SURF, this approach builds a non-linear scale space rather than Gaussian space to suppress noise and extract predominant feature points.

Although Gaussian scale space is a simple and well known scale space representation, it can potentially lose distinctiveness and accuracy as it equally treats both desired high frequency details and noise. KAZE uses locally adaptive non-linear diffusion filtering to deliberately smooth out noise while preserving detail information (edges) unchanged. A general non-linear diffusion equation is denoted as

$$\frac{\partial L}{\partial t} = div(c(x, y, t).\nabla L) \tag{2.4}$$

Which incorporates divergence operator (div) to increase scale level and *Conductivity* function c to locally control the diffusion level proportional to the magnitude of image gradient ∇L. Different conductivity functions are proposed by Perona and Malik [52] and Weickert [83] in the form of equation 2.5 that gradually attenuates smoothing strength while approaching image boundaries:

$$c(x, y, t) = g(|\nabla L_\sigma(x, y, t)|), \tag{2.5}$$

where L_σ is Gaussian filtered image with variance of σ^2 and g exhibits a conductivity formulation.

2.1.4 FAST features

The FAST features abbreviated from *Features from Accelerated Segment Test* originally proposed by Edward Rosten and Tom Drummond [61, 60] is a highly expedited feature detection technique which is well suited for real-time detection and tracking processes. When examining each pixel p, FAST requires to carry out a simple test on a ring of 16 pixels encircling the pixel of interest. p is considered as a feature point, if there are n pixels in adjacent locations on the ring that are all either darker or all brighter than the center by a threshold δ_t. The threshold δ_t is employed to control the number of detected keypoints.

In the very early work of Rosten and Drummond, they chose $n = 12$ leading to an algorithm that discards non-corner points as quickly as examining only 3 out of 4 pixels in specific locations on the ring of Figure 2.2. So the large portion of examined points can be rejected quickly by testing pixels 1 and 9, then 5 and 13 before proceeding with the remaining locations.

However, the segment test performs highly efficient for $n = 12$, a machine learning approach is introduced in [61] to adjust segment's locations in a way that it still rejects as many tentative points for $n < 12$. Beside test locations, ordering of comparisons is

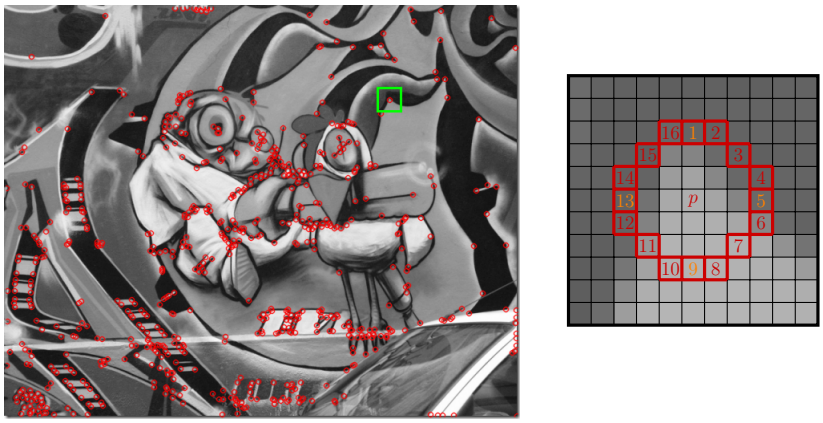


Figure 2.2: 16 pixel locations on Bresenham circle of radius 3 used in FAST technique.

optimally found in this machine learning approach. The learning process is started with finding FAST features in a set of training images by carrying out a full test over 16 pixels around each candidate. Taking a specific threshold into account, pixels are classified into three subsets of *much darker*, *much brighter* and *similar*. A decision tree is then formed employing ID3 algorithm [55] to minimize entropy of random variable which indicates whether a point is corner or not.

2.2 Descriptor Extraction

After detecting feature points from a query image, the next step is to assign a distinctive descriptor structure to each feature. This represents the pixel's appearance while preserving invariance to some possible variations. Each feature point needs to store the more meaningful information allowing to compare them against other features extracted from other images.

For better performance during the matching step, descriptors should remain robust against noise, scale, rotation and illumination changes and this makes descriptor extraction a time consuming process. SIFT, SURF BRISK, FREAK, BRIEF and ORB [41, 7, 38, 3, 12, 62] are different kinds of descriptor extraction techniques. Some of these techniques like SIFT and SURF use histogram of oriented gradients approach to extract descriptors while some others store binary data.

As smart-phones and tablets are becoming more popular in the recent years, detection algorithms should be fast and efficient enough to properly run on low memory and low-

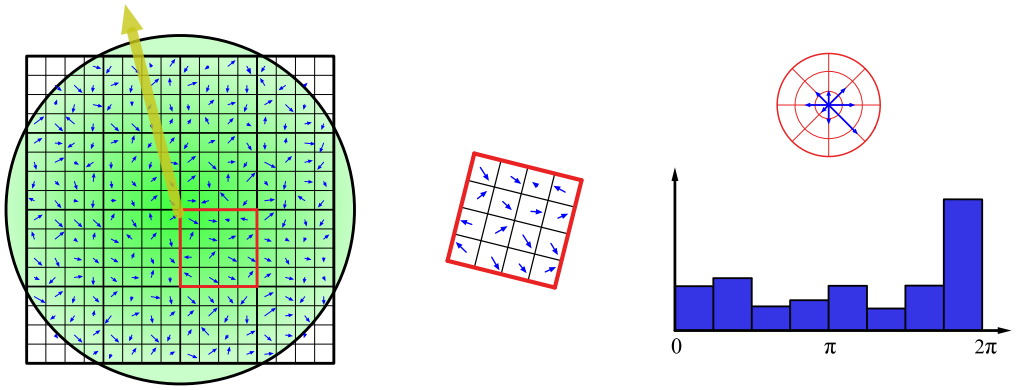


Figure 2.3: Extraction of the SIFT descriptor from image gradients.

powered devices (sometimes even with only fixed-point instructions). Thus the *Local Binary Descriptor* methods such as FREAK, BRISK, BRIEF and ORB perfectly suit these processors.

2.2.1 SIFT Descriptor

SIFT descriptor proposed by Lowe [41], is a 128-dimensional descriptor comprised of histograms of normalized gradients. Given a keypoint detected by SIFT $\mathbf{x}(x, y, \sigma)$, a 16×16 rectangular grid is formed around the center of the point. We compute magnitudes and orientations of gradients from the smoothed image at the scale from which the keypoint is extracted. A 16×16 grid is then split to a 4×4 grid containing histograms of normalized gradients. Each histogram spans the range of $[0, 2\pi]$ in 8 levels of orientation. The four resulting windows are then rotated with respect to the computed orientation to provide invariance to rotation. Figure 2.3 demonstrates the process of extracting SIFT descriptor.

The computational complexity and dimensionality of SIFT descriptor provided the impetus for further investigations [45, 32, 85, 7, 27, 1]. Among these, PCA-SIFT [32] reduces the dimensionality of original SIFT by applying Principal Component Analysis. Although, the resulting vector is significantly compressed, PCA-SIFT adds extra overhead to compress the SIFT descriptor. SURF [7] and its modified version M-SURF [1] has been designed as a faster alternative to the SIFT method.

2.2.2 SURF Descriptor

SURF [7] is another scale and rotation invariant feature descriptor. SURF was designed to reduce the time required for extracting descriptors. Among the entire scale space constructed during the detection step, SURF handles scale invariance by computing descriptors relative to the scale s at which the feature is detected. Similar to the SIFT algorithm, the region around the keypoint to be described is rotated to provide invariance to rotation. For orientation assignment, a Haar-like wavelet filter of size $4s$ is applied to the region around the keypoint with radius of $6s$. The x and y components of the filter responses are weighted by a Gaussian function with $\sigma = 2.5s$. These weighted responses are then used to estimate the local orientation.

Afterward, we form a rotated rectangular window centered on the keypoint aligned with the orientation that is previously computed. The window is then divided into 4×4 sub-regions, each of them contains a 5×5 uniformly sampled grid. Once again, a Haar-like wavelet filter of size $2s$ is applied to each sub-region and Gaussian weighted responses (with $\sigma = 3.3s$) along the horizontal and vertical directions constitute the vector of descriptor. Thus, the descriptor contains sum of the x value, sum of the y value, sum of the absolute x value and sum of the absolute y value as follows.

$$v = [\ \textstyle\sum d_x \quad \sum d_y \quad \sum |d_x| \quad \sum |d_y|\ ] \tag{2.6}$$

2.2.3 KAZE Descriptor

As pointed out in section 2.1.3, the major contribution of KAZE lies in its non-linear scale space representation. The KAZE descriptor works in the same way as the modified version of SURF (M-SURF) that is studied in [1]. The orientation assignment also follows the SURF idea but uses first order derivatives L_x and L_y instead of Haar-like wavelet filters. In addition, the size of the rectangular grids is increased to $24s \times 24s$ and the descriptor vector is built in the form of $v = [\ \sum L_x \quad \sum L_y \quad \sum |L_x| \quad \sum |L_y|\ ]$. Each element of the summations is firstly weighted with a Gaussian kernel ($\sigma = 2.5s$) to decrease the sensitivity to noise. The descriptor vector v is then computed along 16 overlapping sub-regions of the size $9s \times 9s$ (with $2s$ padding). The 64-dimensional vector values are also weighted with $\sigma = 1.5s$ and normalized to a unit vector. KAZE has been subsequently modified by the authors in [5] to accelerate the mechanism by which descriptors are extracted.

2.2.4 Local Binary Descriptors

The idea behind Local Binary Descriptors is to build a binary stream from limited number of points in a specific sampling pattern. An influential pairwise binary test exploited in [34, 49], has become an inspiration for many researchers to further develop the idea in the context of *Local Binary Descriptors*. Of particular interest are the recent methods for fast feature point matching among which are BRISK [38], BRIEF [12], FREAK [3] and ORB [62]. These allow reliable matching of feature points at very low computational costs. Their secret sauce resides in their use of binary descriptors to represent a patch surrounding a keypoint. Matching then involves only simple binary operators which can be computed very efficiently on modern CPUs.

To supply insensitivity to noise and increase keypoint quality, Gaussian smoothing kernels are applied before conducting the pairwise tests. Generally, binary vectors are obtained by comparing intensity values of 512, 256 or 128 pairs of pixels. Therefore, a large number of potential pairs can be selected from all possible choices. The main difference between different descriptors resides in the way they select the points to be compared. Note that, the resulting stream should be used in conjunction with a suitable feature detector that is both scale and rotation invariant [3]. In [3], the binary test is formulated as:

$$\mathcal{T}(P_a) = \begin{cases} 1 & \text{if } \mathcal{G}(P_a^{r_1}) < \mathcal{G}(P_a^{r_2}) \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

where $\mathcal{G}$ indicates intensity of smoothed patch P with a specific size. a is a subset of all possible positions for a pair of patches of length 128, 256 or 512.

According to the binary property of local binary descriptors, features' similarity can be quickly evaluated by computing their Hamming distance (bitwise `XOR` plus a population count) instead of the usual Euclidean distance that brings heavier computational burden.

Different LBD methods utilize different pair-wise sampling pattern. For instance, FREAK uses circular overlapping receptive fields while in ORB and BRIEF, a random sampling pattern is preferred. In BRISK, circular receptive fields are equally distributed around the center. The term receptive field is inspired from neuro-scientists and refers to a region where light stimulates the response of retina cells.

BRIEF

The BRIEF [12] feature point descriptor uses the concept of pair-wise intensity comparison to generate a binary string describing a keypoint patch. BRIEF selects pairs' location using a random distribution over the keypoint patch. Additionally, a fixed size Gaussian kernel is applied to each pair's pixels to provide better robustness to noise.

The pattern shown in Figure 2.4 has been experimentally chosen through a series of pre-defined distributions over a patch of size $S \times S$. These distributions were evaluated based on their capability to recognize a planar target established beforehand. Five spatial distributions for intensity test are as follows.

- Uniform distribution in both x and y directions with PDF $\sim U(-\frac{S}{2}, \frac{S}{2})$.
- Gaussian distribution in both x and y directions with PDF $\sim \mathcal{G}(0, \frac{S^2}{25})$ centered on origin.
- Gaussian distribution with PDF $\sim \mathcal{G}(0, \frac{S^2}{25})$ centered on origin. The second point is distributed PDF $\sim \mathcal{G}(x_i, \frac{S^2}{100})$ centered on the first location.
- Spatial distribution of points in Cartesian coordinates converted from random locations in polar coordinates.
- Circular distribution of points in Cartesian coordinates converted from all possible locations in polar coordinates.

Among all these five combinations, experiments show that spatial distribution with $(x, y) \sim$ i.i.d. $\mathcal{G}(0, \frac{S^2}{25})$ outperforms the other patterns and exhibits a higher recognition rate.

BRISK

Binary Robust Invariant Scalable Keypoints [38] is a rotation and scale invariant method describing a keypoint by binary intensity comparisons over 512 pairs of smoothed regions. As illustrated in Figure 2.5, the positioning of the points in BRISK follows concentric circular receptive fields uniformly distributed around the center with an increasing size with respect to the distance from the center. The BRISK comes with a multi-scale detector to endure changes in scale. The detector uses FAST-9 methodology to evaluate each pixel and select the most salient keypoint in the image spatial domain as well as its neighboring octaves. As Leutenegger et al. [38] proposed, the scale pyramid is built by

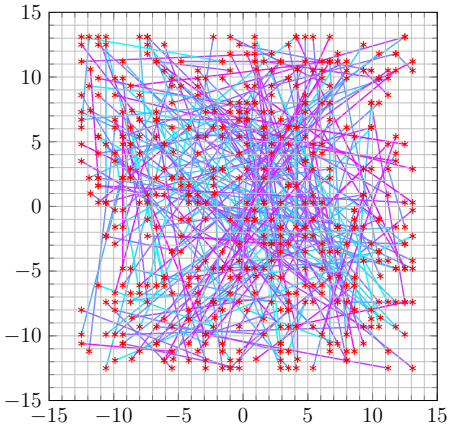


Figure 2.4: The BRIEF pattern with Gaussian distributed pairs.

down-sampling of the original image followed by three consecutive half-samplings. Once a keypoint with maximum response is located, the true sub-scale value is computed by fitting a 1D quadratic function along the scale axis.

For extracting rotation invariant keypoints, the BRISK pattern should be first rotated. All pairs are categorized into long-distance and short-distance classes. The orientation assignment is done by making use of the long-distance class while the descriptor is formed by checking the sign of pair's intensity difference sampled from the short-distance class. Thus the orientation can be formulated as a sum of local gradients.

$$g(p_i, p_j) = (p_i - p_j)\frac{I(p_j, \sigma_j) - I(p_i, \sigma_i)}{\|p_i - p_j\|^2} \tag{2.8}$$

$$\begin{pmatrix} g_x \\ g_y \end{pmatrix} = \frac{1}{L} \sum_{(p_i, p_j) \in \mathcal{L}} g(p_i, p_j) \tag{2.9}$$

$$\theta = \arctan 2(g_y, g_x) \tag{2.10}$$

where $I(p_i, \sigma_i)$ indicates smoothed value of a patch at location p_i filtered by a Gaussian kernel with σ_i, $\mathcal{L}$ is a subset of long-distance pairs with length L.

FREAK

The *Fast Retina Keypoints* FREAK is a method mainly inspired from the human retina structure. From the work of many neuro-scientists, it has been proven that the size of receptive fields in human retina exponentially increases with respect to the distance from

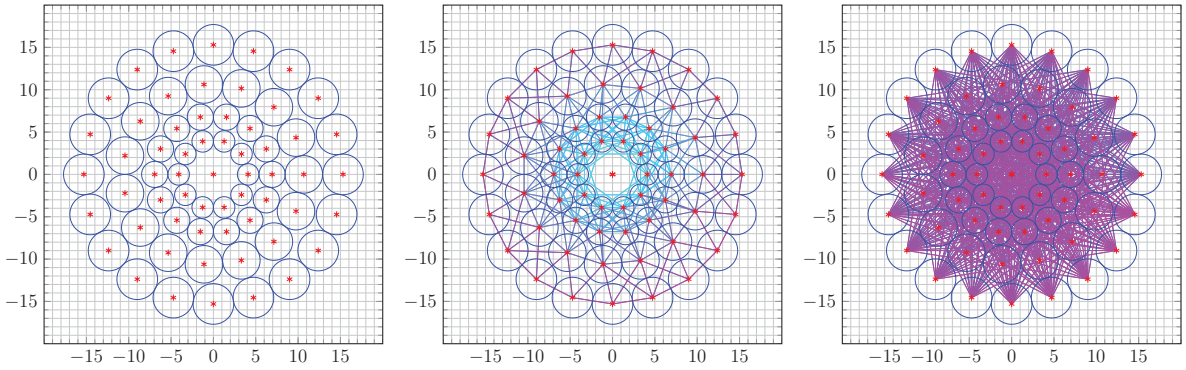


Figure 2.5: a) The BRISK sampling pattern. b) Shows 512 short-distance pairs with $\|p_i - p_j\| < 9.75s$. c) Shows 870 long-distance pairs with $\|p_i - p_j\| > 13.67s$.

the center. Moreover, as we go away from the center (fovea), fewer receptive fields are stimulated, bringing coarser information compared with the smaller but denser receptive fields in the center.

As shown in Figure 2.6, the sampling grid in FREAK method is made of a circular overlapping patterns in which the size of Gaussian kernels exponentially increases. The authors have claimed that various size of Gaussian kernels in a log-polar scale improves performance. A clear advantage of log-polar mapping is the property of being invariant to scale and rotation. It means that any variation in scale or orientation, results in pure translation of the patch in cortical coordinates [76].

ORB

Although BRIEF tolerates small departures from the local orientation, there is still an issue with larger rotation or scale transformations. The *Oriented FAST and Rotated BRIEF* (ORB), as is aptly named, extracts FAST corners in the image pyramids scored by the measure that Harris and Stephens proposed [25]. Once features with higher score are selected, the surrounding binary patterns are rotated to be aligned with the estimated orientations. As a consequence, the resulting descriptors endure larger variations in scale and rotation.

The orientation assignment follows the *Intensity Centroid* approach [59] that employs weighted sum of pixel intensities considered as raw moments of the patch. Raw moments

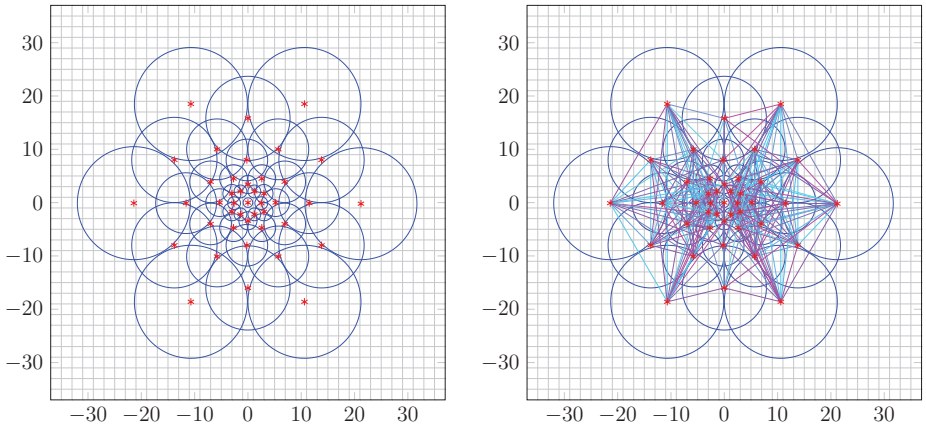


Figure 2.6: The retinal FREAK sampling pattern and the most 256 discriminant pairs depicted by lines.

are computed by

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \tag{2.11}$$

The local orientation is then simply computed using x and y components of the patch centroid $C = (\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}})$.

$$\theta = atan2(\frac{C_x}{C_y}) \tag{2.12}$$

2.3 Conclusion

An essential basis for any marker-less object detection and tracking algorithm depends on natural features that properly describe a target characteristics. In this chapter, we briefly reviewed some well known and popular methods for detecting the stable features within images. These features need to be detected under various kinds of deformation, different lighting conditions and in the presence of noise. Although SIFT, SURF and KAZE are highly robust and multi-scale algorithms, they are expensive to compute. So, according to the limited time budget for feature detection, we chose the FAST-9 algorithm. This can be implemented to run very quickly on portable devices. Among feature descriptors that are already discussed, we preferred light-weight binary descriptors to histogram of gradients methods which are much faster to compute. We experimentally found the BRIEF method, efficient enough to fit the requirements of real-time mobile applications.

Admitting the inferiority of FAST and BRIEF combination to SIFT-like methods in terms of robustness against scale and affine transformations, they can be implemented noticeably fast on low-powered CPUs. To overcome the lack of robustness and invariance of the FAST and BRIEF methods we employ an offline phase based on a classification approach described in the next chapter, to create a rich model of the target.

Chapter 3

Target Detection System Overview

Object recognition and visual tracking are two important tasks of machine vision. The past few years have seen important progress in this direction. Various approaches exploiting local appearance have been extensively studied in the literature in the context of applications such as structure from motion [84, 73, 66, 48], 2D/3D object recognition [81, 70, 51, 42, 36] and Simultaneous Localization And Mapping (SLAM) [78, 18, 19]. Most of these applications rely on keypoint recognition using the feature point detectors and descriptors discussed in Chapter 2.

This chapter describes some important strategies to match features extracted from query images against those from target images. Classic algorithms use block matching techniques to find the most similar patch to the patch of interest, either in the whole image or within a specific size window [79]. In these expensive methods, the similarity between patches is usually computed in the resolution of *Sum of Absolute Differences* (SAD), *L2* norm or *Normalized Cross-Correlation* (NCC) over pixels' intensity.

Matching algorithms are grouped into two categories namely *local descriptor-based* and *global classification-based* algorithms. In the descriptor-based methods, image information is abstracted by using local descriptors, which are extracted from the neighboring region of the most distinctive features across the image. For robust matching, many authors have stressed the importance of descriptors that are invariant to fairly large level of deformations, illumination changes and noise [63, 40, 41, 32, 7, 4]. Most of these algorithms are composed of histogram of gradient orientations and locations to distinctively describe the detected features. Despite the robustness and distinctiveness of such algorithms, the matching process requires high computational power to extract and compare local descriptors. So the current histogram of gradients methods, as they stand, do not

adequately account for the limited time budget needed for real-time applications. Therefore, the problem of fast feature matching, imposed by real-time applications, merits further investigation.

3.1 Classification-based Methods

The computational complexity and dimensionality of descriptor-based methods, especially when considering low-powered platforms, stimulated the development of new approaches [34, 35, 49, 80, 69] that take advantage of a training stage to which a great extent of computational burden at run-time is transferred. One of the earliest attempts in this direction is the work of Nayar et al. [47] in which a classifier is trained given instances of 100 3D objects. The classifier exploits image eigenvectors as orthogonal bases to represent the set of images. Principal Component Analysis and Nearest Neighbor search are used for data compression and finding the closest object manifold respectively.

A classical approach for feature-based matching was later introduced by Skrypnyk and Lowe [65] in the context of AR applications. In the offline stage of their algorithm, a sparse 3D model of a reference object is built from a set of images acquired from different viewpoints. The run-time stage of their algorithm involves extracting SIFT features from the query frame. The extracted features are then matched to those extracted at the offline process. Basically, Euclidean distance between two SIFT descriptors is used as a similarity measure and matches are established by conducting k-Nearest Neighbor (k-NN) search. A distance ratio test with $k = 2$ is employed to limit the number of tentative matches by rejecting candidates whose first to the second nearest neighbor distance is greater than a specific threshold (typically set to $0.6 - 0.8$). The large number of features representing the model and the query image, makes the matching process extremely slow. Therefore, the authors proposed approximate Best Bin First (BBF) strategy based on k-d tree [10] searching algorithm. For robust estimation of an epipolar geometric model, RANSAC algorithm [21] is employed given a set of putative matches (See Chapter 5 for more details on the RANSAC algorithm).

As we discussed in Chapter 2, a faster alternative for point matching would be benefiting from binary descriptors. So the dissimilarity score can be efficiently computed on modern CPUs using binary operations that perform much faster compared to computing Euclidean distance. In the following sections, we review some well known and popular classification frameworks for efficient matching.

3.1.1 Matching as a Classification Problem

In the classification-based methods, efficient recognition of targets is achieved by building, during an off-line phase, a rich probabilistic model of a target that will then be used to reliably detect this target in live video under a wide range of viewpoints. In this particular direction, Lepetit et al. [35], put forward *matching as classification*, as an idea for wide baseline matching problems.

In the training phase of their algorithm, N prominent feature points are extracted from the object to be trained. Under local planarity assumption, the patch around each feature point and its distorted version generated under all possible affine transformations, participate in learning the classifier. Given a patch $\mathbf{p}$ at run-time, the classifier assigns a class label in $Y(\mathbf{p}) \in \mathbf{C} = \{-1, 1, 2, ..., N\}$, determining to which class, if any, the patch $\mathbf{p}$ belongs. Here label -1 denotes that the patch does not belong to the object. Principle Component Analysis is performed to reduce the dimensionality of patches. Since Y is not directly observable, K-means clustering procedure (with $K = 20$) is used to construct a classifier $\hat{Y} : \mathbf{P} \rightarrow \mathbf{C}$ such as $P(Y \neq \hat{Y})$ is small. In order to avoid mis-classification, class labels with conditional probability $P(Y \neq \hat{Y}|c)$ greater than a specific threshold are removed from the set of classes.

Lepetit et al. further proposed to use a randomized decision tree [34] instead of PCA-based classification to select features, which tend to provide higher recognition rate at run-time. In the randomized tree approach, a reference model is built by considering random pairs of pixel locations inside a defined neighborhood around the keypoint. It is shown that, simple intensity comparisons would yield distinctive description and efficient matching at run-time. Random Fern [49] is another classification method proposed by Özuysal et al. that is perceived as an improvement over random trees bringing higher recognition rate at lower computational cost.

3.1.2 Random Ferns Classification

Feature matching for target detection has also been studied as a classification problem in the Randomized Ferns [50] method. In Fern, a simple binary test is performed to compare the intensity values of those pixels leading to binary features that are grouped together to create small binary space partitions called Ferns. By generating thousand of viewpoints, the training phase estimates the class conditional probability of each Fern for each keypoint. During the training process, each patch describing a keypoint is considered as a labeled class. To this end, a large number of corresponding views participate to learn

the appearance model of this class.

The likelihood of a new patch to correspond to a patch in the model can then be computed by assuming independence of these estimated probability distribution. Once the classifier is learned during an offline process, a set of binary tests is carried out and the classifier's output that gives maximum posterior probability over a set of classes $c_i, i := 1, ..., H$ is selected. Denoting the binary test responses as features $f_i, i := 1, ..., N$, the maximum posterior probability is evaluated as follows:

$$\hat{c}_k = \arg\max_k P(C_k|f_1, f_2, ..., f_N) \tag{3.1}$$

According to the Bayes' rule, 3.1 can equivalently expressed as the product of likelihood and prior distributions.

$$\hat{c}_k = \arg\max_k P(f_1, f_2, ..., f_N|C_k)P(C_k) \tag{3.2}$$

The joint distribution of all features' likelihood requires $H \times 2^N$ units of memory. Using a Naïve Bayes' assumption, all features' likelihood can be assumed to be conditionally independent. So equation 3.2 is reduced to the form:

$$\hat{c}_k = \arg\max_k P(C_k) \prod_{n=1}^{N} P(f_n|C_k) \tag{3.3}$$

Although, the Naïve Bayes' assumption simplifies and thus compresses the size of classifier, it causes the estimated posterior probability to drastically deviate from its true value. Özuysal et al. [50], proposed grouping features into M out of N groups of size $S = \frac{N}{M}$ called *Ferns*. In this Semi-Naïve Bayes approach, all Ferns are assumed to be conditionally independent and the joint feature probabilities within each Fern is estimated and stored. Thus we can write:

$$\hat{c}_k = \arg\max_k P(C_k) \prod_{n=1}^{M} P(F_n|C_k) \tag{3.4}$$

Bearing in mind that a full joint probability model of each Fern is required, there could still be an issue with the memory requirements of this method that exponentially grows with the size and number of Ferns.

3.1.3 Histogrammed Intensity Patches

For efficient matching at run-time, Simon Taylor and Tom Drummond introduced a fast and efficient local binary descriptor well suited for real-time target detection. In their *Histogrammed Intensity Patches* (HIP) method [69, 71], the local appearance of each pixel is extracted from a rectangular patch of size 8×8 pixels which is itself sampled from a 16×16 patch. For efficiency considerations, they suggested to use a sampling step equal to 2 pixels. After this sampling step, the patch is rotated in order to be aligned with the local orientation of the patch. To extract new position and intensity of pixels, bi-linear interpolation is used. The proximate orientation assignment scheme proposed in [71], uses the same 16 pixels ring as was used in FAST detection algorithm described in section 2.1.4. The coarse orientation is nothing more than a weighted sum of intensity differences over eight symmetric pairs relative to the center.

The 64 array of the rotated patch is then quantized into 5 intensity levels and stored in the form of a binary descriptor of size 64×5. The quantization is done by calculating mean and standard deviation over 64 pixels. Assuming Gaussian distribution, intensity bounds are computed as follows:

$$B_i = \mu + \Phi^{-1}(\frac{i}{5})\sigma \tag{3.5}$$

$$\Phi(x) = \frac{1}{2\pi}\int_{-\infty}^{x} \exp^{-\frac{t^2}{2}}\, dt \tag{3.6}$$

in which B_i indicates upper bounds for five intensity levels and $\Phi(x)$ is the inverse Gaussian distribution function.

Figure 3.1 and 3.2 briefly summarize the Histogram Intensity Patch approach to describe local appearances. Note that these fast-to-compute binary descriptors do not bring high distinctiveness and robustness to illumination, scale and transformation. To overcome this shortcoming, an offline process is employed to build rich descriptors for a target. This avoids extracting expensive descriptor computation at run-time. The authors claimed that it can reduce the number of required features by a factor of around 15.

Offline Training

In the Histogrammed Intensity Patches method [69, 71], Taylor et al. introduced the idea of grouping the artificially generated random views into viewpoint bins. The model is built by first computing coarse histograms of the intensities of neighboring pixels around

Figure 3.1: The left image is 16×16 pixel original interest point appearance. The middle shows 8×8 down-sampled extracted patch. The right shows aligned rotated patch.

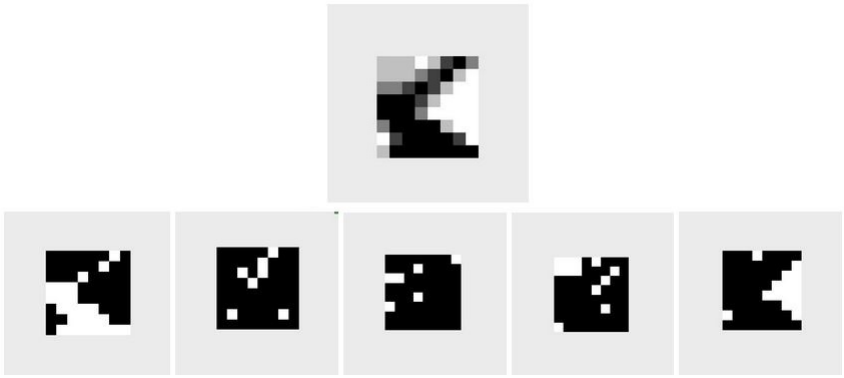


Figure 3.2: The above image shows 8×8 pixels quantized patch. The below images show 5 levels of quantized patch. White pixels represent location of pixels of a particular color.

a keypoint for each viewpoint bin. Once computed, each histogram is averaged and binarized. Given a pair of HIP models, a dissimilarity score is computed by identifying bins that are rarely hit with the idea that corresponding patches in the live view should have a small number of pixel values falling into these rare bits. Figure 3.3 illustrates the matching process proposed by Taylor et al.

In the matching step, choosing a proper threshold plays an important role. For instance, setting the threshold too low results in extremely high distinctiveness and mis-

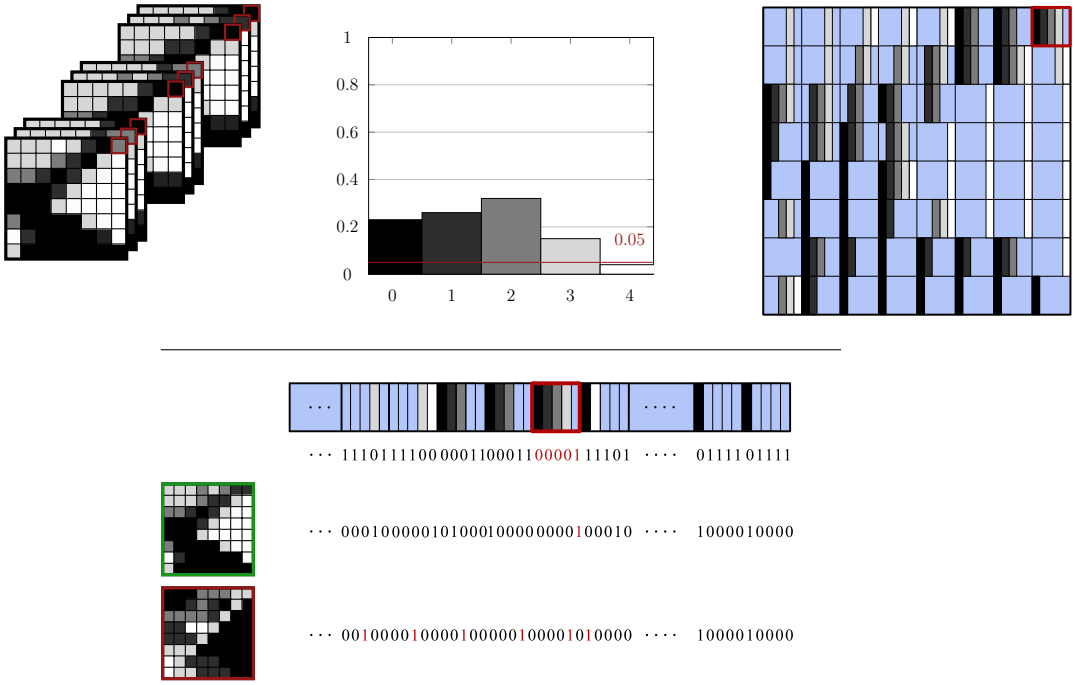


Figure 3.3: Taylor's method for building a HIP model. Patches from different viewpoints falling within an approximate location constitute a class. A sample histogram of a pixel is shown as a red box. Histograms of five intensity levels for each class of the 8×8 patches are similarly computed. After thresholding all histograms, samples with frequency less than 0.05 are considered as rare bits and stored with 1 in a binary stream. The computed HIP's binary stream is then used to compare with different patches. Correct matches (green) and wrong ones (red) are determined according to the number of bits that hit one of the HIP's rare bit.

matching many inliers as outliers. It is also worth mentioning that number of histogram levels is dependent on choosing a proper threshold. A detailed analysis concerning the choice of the threshold, number of quantization levels and sampling step can be found in [71].

For clustering histograms, pixels are selected in spatial and angular vicinity of a specific keypoint. Considering a 2-pixel error bound for position and 10 degrees for orientation suffices for ensuring constituents' similarity. The method continues to add the most populated clusters in sequence until coming up with a database consisting of a sufficient number of components representing a target. The database stores clusters' center position picked from the reference frame along with the corresponding HIP model

and its local orientation.

At run-time, features and descriptors are extracted from the query image and from its pyramids to ensure collecting sufficient number of stable features supporting greater range of scale levels. Down-sampling the original image can also reduce the effect of artifacts induced by motion blur and noise. To further accelerate the matching process at run-time, an indexing approach is used to avoid exhaustive search through the entire set of features in the database. To this end, *Locality Sensitive Hashing* strategy firstly introduced in [31] is used, in order to assign an index number for each feature by computing a hash function on the image region around that feature point. So at run-time, features are only compared against those with the same index number. Robustness of the matching is ensured by adding large portion of common indices observed for each patch. Figure 3.4 shows the arrangement of samples for two index patterns used in [71] with 5 and 13 samples.

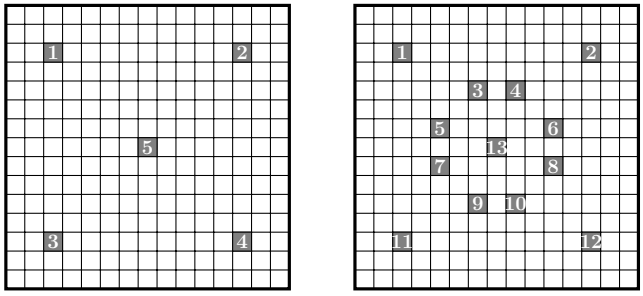


Figure 3.4: Two rotation-aware hash patterns with 5 and 13 samples.

View-set Generation

The core benefit of exploiting local binary descriptors for image registration is their low computational burden and ease of implementation. However, these methods do not inherently provide invariance to various visual aspects such as scale, orientation, affine transformation and changes in illumination. To counterbalance this drawback, the Taylor's approach takes advantages of an offline training stage to which a great deal of computational burden normally performed at run-time is transferred.

To cover all possible visual variations, several hundreds of views are artificially generated from a single target view. As in [69], generated views with relatively similar deformation parameters are considered to fall in the same viewpoint bin. The viewpoint bins are uniformly distributed in accordance with corresponding ranges of orientation

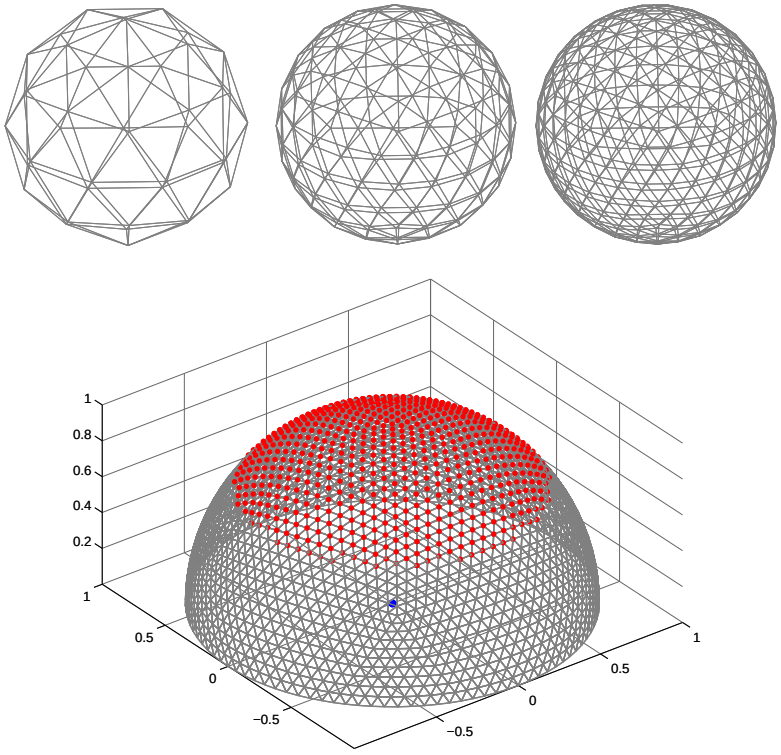


Figure 3.5: Shows three subdivision levels of a polyhedron and resulting viewpoint distribution with up to 40 degrees out-of-plane rotation

angles and scale factor. As discussed, assigning a coarse orientation to local patches, yield distributing all bins only based on their skew angles and scale factor ignoring the camera-axis rotation. In consequence, each bin covers random roll angle spanned from 0 to 360 degrees with random skew and scale parameters within the range of that particular bin. Generating uniformly distributed viewpoints needs uniform 2D space sampling. As shown in figure 3.5, Hinterstoisser [28] proposed to recursively subdivide faces of a polyhedron and normalize each produced face until sufficient number of vertices has emerged.

Moreover, to extract stable feature points, the artificial views are smoothed with a small and random Gaussian blur. This simulates the effect of motion blur that comes out into camera output at run-time.

3.1.4 Trained Binary Descriptors

Although, the above explained methods exhibit fast and reliable matching at the online stage, they suffer from a high training time and/or memory consumption. Being inspired from the training method of [69], Akhoury et al. [2] proposed a similar framework which reduces the training time more than 80% as compared to [69, 71].

Under the assumption of planar targets or at least planarity of local regions around feature points, most classification-based methods [34, 49, 69, 71] apply affine transformations to generate artificial views. However, the time reduction achieved in [2], is rooted in applying more representative transformations to synthesize the viewset images instead of affine transformations. This is done by using a heuristic approximation of internal camera parameters to generate perspective transformations that results in less images required per viewpoint bins.

In the proposed framework, feature aggregation is done by taking the repeatability of feature points in each viewpoint bin into account. Unlike the HIP method, in which model descriptors are aggregated by identifying rarely hit bits, most binary descriptors like BRIEF can be aggregated by applying a majority vote on each bit of the descriptor.

3.2 Conclusion

This chapter briefly reviewed some of the important works related to the problem of real-time planar object recognition. Different matching algorithms were outlined that can be grouped into local descriptor-based and global classification-based classes. The goal of local descriptor-based approaches is to detect and describe distinctive features using the local regions around each feature. Although these algorithms are highly robust against variations in scale, rotation and illumination, their computational complexity is a drawback for real-time mobile applications. On the other hand, classification-based approaches are computationally efficient but requiring an offline training of the objects. Thereafter, the matching process can be efficiently performed given the model of each target at run-time.

In this chapter, we also studied some popular classification strategies. The Fern and Taylor's approach are explained for matching features extracted from input image against those from a reference image. Fern is an extremely memory consuming method, HIP is therefore preferred to provide correspondences required for solving the *Geometric Pose Estimation* problem presented in the next chapter.

Chapter 4

The Geometric Pose Estimation

Camera Pose Estimation is widely perceived as a key element in numerous computer vision applications including augmented reality [81, 51, 58, 78], automatic panoramic image stitching [11] and robotics [18, 64, 72]. The camera pose estimation problem, which is well studied both in photogrammetry [13, 23] and computer vision literature [26, 53], is the problem of recovering relative 3D positioning of a camera with respect to 3D world coordinates. Whether dealing with a moving camera, a moving scene relative to a fixed camera, or both, the camera pose can be described by determining the relative three-dimensional position and orientation of the camera.

For parameterizing the position and orientation of a camera, usually represented by a rotation and translation matrix $[R|t]$, an in-depth study of a pinhole camera *intrinsic* and *extrinsic* parameters is needed. Various multisensory approaches [39, 8] exist but, algebraic methods that are based on a single visual sensor (CCD camera) are of particular interest. These methods require a set of 2D points in the image plane and the coordinates of the corresponding 3D scene points.

In the case of planar targets, a parametric model that describes a camera pose can be formulated by a homography relation mapping two views of a planar object. For identifying a 2D homographic transformation between two views, the minimal subset size for correspondences is four. This transformation is given by directly solving a system of linear equations. The problem of retrieving position and orientation of internally calibrated camera using n point correspondences, is referred to as the *Perspective-n-Point* (PnP) problem. Different analytical and iterative solutions have been proposed by computer vision scientists for solving this problem. For instance, closed-form solutions have been suggested that include solving a set of polynomial equations [21, 24, 29, 54,

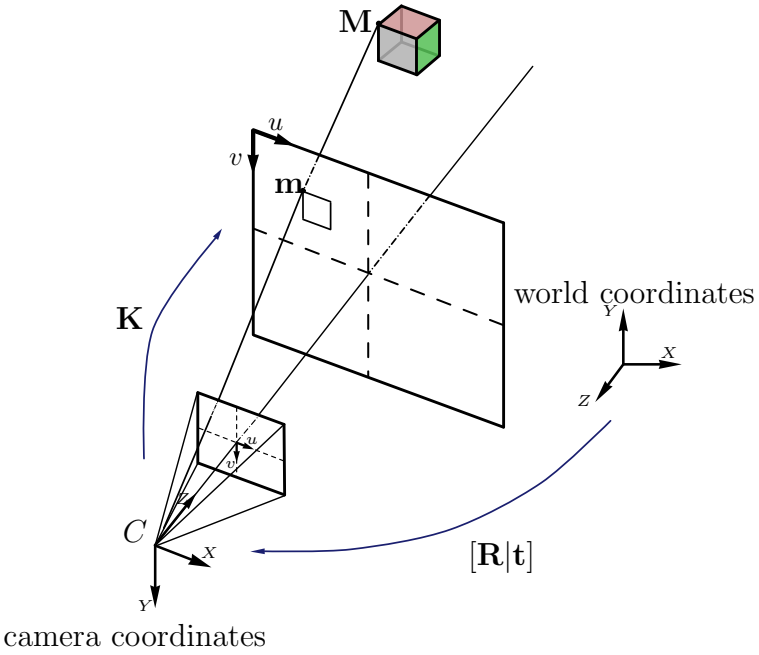


Figure 4.1: Perspective model of a pinhole camera.

77, 20, 37]. This chapter briefly reviews some of the most popular approaches for the camera pose estimation.

4.1 The Perspective Projection

A realistic model for standard pinhole camera model can be formulated as a perspective projection. As illustrated in Figure 4.1, the perspective transformation is a $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ mapping from a point $\mathbf{M} = [X, Y, Z, 1]^T$ represented in homogeneous 3D coordinates to a pixel $\mathbf{m} = [u, v, 1]^T$ represented in 2D homogeneous coordinates, such that:

$$\mathbf{m} = P\mathbf{M} \tag{4.1}$$

Where P is a 3×4 matrix, usually called projection matrix with 11 degrees of freedom thus only described up to scale. These 11 camera parameters can be further separated to *intrinsic* and *extrinsic* parameters with 5 and 6 degrees of freedom respectively. So the Equation 4.2 can be expressed as:

$$\mathbf{m} = K[R|\mathbf{t}]\mathbf{M} \tag{4.2}$$

4.1.1 Intrinsic Parameters

The camera intrinsic parameters also known as the camera matrix $K_{3\times3}$, is used to map 3D camera coordinates to 2D image pixels. Matrix K is formed by 5 camera internal parameters as follows:

$$K = \begin{pmatrix} k_u f & s & c_u \\ 0 & k_v f & c_v \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

The terms $k_u f$ and $k_v f$ are focal length represented in pixels, where k_u and k_v are scale factors proportional to pixel density per unit distance. c_u and c_v are camera principal points that are ideally taken to be at the center of image plane. s is a small non-zero skew parameter between u and v axes. It can be usually approximated by zero when image axes are considered to be perpendicular. Under a reasonable assumption proposed in [26], the 5 internal parameters can be reduced to 3 if one assumes $s = 0$ and pixels are square. This is a common case in modern cameras resulting in $k_u f$ and $k_v f$ to be equal.

4.1.2 Extrinsic Parameters

A 3D point represented in world coordinates is translated to 3D camera coordinates by the camera extrinsic parameters. In other words, extrinsic parameters explain the camera position and orientation (pose) in world coordinates. It can be denoted by 3×4 matrix $[\mathbf{R}|\mathbf{t}]$ comprising of a rotation matrix $\mathbf{R}_{3\times3}$ and translation matrix $\mathbf{t}_{3\times1}$:

$$[\mathbf{R}|\mathbf{t}] = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \tag{4.4}$$

Referring to Figure 4.1, the transformation from a 3D point in the camera coordinates (M_w) to the world coordinates (M_c) is expressed as:

$$M_w = \mathbf{R}^{-1}(M_c - \mathbf{t}) = \mathbf{R}^{-1}M_c - \mathbf{R}^{-1}\mathbf{t} \tag{4.5}$$

In which, $-\mathbf{R}^{-1}\mathbf{t}$ is the position of camera center in world coordinates. We can equivalently write:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} X_w + t_1 \\ Y_w + t_2 \\ Z_w + t_3 \end{pmatrix} \tag{4.6}$$

4.1.3 Distortion Coefficients

Although, determining the intrinsic and extrinsic parameters of the camera would usually suffice for estimating the camera pose, in some cases, the camera lens causes a distortion that must not be neglected. This distortion can be compensated by 2D displacement terms called *radial* distortion and *decentering* distortion. Let $\mathbf{x}_u$ and $\mathbf{x}_d$ be un-distorted and distorted image point respectively. We can write:

$$\mathbf{x}_u = \mathbf{x}_d - \mathbf{d}_{radial} - \mathbf{d}_{decenter} \tag{4.7}$$

While the decentering distortion is usually ignored, the radial distortion can be determined by a polynomial expression as follows:

$$\mathbf{d}_{radial} = 1 + k_1 r^2 + k_2 r^4 + \dots \tag{4.8}$$

Where r is the radial distance from the center of image plane. The process of estimating radial coefficients and camera intrinsic parameters is called *camera calibration* process.

4.2 The Homography Estimation

A homography or projection transformation is a plane-to-plane ($\mathbb{P}^2 \rightarrow \mathbb{P}^2$) relation in a projective space. It is algebraically defined by a non-singular 3×3 matrix H that maps two views of a planar object. Let $\mathbf{X} = [X, Y, Z]^T$ and $\mathbf{x} = [x, y, 1]^T$ be respectively the homogeneous coordinates of an identical point on a reference plane and its projection that is seen from a different view; the 2D homography transformation is described as:

$$\mathbf{X} = H\mathbf{x} \tag{4.9}$$

The 2D points in the image can be represented by $(u, v) = (X/Z, Y/Z)$. Unlike the perspective transformation which has 9 degrees of freedom, this equality that is up to a scale factor, has 8 degrees of freedom and consequently can be computed from four point correspondences.

According to Hartley and Zisserman [26] , the homography is an invertible and line preserving mapping such that, any set of three points is collinear if and only if the corresponding set in the other plane preserves collinearity too. Note that this property is not true for the perspective transformation as the inverse transformation P^{-1}, maps a point from camera coordinates to a ray containing that point expressed in the world coordinates.

4.2.1 Direct Linear Transformation (DLT)

A typical and perhaps the simplest approach for retrieving the homography relation with 8 degrees of freedom is the *Direct Linear Transformation*. This method, which was first introduced in photogrammetry [68] and then used in computer vision communities [26], treats the pose constraints as a system of linear equations. This system can be solved by minimizing an algebraic distance. DLT is used in camera pose estimation especially when intrinsic parameters of the camera are not available. The homogeneous system of equations corresponding to the homography relation can be solved by posing:

$$X_i \times Hx_i = 0 \tag{4.10}$$

in which H is computed using the homogeneous source points $(x_i, y_i, 1)$ and target points (X_i, Y_i). This equation can then be rewritten as:

$$\begin{pmatrix} 0 & 0 & 0 & -x_1 & -x_2 & -1 & Y_1x_1 & Y_1y_1 & Y_1 \\ x_1 & x_2 & 1 & 0 & 0 & 0 & -X_1x_1 & -X_1y_1 & -X_1 \\ -Y_1x_1 & -Y_1y_1 & -Y_1 & X_1x_1 & X_1y_1 & X_1 & 0 & 0 & 0 \\ & & & & \vdots & & & & \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = \mathbf{0} \tag{4.11}$$

which results in equations of the form $\mathbf{A_i h} = \mathbf{0}$, with $\mathbf{A_i}$ being the lines of the left matrix and $\mathbf{h}$ being a 9×1 vector made of the entries of the homography matrix. The above Equation 4.11 can be expressed in a vector-form as:

$$\begin{pmatrix} \mathbf{0^T} & -\mathbf{x_i}^\mathbf{T} & Y_i\mathbf{x_i}^\mathbf{T} \\ \mathbf{x_i}^\mathbf{T} & \mathbf{0^T} & -X_i\mathbf{x_i}^\mathbf{T} \\ -Y_i\mathbf{x_i}^\mathbf{T} & X_i\mathbf{x_i}^\mathbf{T} & \mathbf{0^T} \end{pmatrix} \mathbf{h} = \mathbf{0} \tag{4.12}$$

One common technique to find non trivial solution of this system of equations is to use *Singular Value Decomposition* (SVD). This technique is particularly useful when more point correspondences are available, in which case SVD will identify the optimal least-square algebraic solution. However, other more computationally expensive (and iterative) approaches could also be used to obtain a geometrically optimal solution [26]. To find least square solution for the system of equations, SVD decomposes $\mathbf{A}$ to:

$$\mathbf{A} = \mathcal{U}\mathbf{\Sigma}\mathcal{V}^\mathbf{T} \tag{4.13}$$

where $\mathcal{U}$ and $\mathcal{V}$ are the left and right singular vectors of A respectively and Σ is a diagonal matrix containing singular values of A in a descending order. To avoid trivial solution, the norm constraint ($\|\mathbf{h}\| = \mathbf{1}$) is also imposed. Thus the optimal solution for $\mathbf{A_i h} = \mathbf{0}$ is given by the right null vector of A which is the last column of $\mathcal{V}$. Note that the right null vector of A corresponds to the smallest eigenvalue thus minimizes:

$$\mathbf{h} = \min_{\mathbf{h}} \frac{\|\mathbf{Ah}\|}{\|\mathbf{h}\|} = \min_{\mathbf{h}} \|\mathbf{Ah}\| \tag{4.14}$$

Although [26] gives specific circumstances, under which 11 d.o.f projection matrix can be reduced to 9, DLT still exhibits poor performance and low stability when the camera intrinsic parameters are already known. This is due to an over-parameterization of the problem [36].

4.2.2 Perspective-n-Point (PnP)

Recovering 8-DOF projective transformation of a (projective) plane that maps 3D world coordinates to 2D image plane requires at least 4 set of 2D correspondences. The solution given by directly solving the system of linear equations does not always suit practical considerations. For instance, DLT suffers from over-parameterization problem when dealing with an internally calibrated camera. The *Perspective-n-Point* (PnP), which has been originally developed for camera calibration problems, is also a solution for the pose estimation. There are several methods to solve the PnP problem for $n \geq 3$, where n is the number of correspondences returned by matching algorithms. Some of these methods provide a closed-from solution for only a specific n and some others can handle any arbitrary n.

Among different methods capable of explicitly solving the PnP problem, those who bring stable and accurate solution with low complexity are particularly interesting. In this section, we will review a non-iterative solution for the PnP problem.

P3P Rigid Body Transformation

One of the earliest attempts on PnP problem has been made in 1981 in [21] to provide an explicit solution for the P3P by forming a bi-quadratic polynomial that gives up to four solutions for $n = 3$. They also provided a unique solution for $n = 4$, for the case of all points lying on a common plane.

In order to retrieve a rigid body transformation which describes the camera pose, three pairs of 2D/3D correspondences would usually suffice. Considering a contaminated

set of points, sampling 3 pairs instead of 4, leads to a lower probability of choosing a degenerate sample. However, this approach requires a prior knowledge of the camera intrinsic parameters. Following the approach proposed by Fischler and Bolles [21], the transformation is estimated by firstly computing the distance between 3D object points and the camera center C. In Figure 4.2, let R_a, R_b and R_c be the length of three legs that connect the camera center to the 3D object points A, B and C respectively. Let a, b and c be the projection of these points on the image plane, the goal is to determine R_a, R_b and R_c.

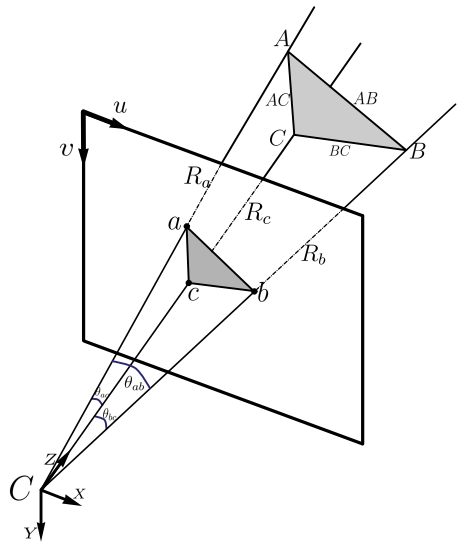


Figure 4.2: Geometric view of P3P

By applying basic geometry, one can determine the length of three sides and interior angles of $\triangle ABC$ from 3D object points. Posing the law of cosines yields:

$$\begin{aligned} AB^2 &= R_a^2 + R_b^2 - 2R_aR_b\cos(\theta_{ab}) \\ AC^2 &= R_a^2 + R_c^2 - 2R_aR_c\cos(\theta_{ac}) \\ BC^2 &= R_b^2 + R_c^2 - 2R_bR_c\cos(\theta_{bc}) \end{aligned} \tag{4.15}$$

The equation 4.15 comprises of three second degree polynomials, therefore it brings up to eight unique solutions. However, since the polynomials only contain second degree and constant terms, it has a maximum of four positive solutions. If we define $x = \frac{R_a}{R_b}$ and

$y = \frac{R_c}{R_a}$, it is shown in [21] that a biquadratic (aka quartic) polynomial can be derived expressing 4.15.

4.3 Distance Evaluation

The problem of finding a homography given a set of correspondences is solved by minimizing a specific cost function. The overall correspondences cost for a hypothesized model is minimized either through an iterative scheme or hypothesize-and-verify scheme. Choosing a proper cost function directly impacts on the accuracy of solution and the speed of convergence. Some common cost functions are described as follows:

4.3.1 The Algebraic error

The linear function that is minimized through DLT method by the SVD analysis is the sum of algebraic distances. Since minimizing the algebraic distance has a closed form solution, it leads to lower complexity and ease of implementation. However, algebraic distances can not be interpreted as geometric parameters. However, the minimizing algebraic error does not lead to accurate solution; a normalization step is proposed in [26] that improves the accuracy but at the cost of losing closed form solution. In the case of homography estimation, the sum of algebraic distances that is minimized through DLT method is as follows:

$$\sum_i = d_{alg}(X_i, Hx_i)^2 = \|\mathbf{Ah}\|^2 = \|\epsilon\|^2 \tag{4.16}$$

4.3.2 The Geometric error

A more geometrically meaningful alternative to the algebraic error is *Geometric error*. The geometric error is a quantity that is defined by the Euclidean distance between a measured point and a projected point on the image.

By denoting a set of correspondences between two planes that are mapped with a homography matrix as $x_i \xrightarrow{H} X_i$, a transfer error in the second plane is defined as follows:

$$\sum_i = d(X_i, Hx_i)^2 \tag{4.17}$$

Where $d(.,.)$ is Euclidean distance between two 2D points in the image plane.

The geometric error can be more accurately estimated by considering the Euclidean distance in both images. Therefore, in addition to the forward error term of 4.17, a backward term computed using H^{-1} is also taken into account. Then we can write:

$$\sum_i d^2 = d(X_i, Hx_i)^2 + d(x_i, H^{-1}X_i)^2 \tag{4.18}$$

So in the process of computing homography the cost function which is subject to be minimized is the sum of geometric errors. In contrast to the algebraic error that could be minimized through a non-iterative process, the minimization of geometric cost functions provides more accurate estimates but through an iterative process.

4.3.3 The re-projection error

Assuming a set of correspondences $x_i \xrightarrow{H} X_i$, one can find an estimate of the homography $\hat{H}$ by minimizing the sum of re-projection errors. A corrected set of correspondences ($\hat{x}_i \xrightarrow{\hat{H}} \hat{X}_i$) is established that returns pairs of perfectly matched points by the forward and backward projections.

$$\sum_i d^2 = d(X_i, \hat{X}_i)^2 + d(x_i, \hat{x}_i)^2 \tag{4.19}$$

Indeed, the re-projection error is a sum of distances in both images between the measured points and projections of a world point lying on a global plane. Although the minimization of this error function results in a set of perfectly matched correspondences, it requires an expensive iterative process.

4.4 Conclusion

In this chapter we have presented a number of issues that directly pertain to the problem of camera pose parameterization based on 2D/3D correspondences. Firstly, we have sketched a realistic model for the geometric pose of a CCD camera as a perspective projection. We have also provided a deeper study of the perspective projection by analyzing intrinsic and extrinsic parameters of a pinhole camera.

As an essential part of the planar target recognition framework, the more specific case of 2D plane-to-plane transformation has been addressed by introducing the most commonly used algorithm for estimating the homography matrix. Although iterative pose estimation algorithms return more accurate estimates, in this chapter we focused

on non-iterative approaches that do not require expensive iterative process minimizing projection error.

The next chapter will introduce the concept of robust model estimation in which non-iterative homography estimation solutions are used to obtain good camera pose solution from a contaminated set of matches.

Chapter 5

Robust Model Estimation

Robust estimation of model parameters constitutes an essential tool in many fields of science. Considering the wide range of applications dealing with the problem of parameter estimation, a great deal of effort has been devoted by many researchers to tackle this problem.

There is now substantial body of research on robustly retrieving model parameters in the presence of outliers [6, 26, 21, 15, 44, 75, 74, 56]. In the camera pose estimation problem, the number of correspondences is usually considerably larger than the number of model constraints. A traditional approach to address this overdetermined problem is to use *Least Square* (LS) methods. The LS method is responsible for adjusting the model parameters such that the sum of square residuals is minimized. The *Maximum Likelihood Estimation* (MLE) is an alternative solution that relies on a statistical knowledge of residuals' distribution.

The matching approaches presented in Chapter 2 produce a large set of putative matches. Depending on the quality of the keypoint descriptors and on the level of noise associated with this set, it is likely to be contaminated by a more or less large number of false matches (outliers). This is where robust model estimation comes into play. A major drawback of the classic regression techniques is their zero breakdown point that makes them highly sensitive to a single outlier. However, there exists a variant of LS method named *Least-Median of Squares* (LMS) that withstands a higher fraction of outliers (50%). All these methods exhibit poor stability when the data set follows multiple dominant distributions. This is a common case in camera pose estimation problems especially when the camera moves toward a scene including independent moving objects [67]. The *RANdom SAmpling Consensus* (RANSAC) is a powerful approach introduced

by Fischler and Bolles (1981) [21] that has been extensively applied to many computer vision problems.

In the case of planar targets, the model that has to be estimated is a two-view homography between the current frame and the reference target. This model is then verified against all data points in the set in order to compute its support that is the number of data points that are in agreement with the hypothesized model. By repeating several times this process with different random sampling of the data set, a solution with a strong support should be found. In this chapter we will investigate technical aspects of different RANSAC-based algorithms.

5.1 Maximum Likelihood Estimation (MLE)

The Maximum Likelihood Estimation relies on a prior knowledge of single distribution of the observations. In geometric pose estimation, the distribution of the observed data is not always available or does not follow a single distribution. However, understanding of MLE theory is still beneficial because of its relevance to model fitness in the context of statistical tests such as T-test, Chi-squared and G-test.

Given a set of observations (e.g. set of correspondences between two views), the Maximum Likelihood determines, from a parametric model $f(.|\theta)$, the vector of parameters θ that exhibits the highest degree of consistency with the observed data. By denoting the observations as $\mathbf{x} = (x_1, x_2, ..., x_n)$, R. A. Fisher (1912) [6] introduced a likelihood function as the joint probability distribution of $x_1, x_2, ..., x_n$:

$$\mathcal{L}(\theta|\mathbf{x}) = f(\mathbf{x}|\theta) = f(x_1, x_2, ..., x_n|\theta) \tag{5.1}$$

Assuming that the observed samples are independent and identically distributed, one can simplify the likelihood function 5.1 as the product of all likelihoods corresponding to each sample.

$$f(x_1, x_2, ..., x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta) \tag{5.2}$$

Therefore, any estimate of model parameters $\hat{\theta}$ that maximizes $f(x_1, x_2, ..., x_n|\theta)$ would be the maximum likelihood solution. We can also go further by revisiting Bayes' rule expressing that the posterior probability is proportional to the product of the prior probability and likelihood.

$$p(\theta|x_1, x_2, ..., x_n) \propto f(x_1, x_2, ..., x_n|\theta) \times p(\theta) \tag{5.3}$$

It is important to note that the maximum likelihood is similar to the maximum a posterior estimation if $p(\theta)$ is assumed to be uniformly distributed.

5.2 RANdom SAmpling Consensus (RANSAC)

A common robust approach to deal with contaminated samples with outliers, is utilizing *RANdom SAmpling Consensus* (RANSAC) scheme firstly introduced by Fischler and Bolles (1981). This *hypothesize-and-verify* approach randomly selects a minimal set of samples and estimates the pose parameters until achieving consensus with the strongest support for the parameters. In the hypothesize step, unlike the MLE and LS-like methods, RANSAC incorporates a minimal set of participating samples according to the fact that the probability of selecting a contaminated set is exponentially decreased by the sample size [21]. Once the model is generated, the hypothesis must be verified against the subset of all available samples $\mathcal{U}_N$. The hypothesis quality is measured by its *support* which is the number of samples from $\mathcal{U}_N$ with error lying within a specified threshold.

The RANSAC loop is designed to keep iterating until assuring, with some level of confidence η_0, that a set of uncontaminated samples is already selected. So the upper bound, (k) for the number of RANSAC iterations can be computed at each iteration by accounting for the best support found so far.

Taking desired level of confidence η_0 into account, one can determine maximum number of required iterations k_{max} to guarantee this level of confidence. The probability of selecting n pairs, all as inliers is w^n where w is the inlier rate of a data set. Consequently the probability of selecting n pairs with at least one outlier is $1 - w^n$. Thus, $(1 - w^n)^k$ becomes the probability of never selecting outlier-free samples in k iterations and this needs to be less than $1 - \eta_0$. So at each iteration, k is updated as follows:

$$k \leq \frac{\log(1 - \eta_0)}{\log(1 - w^n)} \tag{5.4}$$

In practice, to obtain *a priori* knowledge of w, one can estimate it by its lower bound using the support of the best hypothesis found so far. In the RANSAC scheme, this termination constraint is known as the *maximality* constraint. The Algorithm 1 attempts to briefly explain the classical RANSAC steps.

Despite the simplicity and efficiency of the original RANSAC algorithm even when samples are significantly contaminated, it still suffers from serious drawbacks that impact on the effectiveness of the target matching. Since determining the optimal number of

Algorithm 1 The classic RANSAC

Require: $\mathcal{U}_N$, η_0

Initialize $I_{best} = 0$ and k_{MAX} with maximum number of iterations.
for $k = 0$ to k_{MAX} **do**
Randomly select a minimal subset from $\mathcal{U}_N$.
$H \leftarrow$ Generate a hypothesis using minimal set of correspondences.
$I_k \leftarrow$ Evaluate the current hypothesis support by counting the number of inliers.
if $I_k < I_{best}$ **then**
$I_{best} = I_k$
Update the hypothesis with the the strongest support.
$k_{MAX} \leftarrow$ Update the termination bound using eq. 5.4
end if
end for

RANSAC iterations requires prior information about the level of contamination, we rely on an estimate of outlier's rate by accounting for the support of the best hypothesis found so far. This rough estimate sometimes results in considerably large number of iterations yielding a long execution time for model generation and verification. The second critical issue is about setting a proper distance threshold for the inliers' error bound, i.e., setting the threshold too low may result in an infinite or long lasting loop and setting it too high may cause the loop to end up with a completely nonsense model. Moreover, even with a slightly contaminated set, RANSAC fails to come up with an optimal solution in some degenerate configurations. Figure 5.1 graphically illustrates three cases of RANSAC failure in a 2D line fitting problem given a set of points lying on a plane. An example of a degenerate configuration in the problem of homography estimation could be sampling co-linear set of points to estimate parameters a plane.

In the following sections, to account for these boundaries we go over some RANSAC-flavors designed to overcome these drawbacks.

5.2.1 PROSAC

The *PROgressive SAmple Consensus* (PROSAC) [15] is a variant of RANSAC that aims at early identification of a hypothesis with strong support. It uses a sampling approach in which samples are selected from a smaller subset of correspondences. The size of this subset progressively increases until it eventually coincides with the full set $\mathcal{U}_N$. PROSAC

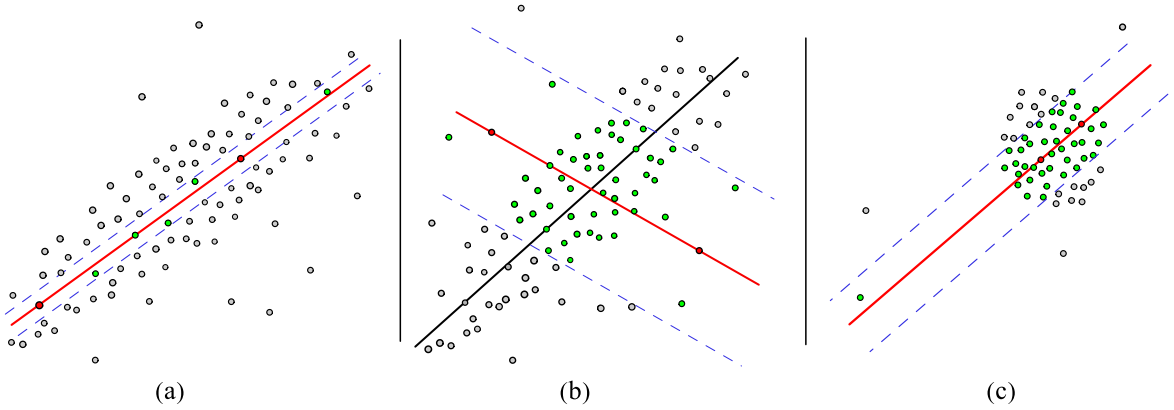


Figure 5.1: Examples of RANSAC deficiency. a) Setting the distance threshold too low limits the number of inliers (support), thus prevents RANSAC from convergence to the solution. b) Setting the threshold too high results in a wrong fit with a strong support. c) A geometric degenerate configuration. Sampling from a dense cloud of points yields an infinite number of models consistent with the set.

assumes that samples with higher quality are more likely to generate a hypothesis with a strong support. In the context of point matches, the quality of samples is measured by the similarity score of the two matched points. Compared to the typical RANSAC method, PROSAC greatly reduces the number of attempts for the best support hypothesis, in the sense that it is more likely to choose outlier-free samples from the subset of correspondences with higher scores.

Let $\{\mathcal{M}_i\}_{i=1}^{T_N}$ be a subset of all samples with size m out of all correspondences $\mathcal{M}_i \subset \mathcal{U}_N$ and denote the size of this subset by T_N. In PROSAC we want to find $\mathcal{U}_n \subset \mathcal{U}_N$ that is the subset of n ordered points from $\mathcal{U}_N$. If we define T_n to be the average number of possible samples that only belong to $\mathcal{U}_n$,

$$T_n = T_N \frac{\binom{n}{m}}{\binom{N}{m}}, \tag{5.5}$$

then T_n can be recursively computed by

$$T_{n+1} = T_n \frac{n+1}{n+1-m} \tag{5.6}$$

where T_{n+1} indicates the number of samples from $\mathcal{U}_{n+1}$ only. Thus, one can conclude that there are $T_{n+1} - T_n$ samples containing data point u_{n+1}. Therefore, the non-uniform

random sampling is done by drawing $m-1$ random samples from $\mathcal{U}_n$ in addition to the point u_{n+1}. To avoid non-integer values, we reformulate 5.6 by defining $T'_m = 1$ and

$$T'_{n+1} = T'_n + \lceil T_{n+1} - T_n \rceil \tag{5.7}$$

So at each iteration t, samples are drawn from $\mathcal{M}_t$ which is determined by a growth function $g(t)$ as follows.

$$\mathcal{M}_t = \{u_{g(t)}\} \cup \mathcal{M}'_t \tag{5.8}$$

$$g(t) = min\{n : T'_n \geq t\} \tag{5.9}$$

The PROSAC sampling procedure is outlined in the Algorithm 2.

Algorithm 2 The PROSAC algorithm

Require: Set of ordered points $\mathcal{U}_N$, k_{MAX} and η_0

Initialize $I_{best} = 0$, $n = m$

for $k = 0$ to k_{MAX} **do**

Compute T'_n. If $t > T'_n$ then $n = n + 1$

if $t < T'_n$ [1] **then**

Select $m-1$ random samples from $\mathcal{U}_{n-1}$ and $u_{g(t)}$ (see eq. 5.9)

else

Select m random samples from $\mathcal{U}_n$

end if

$H \leftarrow$ Generate a hypothesis using m samples.

$I_k \leftarrow$ Evaluate the current hypothesis support by counting the number of inliers.

if $I_k < I_{best}$ **then**

$I_{best} = I_k$

Update the hypothesis with the the strongest support.

$k_{MAX} \leftarrow$ Update termination criterion as described in 6.1.1

end if

end for

5.2.2 Randomized RANSAC (R-RANSAC)

Matas and Chum in [44] introduced *Randomized RANSAC* (2002) to address a practical limitation of RANSAC algorithm in real-time applications. This limitation concerns the time needed for generating and evaluating a substantial number of hypotheses.

[1] Note that the inequality provided in the original paper is mistakenly reversed.

The R-RANSAC specifically targets the evaluation step in order to adapt the process for real-time applications. Imagine a highly contaminated set (with a small inlier rate of ε) as the input of RANSAC process in which $\frac{1}{\varepsilon^m}$ samples on average have to be drawn. Therefore, averagely $\frac{1}{\varepsilon^m}$ hypotheses are evaluated against N correspondences regardless of being contaminated by outliers or not. The R-RANSAC benefits from a randomized evaluation algorithm that attempts to reduce the number of verifications by evaluating contaminated hypotheses only against a small subset of points. This randomized evaluation is a two-fold process. The first step contains a statistical pre-test on a small number of randomly selected points. The next step includes verifications against all N points and is only performed if the pre-test is successfully passed. By performing such a statistical test on a small fraction of points, it can be rapidly inferred whether the model is worth to be evaluated against the remaining points or not.

The pre-test that plays a key role in the performance of the process is first started with the $T_{d,d}$ test [14] then leads to an optimal sequential test called SPRT in [43]. We will elaborate on these pre-verification tests in section 6.1.2 of Chapter 6.

5.2.3 Preemptive RANSAC

David Nisteŕ introduced *Preemptive RANSAC* [48] with a modified verification scheme also taking real-time constraints into account. Unlike *depth-first* approaches in which each hypothesis is verified against all correspondences before generating new hypothesis, in the proposed *breadth-first* verification, a fixed number of hypotheses are firstly generated. These are then sorted in a descending order based on the score achieved over a fraction of observations. In this parallel evaluation method, a chunk of random observations is selected and used to evaluate the generated hypotheses $h = 1, \ldots, f(i)$ $i := 1, \ldots, N$. As we iterate over i, the $f(i+1)$ highest score hypotheses are retained until only one superior hypothesis remains or all observations have been used. The preemption function that specifies the number of retaining hypotheses at each step is described as

$$f(i) = \lfloor M 2^{-\lfloor \frac{i}{B} \rfloor} \rfloor \tag{5.10}$$

Where M is the number of hypotheses have to be generated beforehand and B is the size of observation block. From this strictly decreasing preemptive function, it is clear that the size of retaining set is reduced to half every time i reaches a multiple of B.

5.2.4 Adaptive Real-time RANSAC (ARRSAC)

More recently, the ARRSAC [57] framework is designed to take advantages of both *depth-first* and *breadth-first* to impose real-time application in an adaptive manner. This adaptivity is provided by choosing the number of initial hypotheses M based on an online estimate of inlier rate ε. In the Preemptive RANSAC [48], overestimating the contamination level results in a too large M and underestimating it may cause the preemptive procedure to be unable to find a correct solution. The ARRSAC improves this behavior by computing the inlier ratio ε every multiple of B iterations and update the M accordingly. An upper bound for M is also considered to guarantee a fixed time budget.

The ARRSAC profits from Wald's SPRT [43] evaluation method in order to estimate inlier ratio only using a fraction of observations. As a consequence, this estimate $\hat{\varepsilon}$ may differ from the true value thus requiring additional hypotheses to be generated in the next stage. It is claimed in [57] that the parallel evaluation of hypotheses against each other is beneficial in the sense that it does not require too much time for local optimization step. Note that the ARRSAC employs an inner RANSAC loop for generating the initial hypothesis set.

5.2.5 MLESAC

The RANSAC sampling strategy can perform more effectively when combined with the likelihood of the synthesized model as a measure of quality. Torr and Zisserman [75] introduced MLESAC and MSAC algorithms that capitalize on the Maximum Likelihood Estimation approach and a simple M-estimator respectively.

As discussed earlier, setting a proper threshold for the range of inliers' error can drastically degrade the overall performance of a RANSAC process. As an alternative to the number of inliers, RANSAC can be conducted in a way to minimize the cost of reprojective residuals considering a specific cost function $\rho(.)$. The error for each point is computed by the distance between the point $\mathbf{x_{1,2}}$ and its projection $\hat{\mathbf{x}}_{\mathbf{1,2}}$ using the estimated model.

$$e_i{}^2 = \sum_{j=1,2} (\hat{x}_j - x_j)^2 + (\hat{y}_j - y_j)^2 \tag{5.11}$$

The $\sum_i \rho(e_i{}^2)$ which has to be minimized is ruled by a M-estimator that gives outliers a constant penalty c while inliers are weighted according to their squared error.

$$\rho(e^2) = \begin{cases} e^2 & \text{if } e^2 < c^2 \\ c^2 & \text{otherwise} \end{cases} \tag{5.12}$$

To gain 95% performance from the M-estimator, c is set to 1.96σ.

Using Equation 5.1, MLESAC scores all correspondences by their maximum likelihood error. Under the assumption of independent and identically distributed samples with zero-mean Gaussian distribution, the likelihood is defined as

$$\mathcal{L}(\theta|\mathbf{e}) = \prod_{i=1}^{N} p(e_i|\theta) = \prod_{i=1}^{N} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{n} e^{-\sum_{j=1,2}(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 / 2\sigma^2} \tag{5.13}$$

To account for distribution of outliers' error, the above likelihood error can be more accurately expressed as a mixture of a Gaussian and uniform distribution. This mixture model thus relies on a priori estimate of the mixture parameter γ.

$$p(e) = \gamma \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{e^2}{2\sigma^2}\right) + (1 - \gamma\frac{1}{w}) \tag{5.14}$$

Where w is a constant value specifying the pixel range of uniform distribution $(\frac{-1}{w} \ldots \frac{1}{w})$. The negative log likelihood is denoted by

$$-L = -\sum_{i} \log\left\{\gamma\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{N} exp\left(\frac{-\sum_{j=1,2}\left((\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2\right)}{2\sigma^2}\right) + (1-\gamma)\frac{1}{w}\right\} \tag{5.15}$$

Torr and Zisserman suggest to use an expectation maximization method (EM) to estimate the mixture parameter γ that maximizes $-L$ for each hypothesis. Despite the fact that more computational budget is needed for MLESAC due to the estimation of the mixture parameter, it slightly outperforms MSAC which requires no extra load.

5.2.6 Guided MLESAC

The *Guided MLESAC* is introduced in [74] as a modification to the MLESAC in which the number of iterations is reduced by an order of magnitude. The Guided MLESAC made this modification by challenging two major flaws in the mixture model defined by Torr and Zisserman. The first flaw is due to the assumption stating that the prior probability γ is equal for all samples regardless of their quality. Additionally, it can be inferred that the prior probability of a match to be valid does not depend on the hypothesized model.

As opposed to MLESAC in which the probability that indicates whether a sample is valid or invalid is uniformly distributed across all matches, the guided MLESAC reasonably assumes these probabilities to be distributed according to the matches' score. Let r_i be a random variable stating whether the hypothesized model is consistent with i_{th} data point, the conditional probability of a mismatch given the corresponding score s_i is denoted by $p(\bar{r}_i|s_i)$ and modeled with a quadratic relation. On the other hand, the conditional probability of a valid match $p(r_i|s_i)$ is modeled with a normalized Gaussian distribution. Note that the relations between probabilities and match correlations are modeled by considering their frequency of occurrence in different test cases for the both valid and invalid matches.

It is also shown in [74] that iterative estimation of the mixing parameter γ is not necessary for each hypothesis. Instead, the prior probabilities of each sample can straightforwardly be computed by the corresponding score and the total number of samples.

5.2.7 Locally Optimized RANSAC (LO-RANSAC)

Since the model that is returned by RANSAC is hypothesized using a noisy minimal set, the noise propagates to the model that makes it inconsistent with all inliers. The goal of *Locally Optimized RANSAC* [17] is to refine a sought model given a set of putative inliers. Hence, a locally optimized model is found by utilizing non-minimal set through an inner-RANSAC loop with a fixed number of iterations. The inner loop uses the support of the best hypothesis found so far as a starting point to sample non-minimal sets.

The refined model generated using non-minimal sets that are chosen from uncontaminated inliers consequently is on course to be consistent with more correspondences implying more accurate model. Although the LO-RANSAC puts extra computational cost ahead of parametric model estimation, the refined model causes the procedure to be terminated earlier.

5.3 Conclusion

As one of the core requirements for most target detection algorithms, we studied the parameter estimation problem in the context of robust pose estimation. The uncertainties associated with feature based matching algorithms urged us to devise a robust solution capable of handling significantly contaminated correspondences. So we comprehensively investigated several estimation algorithms and discussed their limitations as well as their

advantages over the other approaches. Firstly we started from robust regression algorithms such as Maximum Likelihood Estimation, Least Median of Square residuals and M-estimator. According to the poor performance of these algorithms in the presence of highly contaminated observations, robust algorithms based on the hypothesize-and-verify scheme were introduced.

The basic RANSAC scheme is extensively challenged on several grounds yielding several variants of the original RANSAC that aim at improving the efficiency of the algorithm. For each variant of the hypothesize-and-verify method, we provided a concise explanation of fundamental steps toward a suitable framework for real-time model estimation problems.

In the following chapter, we will propose an integrated framework that accounts for limitations and advantages of each explained algorithm to come up with a robust and device-friendly implementation for plane-to-plane homography estimation.

Chapter 6

Robust Target Matching Framework

The first step toward a typical framework for a real-time 2D/3D registration is to extract stable features from a video frame. These features are then compared with those of a reference target already extracted during an offline process, resulting in a set of putative correspondences. In our implementation, we propose the FAST-9 feature detector [61] and the BRIEF binary descriptor [12] for faster performance. These are used in a model generation framework from view synthesis as described in Chapter 3. The typical scheme used to implement such a system is illustrated in Figure 6.1.

Once an initial match set is available, the next step consists in real-time and robust estimation of a global transformation that best explains these matches. This is commonly achieved by using a RANSAC scheme based on a model hypothesize-and-verify loop in which each iteration implies a model estimation step. This model is in our case a 2D homographic relation and its parameters are evaluated using the Gaussian Elimination algorithm by selecting a minimal set of correspondences. The proposed homography estimation by Gaussian elimination algorithm has been designed for real-time planar target matching on hand-held devices.

In practice, while establishing the set of detected matches, to some probability we might select wrong matches (outliers). This wrong selection drastically induces inaccurate pose estimation which results in overall performance degradation. To remove the effect of outliers, a robust framework based on the RANSAC scheme is proposed to cope with this uncertainty.

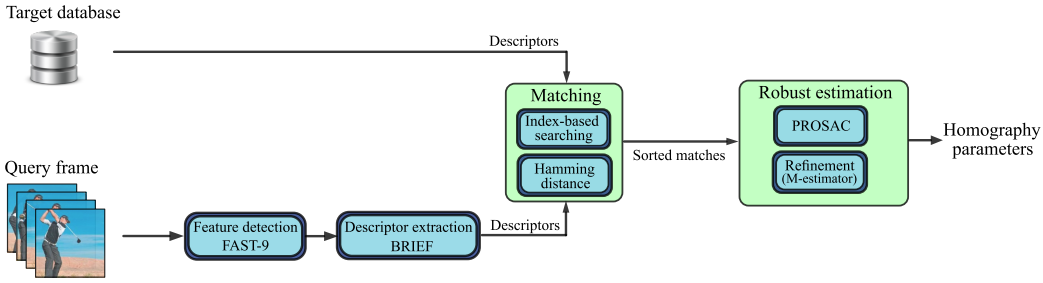


Figure 6.1: Robust homography estimation framework

6.0.1 Homography estimation by Gaussian Elimination

In matching applications, robust estimation of a homography from a set of putative matches is achieved based on the RANSAC algorithm. RANSAC randomly selects four point correspondences from the match set and estimates a homography relation. By repetitively estimating a homography from different random selection, the best homography is identified as the one that is supported by the largest number of point correspondences in the set.

Even if the SVD estimation from four point correspondences can be performed with a relative efficiency, its repetitive computation can still impose a significant computational load in the context of real-time estimation using low-power devices. This observation leads us to consider simpler approaches to resolve the 4-point homography estimation problem. In particular, we selected the well-known Gaussian Elimination scheme that can be used to solve a 4-point non-homogeneous set of equations. Even if this approach is known to be less numerically stable, we show here that in the context of target recognition, stable and accurate solutions can still be obtained. As a starting point, once again we consider the vector-form of homogeneous system of equations that is described in section 4.2.1:

$$\begin{pmatrix} \mathbf{0^T} & -\mathbf{x_i}^\mathbf{T} & Y_i\mathbf{x_i}^\mathbf{T} \\ \mathbf{x_i}^\mathbf{T} & \mathbf{0^T} & -X_i\mathbf{x_i}^\mathbf{T} \\ -Y_i\mathbf{x_i}^\mathbf{T} & X_i\mathbf{x_i}^\mathbf{T} & \mathbf{0^T} \end{pmatrix} \mathbf{h} = \mathbf{0} \tag{6.1}$$

Our implementation of the reduction to reduced-row-echelon form of the matrix is summarized here. It assumes that the minimum configuration is used to estimate the homography, that is 4 matches. If we take the matrix in 6.1 to be decomposed as (after

appropriate row shuffling):

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & X_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -y_3X_3 & X_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & Y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3Y_3 & -y_3Y_3 & Y_3 \end{pmatrix} \tag{6.2}$$

We notice here that the matrix is somewhat sparse, and what's more, the top left 4×3 matrix minor is identical to the bottom middle 4×3 minor. This is of great help, since it means that initially, the same operations will be applied to the top 4 rows and bottom 4 rows of the matrix. Even better, when 4-lane or 8-lane vector processing engines (such as SSE, AVX, Altivec or NEON) are available, the loads of x_i, X_i, y_i and Y_i, the multiplies xX, xY, yX and yY and the row operations can be done in parallel.

In Appendix A, more detailed explanation toward finding the reduced-row-echelon form of 6.1 is presented. After the reduction procedure, the right-most column of the resulting matrix would contain elements of homography matrix. With this non-homogeneous solution, poor estimation would be obtained if the element h_{22} should actually have a value close to zero. Gaussian elimination is however numerically stable for diagonally dominant or positive-definite matrices. For general matrices, Gaussian elimination is usually considered to be stable, when using partial pivoting. [22] In practice, we observed reliable stability when the Z-component of the translation is significant with respect to the $X - Y$ ones; this is the common situation when a handheld device is used for target recognition.

6.1 Robust Parameter Estimation

Once a hypothesis generated through the proposed Gaussian Elimination algorithm, it is verified by evaluating its support using the complete correspondence set. The support for each homography model is determined by counting the number of correspondences whose reprojective error is lying within a specific threshold. As the Hamming distance from BRIEF matching can be used as a measure of the quality of a match, we chose here to use the PROSAC variant [15] in which samples are selected from the ordered set of

correspondences based on their similarity score. As explained earlier, compared to the typical RANSAC method, PROSAC greatly reduces the number of attempts for the best support hypothesis, in the sense that it is more likely to choose outlier-free samples from the subset of highest score correspondences.

Algorithm 3 summarizes our PROSAC implementation for a fast homography estimation. The k_{MAX} parameter defines the computational budget available for performing recognition on a frame. It starts from a maximum acceptable value and can be decreased based on the current estimate of the inliers' rate.

Algorithm 3 Robust framework for H estimation

Require: Set of all detected matches $\mathcal{U}_N$ and η_0.
  Sort the set of correspondences with respect to the similarity score.
  Pre-compute χ^2 approximate of $I_{n^*}^{min}$ satisfying 6.4.
  Initialize $I_{best} = 0$, $m = 4$ and k_{MAX}
  for $k = 0$ to k_{MAX} **do**
    Select m non-degenerate pairs using the PROSAC non-uniform approach (see 6.1.3 for the degeneracy test).
    $H \leftarrow$ Generate a hypothesis by Gaussian Elimination approach 6.0.1.
    $I_k \leftarrow$ Evaluate the current hypothesis support.
    if $I_k < I_{best}$ **then**
      $I_{best} \leftarrow I_k$
      Update H with the hypothesis with the strongest support.
      if $I_k \geq I_{n^*}^{min}$ **then**
        Break out of the for loop.
      end if
      $k_{MAX} \leftarrow$ Apply the maximality constraint to update k_{MAX} (see 5.4).
    end if
  end for

6.1.1 Termination criterion

Since the hypothesize-and-verify scheme is an iterative process, it has to be terminated once a specific termination criterion is met. Although the non-uniform sampling property of PROSAC speeds up the hypothesis convergence, applying standard maximality constraint 5.4 under the assumption of uniform sampling results in a larger number of samples drawn than what is actually required.

The Non-randomness Constraint

Non-randomness constraint is a statistical significance test that guarantees the goodness of a solution in a way that the probability of evaluating the points, which are consistent with a good model, as outliers falls below a specific significance level (typically set to $5 - 10$ percent). The probability distribution of evaluating i outliers out of n points all consistent with the sought model abides by the binomial distribution. The binomial distribution is denoted as follows.

$$P_n(i) = \beta^{i-m}(1-\beta)^{n-i+m}\binom{n-m}{i-m} \tag{6.3}$$

In which β is the of probability of a random point evaluated as inlier given an incorrect model. For each subset of the size n, we compute minimum number of inliers satisfying non-randomness constraint whose accumulative p-value is less than a significance level ψ.

$$I_n^{min} = min\{j : \sum_{i=j}^{n} P_n(i) < \psi\} \tag{6.4}$$

Thus, the PROSAC loop is terminated once the sampling subset size for a candidate solution satisfies the maximality condition and

$$I_n \geq I_{n^*}^{min} \tag{6.5}$$

The Chi-squared Approximation

Although the non-randomness criterion accelerates the PROSAC algorithm by reducing the required number of iterations, it still suffers from a heavy computational burden for recursively computing binomial probabilities and optimizing for the stopping subset size n^*. Considering the randomness of a solution as a null hypothesis H_0, p-value determines the significance of rejecting H_0, given a specific level ψ. To this end, a Chi-squared χ^2 test can be used as a common statistical interpretation of p-value. So the larger observed χ^2 corresponds to the lower p-value and thus stronger evidence against the null hypothesis. Figure 6.2 shows a graphical illustration of normal and χ^2 approximation to the binomial probability density function.

From the central limit theorem, for sufficiently large n, equation 6.3 is approximately a standard normal distribution with $\mu = n\beta$ and $\sigma^2 = n\beta(1-\beta)$ and thus a 1 degree of freedom chi-squared distribution under the null hypothesis. Let χ_ψ^2 be the corresponding

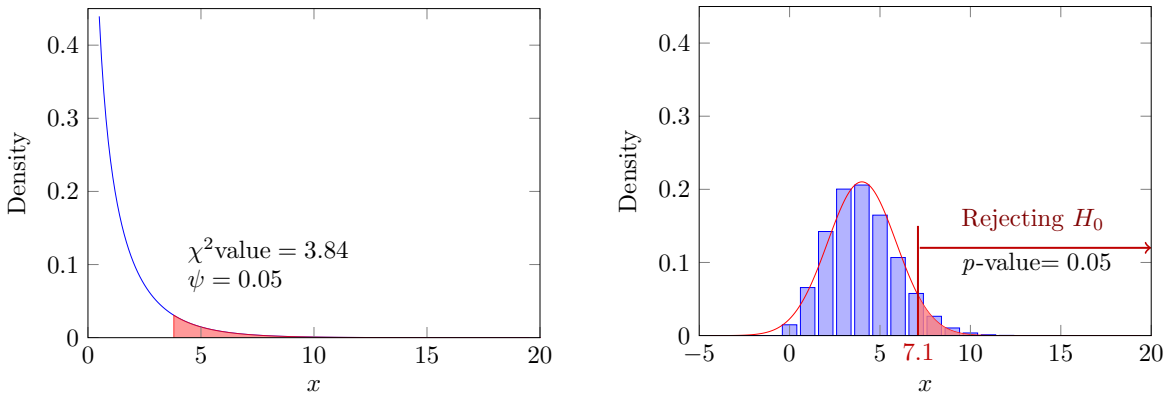


Figure 6.2: Left shows χ^2 distribution with 1 degree of freedom. Right shows Binomial PDF and normal approximation for $n = 30$ and $\beta = 0.1$. The red area under the curves indicate p-value $= 0.05$ under H_0

chi-squared value for the significance level of ψ, which is itself approximately normal PDF, thus equation 6.4 reduces to the form

$$I_{min} = \text{ceil}\big(m + n\beta + \chi_\psi \sqrt{n\beta(1-\beta)}\big) \tag{6.6}$$

It will be shown in Chapter 7, that the Chi-squared approximation saves a big amount of time by computing $I^{min}_{n^*}$ only once prior to the PROSAC loop, yet exhibits nearly similar accuracy to the standard non-randomness approach.

6.1.2 Model verification

For evaluating hypothesis support, note that the standard RANSAC model verification step could have been further optimized by using quick hypothesis filtering strategies such as the $T_{d,d}$ test or the *Sequential Probability Ratio Test* (SPRT) [16].

The $T_{d,d}$ Test

In the RANSAC process, we wish to minimize the number of samples that have to be drawn in addition to the average time $\bar{t}$ required to validate each hypothesis. While the early termination criterion is responsible for optimizing the number of hypotheses, verification tests such as $T_{d,d}$ [14] are designed to minimize $\bar{t}$ (which is proportional to the number of verified points). The $T_{d,d}$ verification algorithm is divided into two simple steps. In the first step, a small portion of N data points are verified from a randomly

selected subset $\mathcal{U}_d \subset \mathcal{U}_N$. The second step involves the verification of all remaining data points, only if the first pre-test passes such that all d selected points are consistent with the hypothesis.

It is proven in [14] that the optimal solution minimizing the average number of verified points $\bar{t}$, leads us to the $T_{1,1}$ ($d = 1$) test. Considering the probabilities of drawing 'good' samples P_g and its complement $1 - P_g$ (drawing 'bad' samples), one can derive $\bar{t}$ as a function of d:

$$\bar{t}(d) = P_g(\alpha N + (1 - \alpha)\bar{t}_\alpha) + (1 - P_g)(\beta N + (1 - \beta)\bar{t}_\beta) \tag{6.7}$$

Equation 6.7 states that when a 'good' sample is provided, it yields the verification of N points with the probability of α (that an uncontaminated point passes the pre-test). Otherwise it requires $\bar{t}_\alpha$ points in average. Similarly when a 'bad' sample is provided, N points have to be verified with the probability of β that a contaminated sample successfully passes the pre-test. Else, averagely $\bar{t}_\beta$ points are verified.

Since the significance of the $T_{d,d}$ is mainly rooted in a quick rejection of contaminated samples, it is prone to reject uncontaminated samples accordingly. Although this behavior results in drawing more samples than the standard RANSAC, the performance gain is noticeable while employing fast hypothesis generation approaches like the *Gaussian Elimination* method.

The SPRT Test

The idea behind the *Sequential Probability Ratio Test* inspired from Wald's theory [82] is to conduct a statistical test on a smaller number of data points to take an earlier decision on accepting or rejecting the generated model. The predominance of SPRT is to optimize for verification time, yet maintaining reliability given probability bounds on pairs of hypotheses namely H_g and H_b that state whether a model is 'good' or 'bad' respectively. The Wald's likelihood ratio [82] to make such a decision is

$$\lambda_j = \prod_{r=1}^{j} \frac{p(x_r|H_b)}{p(x_r|H_g)} \tag{6.8}$$

In this conditional probabilities' ratio, x_r is 1 if r_{th} data point is consistent with the generated model, and 0 otherwise. In practice, $p(1|H_g)$ (the probability of a random point consistent with the model) is *priori* unknown and we approximate it with its lower bound ε which is equal to the best inlier's fraction found so far. In addition,

the probability of a random point consistent with a degenerate model $p(1|H_b) = \delta$ is a Bernoulli distributed probability function that can be estimated using average inliers' fraction of the discarded model.

The time optimization of this statistical decision making algorithm highly depends on an accepting threshold A. It means that the model is known as a degenerate model if for a specific j, λ_j in 6.8 becomes greater than A. The authors of [16] found the optimal solution for A by minimizing the time $t = k(t_M + \overline{m}_S t_v)$. Let k be the average number of points to be verified and $\overline{m}_S$ implies the average number of solutions found given a minimal sample set. t_m indicates the time needed to generate a model while t_v is the time to verify each sample.

6.1.3 The Degeneracy Test

According to the randomness of the RANSAC algorithm, a large fraction of hypotheses may be generated from degenerate configurations of the points. By considering the time needed for evaluation of each degenerate hypothesis against all correspondences, one can improve efficiency by applying a cheap pre-filtering test prior to the model generation stage.

In principle, to greatly speed up the pose estimation algorithm, it is preferable to reject samples quickly rather than speed up the *hypothesize-and-verify* processes. In [71], a pre-filtering test is proposed that prunes out degenerate samples by relying on the rotational and positional consistency of correspondences. This pre-test that requires local orientation of points, ensures the consistency of orientation differences between the selected points in the query frame and the corresponding points in the reference frame. Since in many practical cases, the orientation of participating points are not provided, a *Geometric Constraint* is introduced in [46] to discard samples with degenerate configurations. In the case of homography estimation, the selected samples inducing a unique plane satisfy the geometric constraint only if the points in the query frame do not violate a relative order of their correspondences in the reference frame. Let $S = \{(\mathbf{a_0}, \mathbf{a_1}), (\mathbf{b_0}, \mathbf{b_1}), (\mathbf{c_0}, \mathbf{c_1}), (\mathbf{d_0}, \mathbf{d_1})\}$ be a set of four selected points, it is proven in [46] that the relative ordering of three out of four points is held if and only if

$$\text{sign}\left((\mathbf{a_0} \times \mathbf{b_0})^T \cdot \mathbf{c_0}\right) = \text{sign}\left((\mathbf{a_1} \times \mathbf{b_1})^T \cdot \mathbf{c_1}\right) \tag{6.9}$$

Where $\times$ and $\cdot$ are cross and dot product operators respectively. A *Weak Constraint* is defined when the first three combination of points holds the equation 6.9 while a *Strong Constraint* ensures the relative ordering of all possible subsets from S.

6.2 The Model Refinement

In the previous sections we investigated a robust approach to estimate homography parameters and came up with a unified framework as a combination of different approaches. However even with all improvements we have achieved so far, there is still an issue left as a drawback of such a framework. As the hypothesized model is generated using a minimal set of noisy correspondences, the preferred approach may be unable to return a globally optimal solution due to unaccounted noise in RANSAC-style procedures. Furthermore, giving a minimal set of all-inlier correspondences as an asset for generating a model to be in agreement with all inliers necessitates a large number of RANSAC iterations.

As a solution, a refinement stage can be employed to refine each synthesized model to more precisely explain all available noisy inliers.

6.2.1 Robust M-estimator

The pose refinement can also be seen from the viewpoint of a non-linear minimization problem in which the model with the strongest support is considered as an initial guess for numerically solving this problem. The refined model's parameters are found by minimizing the sum of reprojective distance of residuals such that

$$P(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \sum_{i} d^2(\hat{\mathbf{x}}^i_{\boldsymbol{\theta}} - \mathbf{x}^i) \tag{6.10}$$

Non-linear minimization problems with non-linear cost function $\rho(.)$ are often solved by iterative algorithms such as Gauss-Newton or Levenberg-Marquardt where $\rho(\boldsymbol{\theta}_i + \boldsymbol{\Delta}_\mathbf{i})$ is linearized by its first order approximation as $\rho(\boldsymbol{\theta}_i) + \mathbf{J}\boldsymbol{\Delta}_i$. $\mathbf{J}$ indicates the Jacobian matrix of ρ. In the Gauss-Newton method, at each iteration i, the vector of parameters $\boldsymbol{\theta}_i$ is changed with the gradient of ρ but in the opposite direction.

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_{i-1} + \boldsymbol{\Delta}_{i-1} \tag{6.11}$$

by denoting vector of residuals as $\mathbf{r}$,

$$\boldsymbol{\Delta}_{i-1} = -(\mathbf{J}^\mathbf{T}\mathbf{J})^{-\mathbf{1}}\mathbf{J}^\mathbf{T}\mathbf{r}_{i-1} \tag{6.12}$$

Since the sum of squared residuals is highly sensitive to a single outlier, a more robust algorithm to outliers is called M-estimator that employs different cost functions to reduce the effect of outliers as much as possible. To this end, Huber (1981) [30] and Beaton and

Tukey (1974) [9] introduced two popular robust functions for re-weighting residuals. By defining a weight matrix $\mathbf{W} = \text{diag}(w_1, w_2, ..., w_n)$, the equation 6.12 is changed to

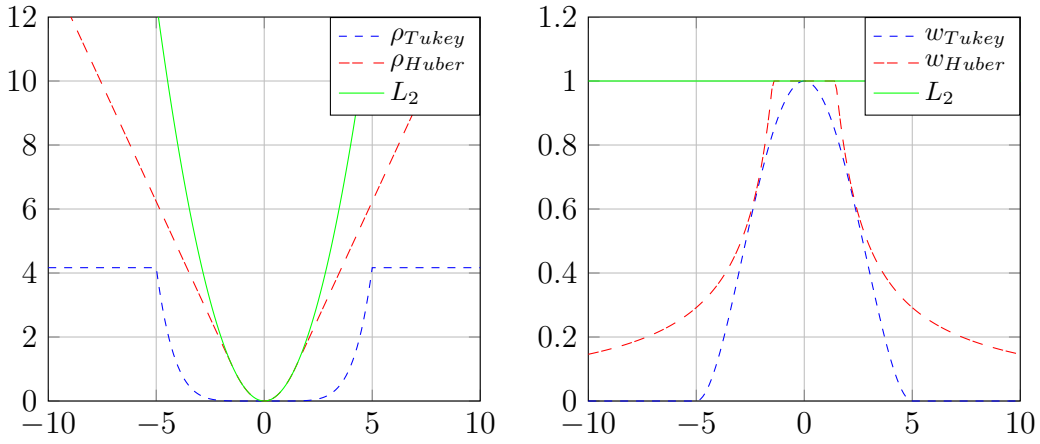
$$\mathbf{\Delta}_{i-1} = -(\mathbf{J^T W J})^{-1}\mathbf{J^T W r}_{i-1} \tag{6.13}$$

The Huber and Tukey's bi-weight functions are both quadratic when the residuals are close to zero. The Huber's function ρ_{Huber} linearly penalizes residuals greater than a constant value c while Tukey's function ρ_{Tukey} more suppressively treats outliers by giving them a constant penalty. Equations 6.14 and 6.15 express cost and weight functions of Huber and Tukey methods respectively.

$$\rho_{Huber}(x) = \begin{cases} x^2/2 & \text{if } |x| \le c \\ c(|x| - k/2) & \text{if } |x| > c \end{cases} \qquad w_{Huber}(x) = \begin{cases} 1 & \text{if } |x| \le c \\ k/|x| & \text{if } |x| > c \end{cases} \tag{6.14}$$

$$\rho_{Tukey}(x) = \begin{cases} c^2/6\Big(1 - \big(1 - (x/c)^2\big)\Big)^3 & \text{if } |x| \le c \\ c^2/6 & \text{if } |x| > c \end{cases} \quad w_{Tukey}(x) = \begin{cases} \big(1 - (x/c)^2\big)^2 & \text{if } |x| \le c \\ 0 & \text{if } |x| > c \end{cases} \tag{6.15}$$

It is worth to note that although Tukey's cost function is not a convex function, it is less likely to get stuck to local minima since the initial estimate given by RANSAC process is close enough to the global minimum. Figure 6.3 illustrates cost and weight functions of Huber and Tukey as well as a quadratic least-square function.



(a) Cost functions (b) Weight functions

Figure 6.3: a)Huber and Tukey's robust cost functions. b) Huber and Tukey's robust weight functions.

6.3 Conclusion

In this chapter, we proposed a fast framework for robust homography estimation that can efficiently run under resource-constrained platforms. This framework profits from the non-uniform sampling approach of PROSAC and approximates the non-randomness stopping criterion using the χ^2 statistic test. The verification stage comprises of the Sequential Probability Ratio Test to accelerate the overall performance. The whole process is significantly improved by employing an effective refinement level based on the M-estimator method. Our proposed framework that can be further extended for even broader range of fitting problems is an ideal choice for low-powered devices with limited resource.

Since the estimation step is repeated many times through the hypothesize-and-verify scheme, we presented an algebraic solution for plane-to-plane homography estimation relying on the well-known Gaussian Elimination algorithm. Moreover, complementary results to this chapter are presented in Chapter 7, which shows that this simplified approach significantly reduces the computational load for a real-time implementation. Additionally, we will show from experimentation that the homographies obtained using our optimized GE implementation have an accuracy comparable to the ones obtained by the more conventional SVD solution.

Chapter 7

Experimental Results

7.1 Experimentation

This section presents experimental results showing the performance of our robust pose estimation framework. The homography estimation framework described in Chapter 6 is entirely written in C/C++ using OpenCV library which is a well known and open source computer vision library. Our cross platform implementation has been also ported to an Android device. For porting the application to Android OS, we used *Java Native Interface* (JNI) and *Native Development Kit* (NDT) that compiles native C++ code under Java virtual machine. Through our experimentation we observed that the frame rate of the recognition process runs at around 40 FPS on a smart-phone with Quad core 1.4 GHz Cortex-A9 processor and at 80 FPS on a PC equipped with a 2.26 GHz processor. Figure 7.1 illustrates a screen-shot of the application on a Samsung Galaxy SIII device.

We assessed the reliability and accuracy of the homography estimation itself as well as the resulting recognition rate and efficiency when used in the context of planar target recognition.

7.1.1 Homography Estimation Performance

In order to validate our homography estimation algorithm, we used the test set proposed in [56] to benchmark the USAC framework. We produced the same performance tables as in [56] in which a homography is estimated using different image pairs. The matches are provided in different level of contamination allowing better evaluation of robustness to the outliers. We conducted comprehensive experiments and analyzed the result for

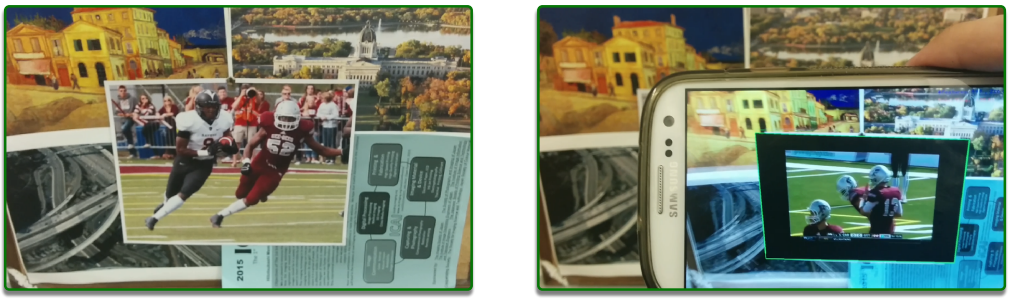


Figure 7.1: AR application based on our optimized homography estimation framework.

each step of the robust estimation process.

By referring to Table 7.1, the first column shows the performance we obtained using the standard USAC 1.0 framework. In the second column, we simply replaced the USAC SVD estimation by our Gaussian Elimination (GE) implementation. Very similar performances are obtained which demonstrate that GE estimation is also able to provide accurate estimates. The computational timings are also similar and this is explained by the fact that under the full USAC framework, the homography estimation stage does not represent a significant portion of the total computation. We therefore ran a new set of experiments in which we removed the more costly local optimization and symmetrical re-projection error steps. The error values indicate that removing symmetrical error cost does not noticeably degrade the overall performance. Moreover, too high accuracy is not required in the context of augmented reality application. In such a case, the benefit of using GE in the estimation of the homography becomes apparent (compared to the third column SVD results, fourth column GE results are 2 to 5 times faster). Finally, the last column shows the performance of our PROSAC implementation based on GE estimation of the homography. It is clear that the proposed approach massively speeds up execution time by decreasing the number of samples drawn and models generated. The values corresponding to the number of verifications per model show the fact that although these values have been increased in the last column, corresponding execution times are overally improved by the lower models generated. It is worth to mention that numbers following ($\pm$) sign are standard variation values. Note that for the timing and

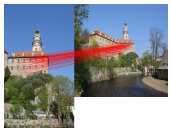
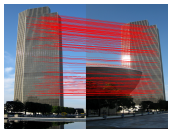
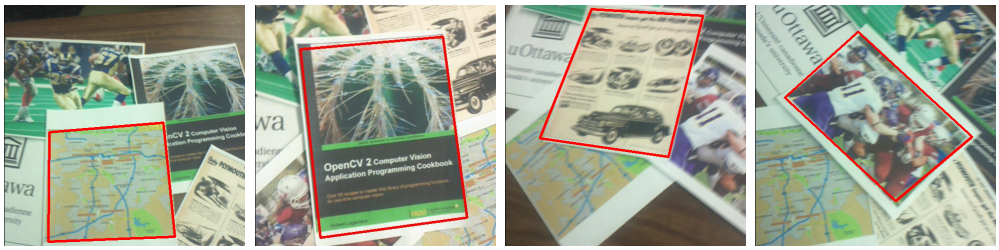
		USAC 1.0	USAC GE	USAC SVD (No LO)	USAC GE (No LO)	**our PROSAC GE**
A: $\epsilon = 0.46$, $N = 2540$ 	I	1147.6 ± 0.1	1147.7 ± 0.1	1074.4 ± 9.1	1017.2 ± 10.1	969.6 ± 10.2
	K	4.8 ± 0	5.9 ± 0.1	7.8 ± 0.1	9.1 ± 0.2	8.4 ± 0.2
	K_rej	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	4.8 ± 0	5.9 ± 0.1	7.8 ± 0.1	9.1 ± 0.2	8.4 ± 0.1
	VPM	755.6 ± 15.6	667 ± 16.4	1021.6 ± 16.6	869.3 ± 16.1	1193.5 ± 14.8
	error	1.27	1.27	1.18	2.22	2.27
	time(ms)	24.78	24.4	0.4494	0.3477	0.0810
B: $\epsilon = 0.15$, $N = 514$ 	I	68.1 ± 0.0	68.0 ± 0.0	67.7 ± 0.5	61.5 ± 0.9	64.3 ± 0.4
	K	925 ± 316	14557 ± 3676	57.0 ± 11.9	165.7 ± 23.0	13.6 ± 0.4
	K_rej	711.2 ± 263.8	12446.8 ± 3226	35.2 ± 10.2	128.0 ± 19.7	3.0 ± 0.1
	models	214.1 ± 53.7	2104.8 ± 451.3	21.8 ± 1.8	36.4 ± 3.3	10.6 ± 0.3
	VPM	49 ± 1.4	42.4 ± 2.3	29.6 ± 2.1	100.4 ± 3.6	294.3 ± 3.6
	error	0.87	0.87	2.08	2.35	2.38
	time(ms)	4.93	3.78	0.2873	0.07323	0.02511
C: $\epsilon = 0.23$, $N = 1317$ 	I	301.0 ± 0.0	300.56 ± 0.3	211.4 ± 1.2	210.9 ± 1.3	202.9 ± 1.4
	K	4.8 ± 0.1	7.5 ± 0.3	4.7 ± 0.1	6.3 ± 0.2	5.0 ± 0.1
	K_rej	0.3 ± 0.0	0.5 ± 0.0	0.3 ± 0.0	0.4 ± 0.1	2.3 ± 0.0
	models	4.5 ± 0.1	4.9 ± 0.3	4.4 ± 0.1	3.9 ± 0.2	2.7 ± 0.1
	VPM	372.6 ± 4.5	593.5 ± 17.5	435.1 ± 6.6	694.9 ± 16.8	1215.7 ± 7.8
	error	0.80	0.8	0.98	1.42	1.35
	time(ms)	6.33	6.3	0.1127	0.07363	0.03406
D: $\epsilon = 0.34$, $N = 495$ 	I	146.2 ± 0.1	146.3 ± 0.1	137.0 ± 1.0	139.6 ± 1.1	136.7 ± 1.2
	K	14.0 ± 0.4	16 ± 0.5	5.1 ± 0.1	5.8 ± 0.1	5.5 ± 0.1
	K_rej	3.7 ± 0.1	4.2 ± 0.2	1.9 ± 0.0	2.0 ± 0.0	2.7 ± 0.0
	models	10.3 ± 0.3	10.8 ± 0.4	3.2 ± 0.1	3.0 ± 0.1	2.8 ± 0.1
	VPM	103.4 ± 2.3	111.4 ± 3.4	307.3 ± 5.4	342.1 ± 5.9	482.4 ± 1.5
	error	1.16	1.16	5.72	5.70	5.87
	time(ms)	2.73	2.68	0.07764	0.04241	0.016903

Table 7.1: Performance result of Homography estimation as in [56]. (I) is the number of inliers found. (K) and (K_rej) are the number of samples drawn and the number of samples rejected by the degeneracy test. (models) is the number of total hypotheses, (VPM) the number of verification per model. The symmetrical reprojection (error) is measured w.r.t. the ground truth. (time) indicates the execution time per frame in ms. Note that all reported results are averaged over a total of 500 runs.

error quantities, corresponding standard variations are omitted due to their trivial value close to zero.



(a) Map (b) Book (c) Adv (d) Football

Figure 7.2: Homography estimation for one frame of each of our test videos.

Accuracy and Recognition Rate

In order to assess the performance of our optimized framework in the context of planar target recognition, we captured four sets of image sequences for four different types of targets, with each sequence comprising of around 250 to 300 frames[1]. The image sequences were captured using an LG Optimus 2X smartphone camera with a resolution of 480×480. The camera was rotated by approximately $45°$ in all directions (i.e. $45°$ in- and out-of-plane rotation). The scale of the target varies from full resolution (where the target fully occupies the frame) to about one third the image size. A majority of the images suffer from perspective distortions and severe motion blur in some cases. The process of generating the ground truth involves identifying four corners of the target in each frame of sequences. These locations were obtained by applying an expensive SIFT matching step followed by manual corrections made by humans.

The matching scheme based on BRIEF described in Chapter 2 was used to match the target features with the ones detected in each frame of the test sequences. Each matching set thus obtained is then fed to our PROSAC estimator in order to obtain a putative homography. The same experiment was repeated for the different homography estimation methods, all of them using the same initial match sets.

Table 7.2 shows the number of matches in the initial set and the number of matches in the final set with best support as found by PROSAC. We report these results for the SVD solution (as implemented in OpenCV) and for our Gaussian Elimination implementation.

The recognition rate is determined by analyzing the maximum error between the estimated target corner locations to the corresponding ground truth corner location.

[1]Available online at www.eecs.uottawa.ca/∼laganier/projects/mobilevision

Target	Total matches	Total Inliers		Iterations		Recognition rate(%)	
		GE	SVD	GE	SVD	GE	SVD
Book	169.0 ± 33.1	67.1 ± 41.3	61.0 ± 34.7	756.0 ± 710.6	772.0 ± 722.5	48.82	45.40
Map	75.3 ± 18.7	38.1 ± 17.2	38.6 ± 17.3	317.4 ± 546.8	299.4 ± 526.3	72.24	74.75
Football	232.7 ± 41.9	82.2 ± 44.8	79.8 ± 40.8	747.9 ± 723.7	742.2 ± 717.4	84.58	79.06
Adv	200.8 ± 52.8	74.8 ± 41.0	83.0 ± 39.6	693.1 ± 726.8	604.1 ± 661.0	88.09	90.49
Average	175.6 ± 65.6	67.5 ± 41.6	67.3 ± 38.9	658.8 ± 709.7	635.4 ± 694.0	**73.44**	**72.56**

Table 7.2: Average number of total matches, inliers, required iterations and recognition rate are shown for the four targets with both GE and SVD method

This error, given in pixels, is obtained as follows:

$$\mathcal{E}_i(\tilde{C}) = \max_j \|H_i \hat{p}_j - \tilde{p}_{ij}\|, 1 \leq j \leq 4 \ , \tag{7.1}$$

where H_i is the estimated homography at frame i, $\hat{p}_j$ is the coordinate of target corner j in the reference frame and $\tilde{p}_{ij}$ is the manually obtained location of corner j in frame i.

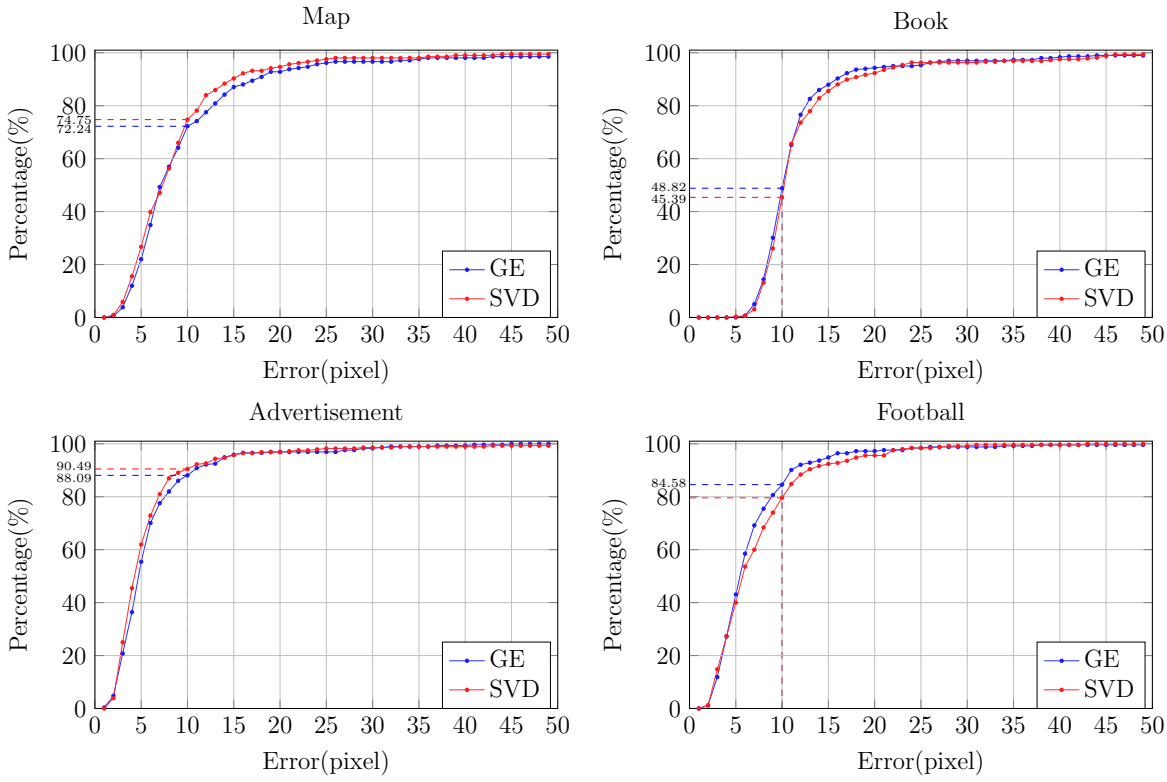


Figure 7.3: Recognition rate for each test sequence, reported as the percentages of frames with maximal positional error less than 10 pixels.

If we consider that a target is successfully detected if $\mathcal{E}(\tilde{C}) \leq 10$ pixels, we then obtain a recognition rate of 72.56% for SVD and 73.44% for Gaussian Elimination averaged over four test sequences (last column of Table 7.2). Here we empirically chose 10 pixels to provide a representative measure of recognition rate to compare the two algorithms (the exact value is not crucial). Individual results corresponding to each row of Table 7.2 are plotted in Figure 7.3.

To illustrate the behavior of the two tested homography estimation methods, we show in Figure 7.4 the evolution of the maximal positional error (reported every 5 frames) for one of the test sequences. As it can be seen, except for one large error made by Gaussian Elimination, both estimation scheme exhibits very similar behavior.

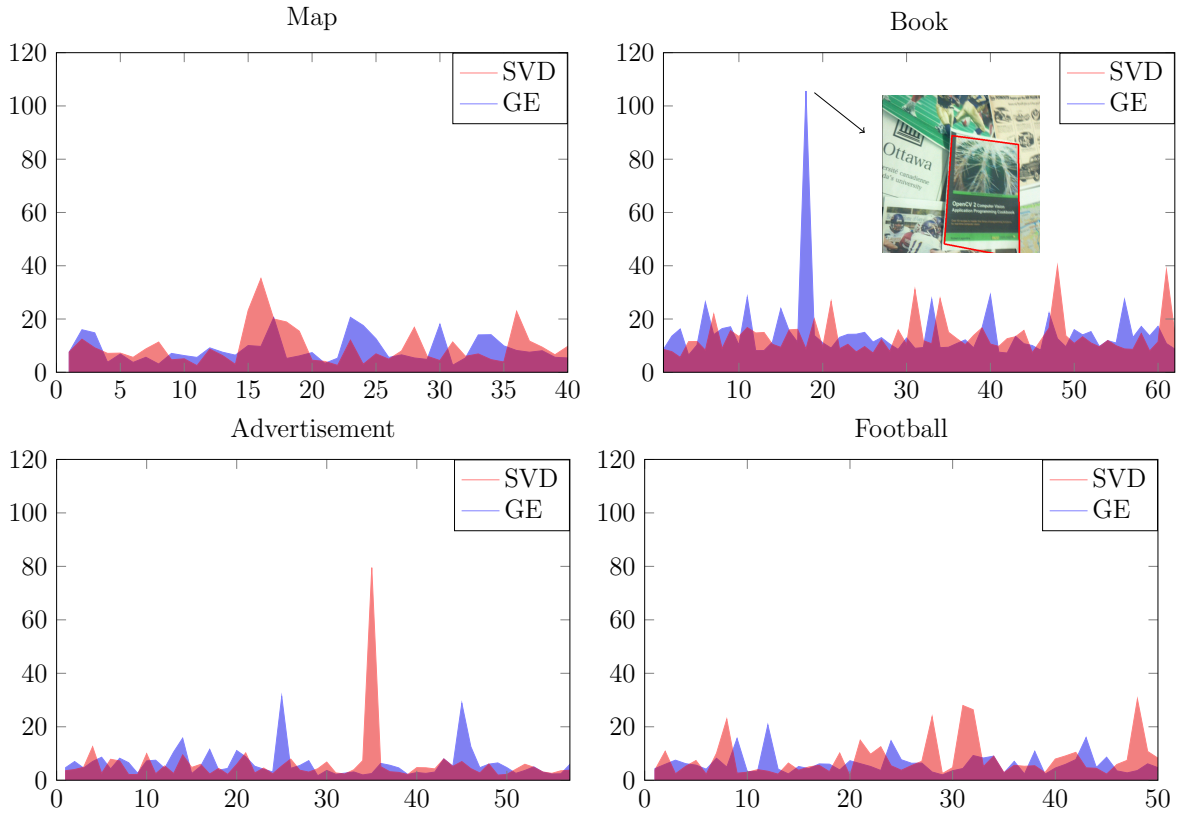


Figure 7.4: Maximum positional error of the four sequences reported every 5 frames.

7.1.2 Target Recognition Performance

Computational Efficiency

We report in this section the global computational efficiency of different homography estimation methods in the context of robust target recognition. Speed is here measured in count of instruction fetches. For completeness, we evaluate the performance of different methods under different contexts; the results are shown in Figure 7.5. First, we measured the speed of the OpenCV `cv::findHomography` function (version 2.4) under the RANSAC mode. We also built our own implementation of the RANSAC scheme inside which we used the OpenCV 2.4 SVD function. We then integrated the same OpenCV 2.4 function under our PROSAC implementation. We also tested the DEGSVD function from the LAPACK package. We also tested a publicly available but non-optimized

Gaussian Elimination implementation [2]. Finally, the last results shown in Figure 7.5 is the one obtained from our proposed optimized Gaussian Elimination scheme. For a device equipped with a $2.26GHz$ CPU, a $30fps$ detection rate corresponds to a maximum number of about 75 millions of cycles.

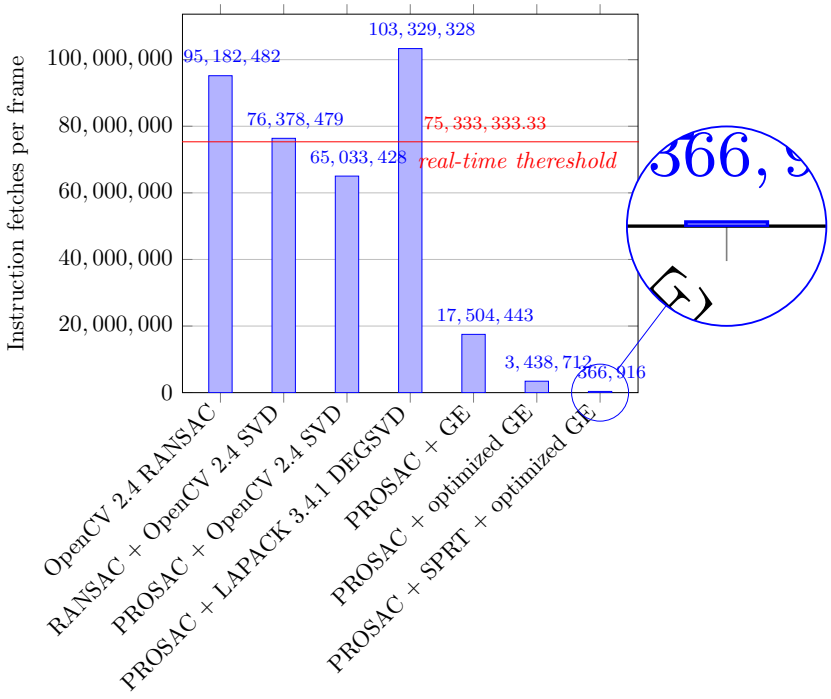


Figure 7.5: Per-frame average instruction fetch count for each H-estimator

We have also compared the speed of our homography estimation with available results as reported in [46]. In Figure 7.6, several estimates of homography, shown by red boxes, are depicted for a sequence of data as used in [46]. On the basis of our results, the runtime for homography estimation is on the order of 5 milliseconds for the approach used in[46], while our optimized framework performs nearly 25x faster on average. Figure 7.7 visualizes the corresponding per-frame numbers for putative matches, inliers's fraction and processing time.

[2]https://github.com/camilosw/ofxVideoMapping

Figure 7.6: Result of homography estimates (shown in red) for the sequence of images as used in [46].

Performance Results of the Model Verification

To compare the performance of the verification stage, we repeated the previous test for different verification methods (see Table 7.3). The GE approach that revealed the best performance in the previous test is also used in the new test for the homography estimation. The results of the first and the third columns show that the number of verifications per model (VPM) is considerably decreased from the standard approach to the verification approach based on $T_{1,1}$. But because the number of generated models in is slightly larger than the standard verification approach, the USAC framework with $T_{1,1}$ does not achieve a significant speed gain. The same reasoning applies to PROSAC with the standard and $T_{1,1}$ verifications. The only difference is that our highly optimized PROSAC framework performs 2-4 times faster than the USAC framework. The last two columns of the table indicate that the verification based on SPRT speeds up the process by a factor of 5 to 10 compared with the standard verification approach. Additionally, SPRT returns more inliers in a smaller number of hypotheses compared with the $T_{1,1}$ test.

Influence of Termination Criteria

As we pointed out earlier, the stopping condition for the iterative hypothesize-and-verify scheme has a significant effect on the speed of the process. The fitness of the sought model also depends on the stopping condition. According to the semi-random sampling strategy of the PROSAC scheme, we altered the maximality termination criterion by imposing

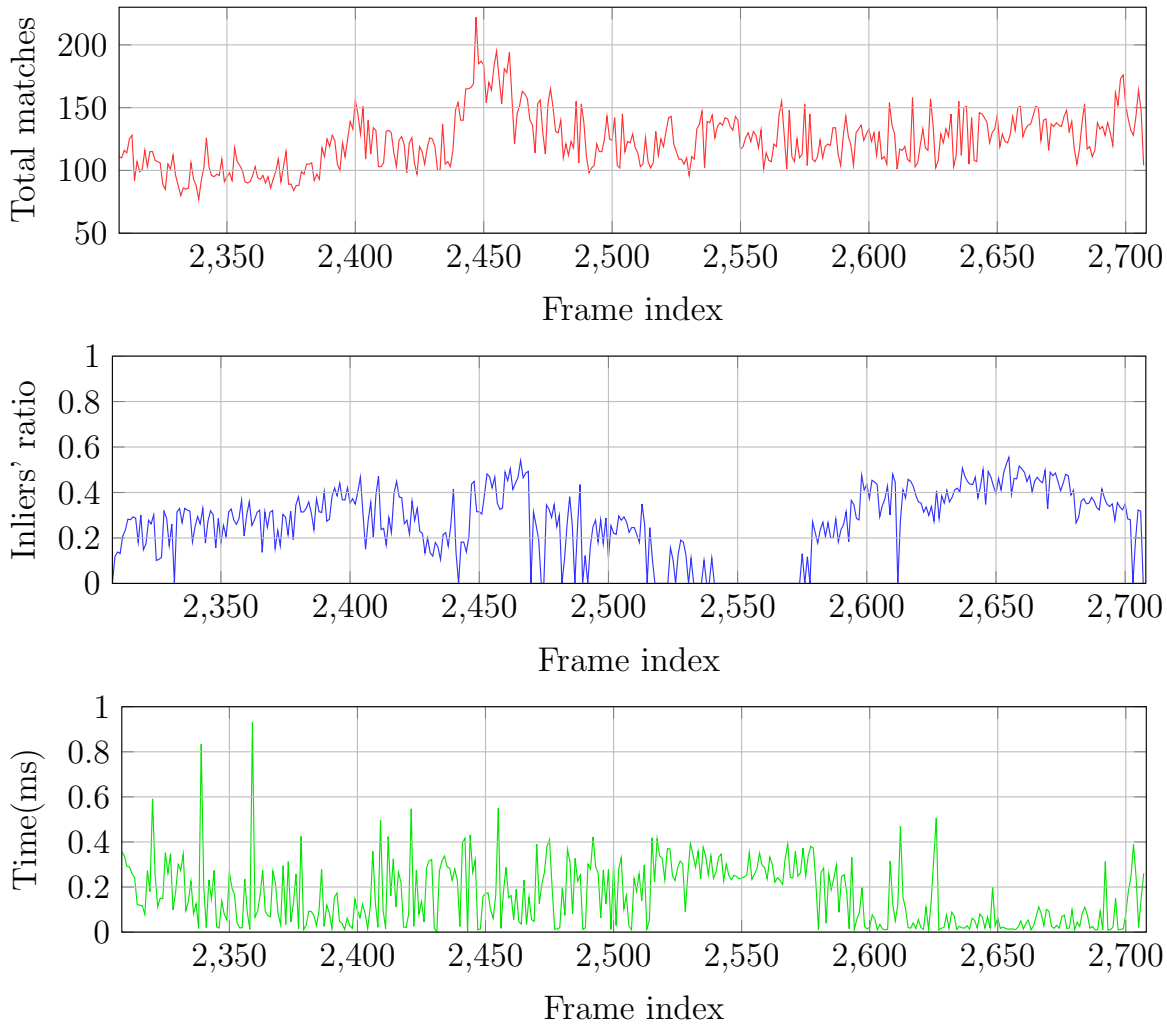


Figure 7.7: Shows number of extracted matches, inlier's proportion(Total inliers vs. total matches) and homography estimation time(ms) for each frame related to Figure (7.6) experiment.

a non-randomness constraint that yields a faster convergence. To carry out a thorough evaluation, we compared our proposed methods with an intuitive termination condition proposed in [71]. This condition determines if the ratio between good inliers (within $2px$) to close inliers (within $15px$) is above a predefined threshold (set to 0.65-0.75). The corresponding results are tabulated in Table 7.4. It can be seen that the number of samples drawn in column 1 with the maximality constraint is significantly larger than the ratio and non-randomness constraints (column 2-5). As a consequence, execution times

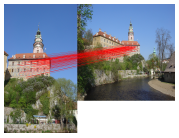
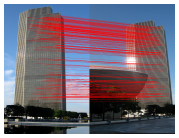
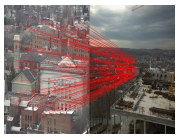
		USAC STD_Verif	PROSAC STD_Verif	USAC $T_{1,1}$	PROSAC $T_{1,1}$	USAC SPRT	**PROSAC SPRT**
A: $\epsilon = 0.46$, $N = 2540$ 	I	1015.7 ± 10.0	957.5 ± 10.4	986.7 ± 11.2	973.9 ± 12.1	1017.2 ± 10.1	969.6 ± 10.2
	K	8.8 ± 0.2	8.5 ± 0.2	14.2 ± 0.4	15.3 ± 0.4	9.1 ± 0.2	8.4 ± 0.2
	K_rej	0.0 ± 0.0	0.0 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	8.8 ± 0.2	8.5 ± 0.2	14.1 ± 0.4	15.2 ± 0.4	9.1 ± 0.2	8.4 ± 0.2
	VPM	2540 ± 0	2540 ± 0	488.3 ± 15.5	371.9 ± 9.0	869.3 ± 16.1	1193.5 ± 14.8
	error	2.23	2.63	2.31	1.13 ± 0.0	2.22	2.27
	time(ms)	0.70	0.117412	0.232	0.070258	0.3477	0.08103
B: $\epsilon = 0.15$, $N = 514$ 	I	58.1 ± 0.5	72.8 ± 0.1	47.1 ± 1.0	39.0 ± 1.5	61.5 ± 0.9	64.3 ± 0.4
	K	11.1 ± 0.1	69.74 ± 13.1	348.0 ± 31.4	411.2 ± 42.4	165.7 ± 23.0	13.6 ± 0.4
	K_rej	1.5 ± 0.0	44.824 ± 11.2	276.0 ± 26.9	361.1 ± 36.5	128.0 ± 19.7	3.0 ± 0.1
	models	8.3 ± 0.2	24.916 ± 1.9	70.5 ± 4.5	97.1 ± 5.9	36.4 ± 3.3	10.6 ± 0.3
	VPM	514 ± 0	514 ± 0	44.2 ± 2.3	51.1 ± 1.4	100.4 ± 3.6	294.3 ± 3.6
	error	2.45	2.50	2.80	3.87	2.35	2.4
	time(ms)	0.12	0.05374	0.073427	0.080152	0.07323	0.025110
C: $\epsilon = 0.23$, $N = 1317$ 	I	205.9 ± 1.3	204.5 ± 1.2	192.1 ± 2.0	206.9 ± 1.5	210.9 ± 1.3	202.9 ± 1.4
	K	4.9 ± 0.1	4.6 ± 0.1	20.4 ± 1.3	30.7 ± 4.5	6.3 ± 0.2	5.0 ± 0.1
	K_{rej}	0.3 ± 0.0	2.3 ± 0.1	2.2 ± 0.5	9.6 ± 2.4	0.4 ± 0.1	2.3 ± 0.1
	models	2.7 ± 0.1	2.4 ± 0.1	15.4 ± 0.8	21.1 ± 2.1	3.9 ± 0.2	2.7 ± 0.1
	VPM	1317	1317 ± 0	295.9 ± 14.6	300.8 ± 14.4	694.9 ± 16.8	1215.7 ± 7.8
	error	1.41	1.30	1.53	1.57	1.42	1.35
	time(ms)	0.12	0.03927	0.070015	0.038238	0.07363	0.034061
D: $\epsilon = 0.34$, $N = 495$ 	I	138.0 ± 1.0	138.1 ± 1.1	142.5 ± 1.4	136.7 ± 1.2	139.6 ± 1.1	136.7 ± 1.2
	K	5.8 ± 0.1	5.7 ± 0.1	10.1 ± 0.3	5.5 ± 0.1	5.8 ± 0.1	5.5 ± 0.1
	K_rej	2.0 ± 0.1	2.7 ± 0.0	2.7 ± 0.1	2.7 ± 0.0	2.0 ± 0.0	2.7 ± 0.0
	models	3.0 ± 0.1	2.9 ± 0.1	6.5 ± 0.2	2.8 ± 0.1	3.0 ± 0.1	2.8 ± 0.1
	VPM	495	495 ± 0	159.9 ± 5.7	482.4 ± 1.5	342.1 ± 5.9	482.4 ± 1.5
	error	5.72	5.74	5.40	5.87	5.70	5.87
	time(ms)	0.05	0.01957	0.030765	0.016903	0.04241	0.016903

Table 7.3: Performance result of Homography estimation for different verification methods

for the ratio and non-randomness constraints have been speeded up ranging between 3x-30x. Comparing the results of column 2 and 3 with column 1, indicates that termination criteria based on the ratio test require much fewer samples have to be drawn. However, the number of generated models are still noticeably larger than the non-randomness approaches. This is due to the fact that, there is no statistical analysis behind the ratio test regarding the goodness of models. Indeed, the aim of adding ratio tests to Table 7.4 is to provide another early termination criterion as a reference to better evaluate non-randomness constraint. It can also be concluded that the criterion based on χ^2

approximation performs quantitatively similar to the non-randomness one in the sense of produced errors.

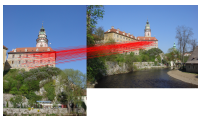
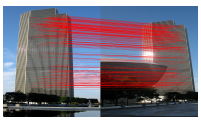
		PROSAC Maximality	PROSAC Ratio= 0.65	PROSAC Ratio= 0.75	PROSAC Non-randomness	**PROSAC χ^2**
A: $\epsilon = 0.46$, $N = 2540$ 	I	1326.4 ± 2.4	1241.7 ± 3.5	1314.6 ± 2.3	969.6 ± 10.2	967.2 ± 10.4
	K	62.7 ± 0.6	17.7 ± 0.7	36.8 ± 1.3	8.4 ± 0.2	8.7 ± 0.2
	K_rej	0.9 ± 0.0	0.1 ± 0.0	0.5 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
	models	61.8 ± 0.5	17.6 ± 0.7	37.2 ± 1.3	8.4 ± 0.2	8.7 ± 0.2
	VPM	741.7 ± 5.6	1138.5 ± 14.3	970.2 ± 13.9	1193.5 ± 14.8	1192.7 ± 14.5
	error	2.06	2.11	2.00	2.27	2.66
	time(ms)	0.16379	0.09050	0.13150	0.08103	0.05638
B: $\epsilon = 0.15$, $N = 514$ 	I	74.1 ± 0.2	65.4 ± 0.3	70.4 ± 0.2	64.3 ± 0.4	72.5 ± 0.2
	K	2000 ± 0	21.8 ± 5.6	42.7 ± 9.8	13.6 ± 0.4	107.2 ± 17.8
	K_rej	1698.93 ± 0.7	10.3 ± 4.8	25.5 ± 8.3	3.0 ± 0.1	77.0 ± 15.2
	models	301.1 ± 0.7	12.4 ± 0.9	18.3 ± 1.5	10.6 ± 0.3	30.3 ± 2.6
	VPM	49.7 ± 0.2	289.4	281.1 ± 3.5	294.3 ± 3.6	267.0 ± 3.6
	error	1.94	2.38	2.29	2.4	3.11
	time(ms)	0.2250	0.02552	0.03190	0.025110	0.036557
C: $\epsilon = 0.23$, $N = 1317$ 	I	296.1 ± 0.4	272.9 ± 0.6	295.4 ± 0.4	202.9 ± 1.4	202.6 ± 1.1
	K	1773.7 ± 5.6	54.7 ± 6.3	708.5 ± 34.8	5.0 ± 0.1	4.6 ± 0.1
	K_rej	1049.7 ± 3.8	19.1 ± 3.6	393.7 ± 21.4	2.3 ± 0.1	2.1 ± 0.1
	models	723.9 ± 2.1	36.6 ± 2.8	315.6 ± 13.4	2.7 ± 0.1	2.4 ± 0.1
	VPM	119.6 ± 0.5	890.7 ± 12.6	381.4 ± 10.8	1215.7 ± 7.8	1244.9 ± 6.5
	error	1.40	1.51	1.40	1.35	1.28
	time(ms)	0.4878	0.093214	0.32967	0.034061	0.018477
D: $\epsilon = 0.34$, $N = 495$ 	I	179.7 ± 0.0	149.5 ± 0.8	160.1 ± 0.6	138.4 ± 1.1	138.4 ± 1.1
	K	263.8 ± 0.1	5.3 ± 0.1	6.8 ± 0.2	5.5 ± 0.1	5.5 ± 0.1
	K_rej	140.2 ± 0.3	2.8 ± 0.1	3.1 ± 0.1	2.7 ± 0.0	2.7 ± 0.0
	models	123.6 ± 0.3	3.5 ± 0.1	4.6 ± 0.1	2.7 ± 0.1	2.7 ± 0.1
	VPM	259.2 ± 0.8	477.2 ± 1.7	460.2 ± 2.4	484.3 ± 1.4	483.9 ± 1.4
	error	3.58	5.50	5.10	5.89	5.9
	time(ms)	0.1343	0.015431	0.017222	0.76780	0.011761

Table 7.4: Performance result of Homography estimation for different stopping criteria

	Maximality	Non-randomness	Chi-squared	Ratio test
Inliers	43.56	38.6	39.3	31.2
Models	160.8	70.4	72.2	111.5
Recognition rate(%)	46.2	34.91	34.81	21.03
Error(pixel)	10.50 ± 16.96	11.92 ± 10.74	10.81 ± 8.38	14.61 ± 46.19
Time(ms)	0.1064	0.15731	0.06488	0.08144

Table 7.5: Performance results of the four sequences with different stopping criteria.

Non-randomness vs. Chi-squared Approximation

We have demonstrated in Table 7.4 the effect of different stopping conditions for still images. However, in the case of target recognition in a live video, the difference between computational timing of the non-randomness and Chi-squared approaches becomes significant. Therefore, we repeated the homography estimation test for one of the sequences of Figure 7.2. Table 7.5 shows, for each algorithm explained in section 7.1.2, the number of inliers, the number of generated models, maximum positional error (in pixels) and the run-time (all reported on average). Except for the ratio test, a very similar accuracy is obtained for the rest of algorithms. Despite the significant reduction in the number of hypothesized models for the non-randomness algorithm, it exhibits a higher execution time. This additional computational cost can be explained according to the recursive process, needed for computing binomial probabilities and optimal value of n^* (the minimum subset size satisfying 6.5). Therefore, the Chi-squared approximation saves a big amount of time by computing n^* only once in prior to the PROSAC loop.

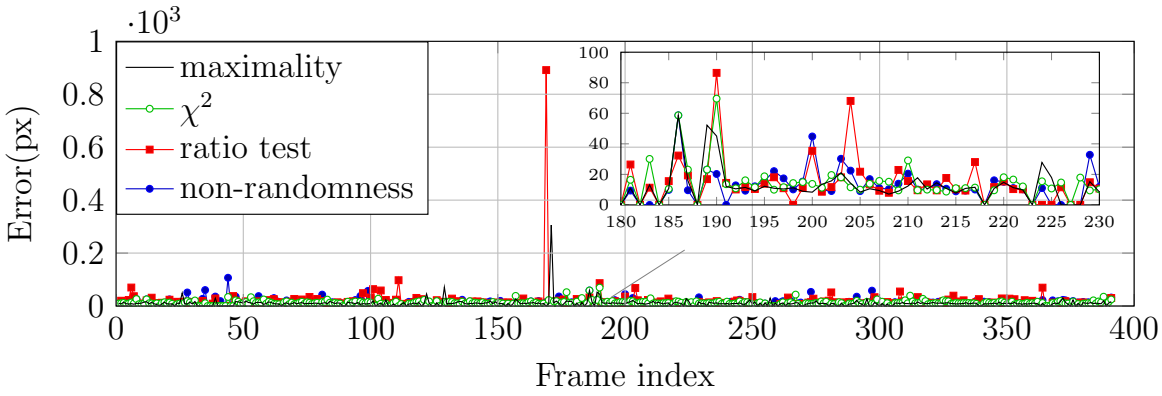


Figure 7.8: Maximal positional error(px) for the homography estimation with different stopping criteria

Figure 7.8 illustrates the the maximal positional error associated with each frame as a measure of accuracy for each algorithm. Although, a very similar range of errors is obtained, the non-randomness and χ^2 approaches exhibit better performances by reducing the peaks induced by the maximality and ratio test approaches. This closely corresponds to the standard deviation values reported in Table 7.5.

For different stopping criteria, we also evaluated a per-frame execution time that is illustrated for each frame in Figure 7.9. The first plot shows the computational timings for all aforementioned algorithms. It is evident from the second and the third plots that the χ^2 algorithm performs faster than the maximality and non-randomness algorithms. It can also be seen from the results that except for the non-randomness algorithm, other stopping conditions do not bring noticeable overhead to the PROSAC loop.

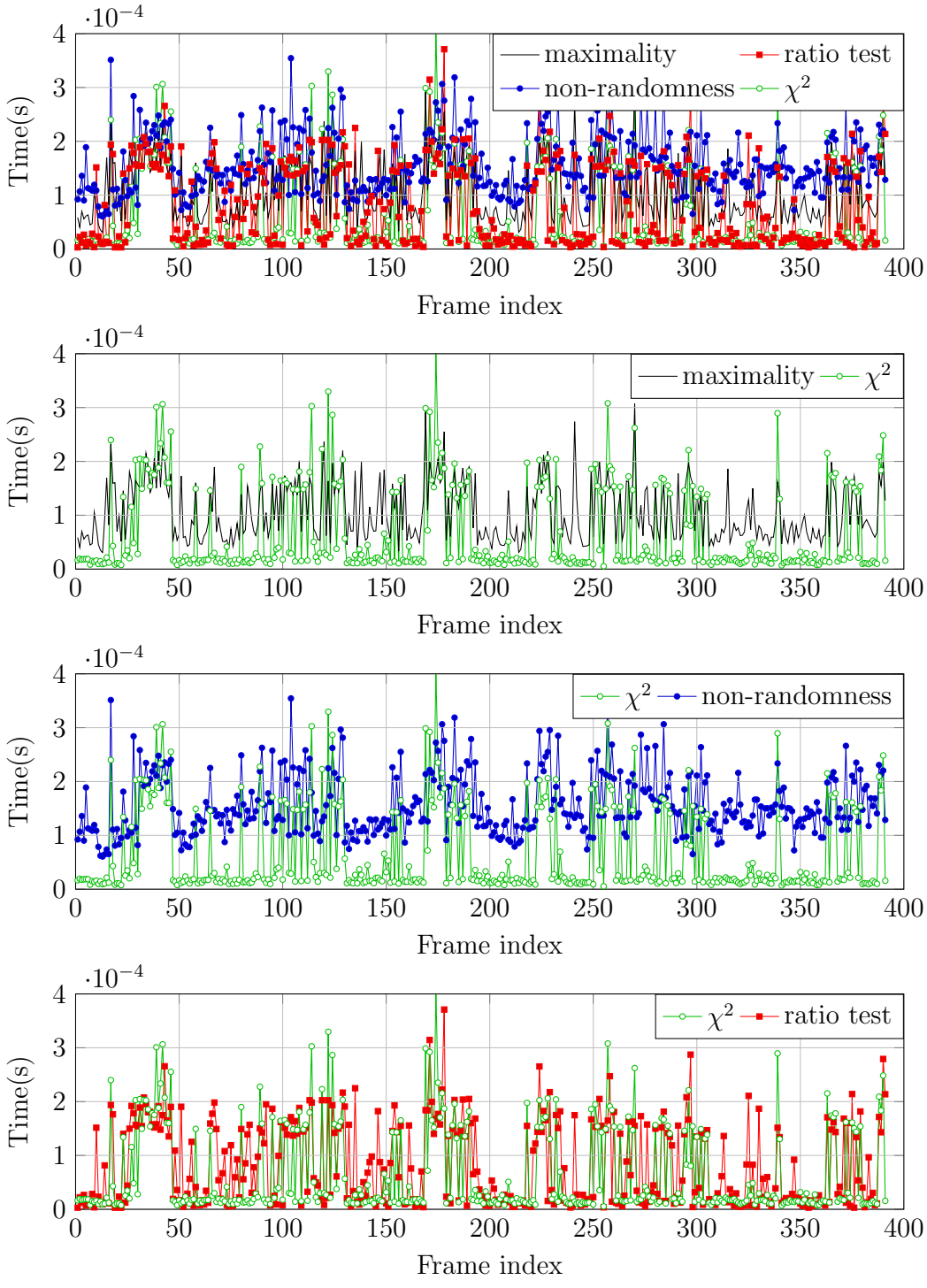


Figure 7.9: Execution time for the homography estimation with different stopping criteria.

Chapter 8

Conclusion

In this thesis, fundamental steps toward a real-time object recognition framework are investigated that yield a reliable Augmented Reality application. The resultant framework has been designed to reliably detect a certain object at high frame rate on mobile platforms. In addition, the position and heading of the object can be recovered through a generic pose estimation algorithm which can be readily employed for other pose models with higher degrees of freedom. The main advantages of our proposed algorithm over other commonly used algorithms, stems from its robustness to high level of uncertainties and the speed of the overall process.

As a consequence of this work, individual developments made in isolation by other researchers are here considered together that gives the reader a broader insight about the conception of object recognition. A hierarchical structure for feature-based detection has been introduced that underlines the importance of feature extraction techniques to boost the performance of the detection. We have also highlighted an offline training strategy that alleviates the computational burden at run-time while preserving the robustness of the algorithm.

As a key element of the recognition framework, we have explored the problem of geometric model parameterization by accounting for real-time constraints. Therefore, we discovered a novel approach for homography estimation that is based on the Gaussian Elimination algorithm. Relying on the results of the thesis, our proposed approach is by far the fastest approach with an accuracy acceptable for posing an augmented reality application.

In this research we have demonstrated that an accurate pose model can be retrieved through a robust parameter estimation scheme based on the RANSAC strategy. We

comprehensively investigated several robust estimation algorithms and analyzed their limitations and advantages in the context of homography estimation. By considering the processing capabilities of modern CPUs, we realized that methods that can be optimized for throughput with streaming SIMD extensions are strongly preferred. So we came up with a highly optimized framework that leverages state-of-the-art algorithms for the robust pose estimation. During this optimization, we have been able to provide theoretical explanations for unexpected behaviors of the process that perhaps only appear in practical implementations. Faster execution time while achieving higher number of verifications, complementary behavior of hypothesize-and-verify scheme to compensate the numerical instability induced by Gaussian Elimination approach and provided analysis behind the χ^2 algorithm are some of these theoretical explanations.

8.1 Future Works

Augmented reality is still under progressive stage and focuses on more compact technologies such as wearable and hand-held devices which are expected to be commonly used in not so distant future. These technologies are not usually brought with powerful processors. In turn, they are equipped with additional components such as IMU, GPS, accelerometers and Gyro sensors that can be come into play for further development of purely vision-based algorithms. For instance, the advantages of vision-based pose estimation algorithms have root in their accuracy and reliability. On the other hand, IMU sensors provide higher frame rate information but at a lower precision compared with visual data. This would be a reason for future investigation of multi-sensory approaches based on the fusion of IMU readings with visual data (aka IMU-vision fusion). Thus, for faster pose estimation without loss of precision, a Kalman filtering technique can be used to provide more accurate estimates at the same rate as IMU readings that are corrected with visual information.

However fairly good performances are achieved by the current feature-based algorithm, there are still serious obstacles toward an impeccable object recognition application. Motion blur can be considered as one of the obstacles that makes stable features almost untraceable. One possible solution for handling the motion blur would be to exploit tracking algorithms when the detection part is unable to tolerate the motion blur. As a consequence of switching between tracking and detection, the user may probably experience undesired augmentation that warrants future researches. Dealing with slightly textured or deformable objects is another obstacle toward reliable tracking of the object.

So a potential solution for these limitations worth exploring in the future would comprise of tracking and patch-wise matching algorithms.

Another interesting extension of this work may be further development of the proposed Gaussian Elimination algorithm for higher-constrained geometric transformation such as fundamental matrix and essential matrix. These extensions would enable us to massively accelerate some expensive processes including 3D reconstruction, auto-calibration and so on. It is worth noting that numerical stability analysis of such algorithms has a significant effect on the overall performance.

Appendix A

Homography Estimation by Gaussian Elimination

This appendix represents implementation of the reduction to reduced-row-echelon form of the following matrix as explained in section 6.0.1.

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0X_0 & -y_0X_0 & X_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1X_1 & X_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -y_3X_3 & X_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0Y_0 & -y_0Y_0 & Y_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1Y_1 & -y_1Y_1 & Y_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3Y_3 & -y_3Y_3 & Y_3 \end{pmatrix} \tag{A.1}$$

We now subtract rows 2 and 6 from the rows 0, 1, 3 and 4, 5, 7 respectively, thus eliminating almost all 1's in column 2 and 5. Since we choose not to scale the rows containing said 1's, they will remain unaffected throughout the remainder of the computation and therefore no storage needs to be reserved for them.

$$\sim \begin{pmatrix} x_0 - x_2 & y_0 - y_2 & 0 & 0 & 0 & 0 & x_2X_2 - x_0X_0 & y_2X_2 - y_0X_0 & X_0 - X_2 \\ x_1 - x_2 & y_1 - y_2 & 0 & 0 & 0 & 0 & x_2X_2 - x_1X_1 & y_2X_2 - y_1X_1 & X_1 - X_2 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -y_2X_2 & X_2 \\ x_3 - x_2 & y_3 - y_2 & 0 & 0 & 0 & 0 & x_2X_2 - x_3X_3 & y_2X_2 - y_3X_3 & X_3 - X_2 \\ 0 & 0 & 0 & x_0 - x_2 & y_0 - y_2 & 0 & x_2Y_2 - x_0Y_0 & y_2Y_2 - y_0Y_0 & Y_0 - Y_2 \\ 0 & 0 & 0 & x_1 - x_2 & y_1 - y_2 & 0 & x_2Y_2 - x_1Y_1 & y_2Y_2 - y_1Y_1 & Y_1 - Y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2Y_2 & -y_2Y_2 & Y_2 \\ 0 & 0 & 0 & x_3 - x_2 & y_3 - y_2 & 0 & x_2Y_2 - x_3Y_3 & y_2Y_2 - y_3Y_3 & Y_3 - Y_2 \end{pmatrix} \tag{A.2}$$

We note here that at this stage, of the 72 potential floating-point values in the matrix, only 32 (excluding the two remaining 1's) are distinct and non-zero. This neatly fits in half of a vector register file with 16 4-lane registers, a common configuration in most modern architectures.

For brevity, after this point only the row operations are given. They were designed to delay the use of reciprocals as long as possible. And the first part is duplicated on both top and bottom half.
First we eliminate column 0 of rows 1 and 3:

$$\begin{aligned} \vec{R}_1 &= r_{0,x} * \vec{R}_1 - r_{1,x} * \vec{R}_0, \qquad \textit{idem on } \vec{R}_5 \\ \vec{R}_3 &= r_{0,x} * \vec{R}_3 - r_{3,x} * \vec{R}_0, \qquad \textit{idem on } \vec{R}_7 \end{aligned}$$

We eliminate column 1 of rows 0 and 3.

$$\begin{aligned} \vec{R}_0 &= r_{1,y} * \vec{R}_0 - r_{0,y} * \vec{R}_1, \qquad \textit{idem on } \vec{R}_4 \\ \vec{R}_3 &= r_{1,y} * \vec{R}_3 - r_{3,y} * \vec{R}_1, \qquad \textit{idem on } \vec{R}_7 \end{aligned}$$

We eliminate columns 0 and 1 of row 2.

$$\begin{aligned} \vec{R}_0 &= \frac{1}{r_{0,x}} * \vec{R}_0, \qquad \textit{idem on } \vec{R}_4 \\ \vec{R}_1 &= \frac{1}{r_{1,y}} * \vec{R}_1, \qquad \textit{idem on } \vec{R}_5 \\ \vec{R}_2 &= \vec{R}_2 - (r_{2,x} * \vec{R}_0 + r_{2,y} * \vec{R}_1), \qquad \textit{idem on } \vec{R}_6 \end{aligned}$$

Columns 0-5 of rows 3 and 7 are zero, and the matrix now resembles this:

$$\begin{pmatrix} 1 & & & & & 0 & a_{06} & a_{07} & a_{08} \\ & \ddots & & & & & a_{16} & a_{17} & a_{18} \\ & & 1 & 0 & & & a_{26} & a_{27} & a_{28} \\ & & 0 & 0 & & & a_{36} & a_{37} & a_{38} \\ & & 0 & 1 & & & a_{46} & a_{47} & a_{48} \\ & & & & \ddots & & a_{56} & a_{57} & a_{58} \\ & & & & & 1 & a_{66} & a_{67} & a_{68} \\ 0 & & & & & 0 & a_{76} & a_{77} & a_{78} \end{pmatrix} \tag{A.3}$$

Let's now cease treating the matrix as two independent 4×9 halves and now consider the rightmost three columns as one 8×3 matrix. We use the barren rows 3 and 7 to eliminate columns 6 and 7, thus:
First, we normalize row 7.

$$\vec{R}_7 = \frac{1}{r_{76}} * \vec{R}_7$$

We eliminate column 6 of rows 0 through 6.

$$\begin{aligned} \vec{R}_0 &= \vec{R}_0 - r_{06} * \vec{R}_7 \qquad \vec{R}_1 = \vec{R}_1 - r_{16} * \vec{R}_7 \qquad \vec{R}_2 = \vec{R}_2 - r_{26} * \vec{R}_7 \\ \vec{R}_3 &= \vec{R}_3 - r_{36} * \vec{R}_7 \qquad \vec{R}_4 = \vec{R}_4 - r_{46} * \vec{R}_7 \qquad \vec{R}_5 = \vec{R}_5 - r_{56} * \vec{R}_7 \\ \vec{R}_6 &= \vec{R}_6 - r_{66} * \vec{R}_7 \end{aligned}$$

We normalize row 3.

$$\vec{R}_3 = \tfrac{1}{r_{37}} * \vec{R}_3$$

We eliminate column 7 of rows 0 through 2 and 4 through 6.

$$\begin{aligned} \vec{R}_0 &= \vec{R}_0 - r_{07} * \vec{R}_3 \qquad \vec{R}_1 = \vec{R}_1 - r_{17} * \vec{R}_3 \qquad \vec{R}_2 = \vec{R}_2 - r_{27} * \vec{R}_3 \\ \vec{R}_4 &= \vec{R}_4 - r_{47} * \vec{R}_3 \qquad \vec{R}_5 = \vec{R}_5 - r_{57} * \vec{R}_3 \qquad \vec{R}_6 = \vec{R}_6 - r_{67} * \vec{R}_3 \end{aligned}$$

The last column of the matrix now contains the homography, normalized by setting $h_{22} = 1$:

$$\sim \begin{pmatrix} 1 & & & & & & & h_{00} \\ & \ddots & & & & & & h_{01} \\ & & 1 & 0 & & & & h_{02} \\ & & 0 & 0 & & & 1 & h_{21} \\ & & 0 & 1 & & & & h_{10} \\ & & & & \ddots & & & h_{11} \\ & & & & & 1 & & h_{12} \\ 0 & & & & & 0 & 1 & h_{20} \end{pmatrix} \rightarrow \begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{pmatrix} \tag{A.4}$$

References

[1] Motilal Agrawal, Kurt Konolige, and MortenRufus Blas. CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching. In *ECCV 2008*, volume 5305 of *Lecture Notes in Computer Science*, pages 102–115. Springer Berlin Heidelberg, 2008.

[2] Sharat Saurabh Akhoury and Robert Laganière. Training Binary Descriptors for Improved Robustness and Efficiency in Real-Time Matching. In Alfredo Petrosino, editor, *ICIAP 2013*, volume 8157 of *Lecture Notes in Computer Science*, pages 288–298. 2013.

[3] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517, June 2012.

[4] Pablo F Alcantarilla, Adrien Bartoli, and Andrew J Davison. KAZE Features. In *European Conference on Computer Vision (ECCV)*, pages 214–227. Springer, 2012.

[5] Pablo F Alcantarilla, Jesús Nuevo, and Adrien Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. *Trans. Pattern Anal. Machine Intell*, 34(7):1281–1298, 2011.

[6] John Aldrich. R. A. Fisher and the Making of Maximum Likelihood 1912-1922. *Statistical Science*, 12(3):162–176, 1997.

[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008.

[8] Hamid Bazargani, Ehsan Omidi, and H Ali Talebi. Adaptive Extended Kalman Filter for Asynchronous Shuttering Error of Stereo Vision Localization. In *Robotics*

and Biomimetics (ROBIO), 2012 IEEE International Conference on, pages 2254–2259. IEEE, 2012.

[9] Albert E. Beaton and John W. Tukey. The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data. *Technometrics*, 16(2):147–185, 1974.

[10] Jeffrey S Beis and David G Lowe. Shape Indexing using Approximate Nearest-neighbour Search in High-dimensional Spaces. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1000–1006. IEEE, 1997.

[11] Matthew Brown and David G Lowe. Automatic Panoramic Image Stitching using Invariant Features. *International Journal of Computer Vision*, 74(1):59–73, 2007.

[12] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In *ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pages 778–792. Springer Berlin Heidelberg, 2010.

[13] Edwin KP Chong and Stanislaw H Zak. *An Introduction to Optimization*, volume 76. John Wiley & Sons, 2013.

[14] O. Chum and J. Matas. Randomized RANSAC with T d,d test. In *IMAGE AND VISION COMPUTING*, pages 448–457, 2002.

[15] O. Chum and J. Matas. Matching with PROSAC - progressive sample consensus. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 220–226 vol. 1, June 2005.

[16] O. Chum and J. Matas. Optimal Randomized RANSAC. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(8):1472–1482, Aug 2008.

[17] Ondej Chum, Ji Matas, and Josef Kittler. Locally Optimized RANSAC. In *Pattern Recognition*, volume 2781 of *Lecture Notes in Computer Science*, pages 236–243. Springer Berlin Heidelberg, 2003.

[18] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, June 2007.

[19] AndrewJ. Davison and DavidW. Murray. Mobile Robot Localisation using Active Vision. In *Computer Vision ECCV98*, volume 1407 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1998.

[20] M. Dhome, M. Richetin, J.-T. Lapreste, and G. Rives. Determination of the Attitude of 3D Objects from a Single Perspective View. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11:1265–1278, Dec 1989.

[21] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[22] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[23] A Gruen and T S Huang (ed). Calibration and Orientation of Cameras in Computer Vision. *Measurement Science and Technology*, 13(2):231, 2002.

[24] R.M. Haralick, D. Lee, K. Ottenburg, and M. Nolle. Analysis and Solutions of the Three Point Perspective Pose Estimation Problem. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 592–598, Jun 1991.

[25] Chris Harris and Mike Stephens. A Combined Corner and Edge Detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[26] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[27] Marko Heikkilä, Matti Pietikäinen, and Cordelia Schmid. Description of Interest Regions with Center-symmetric Local Binary Patterns. In *Computer Vision, Graphics and Image Processing*, pages 58–69. Springer, 2006.

[28] Stefan Hinterstoisser, Selim Benhimane, Vincent Lepetit, Pascal Fua, and Nassir Navab. Simultaneous Recognition and Homography Extraction of Local Patches with a Simple Linear Classifier. In *BMVC*, pages 1–10, 2008.

[29] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle. An Analytic Solution for the Perspective 4-point Problem. In *Computer Vision and Pattern Recognition, 1989.*

Proceedings CVPR '89., IEEE Computer Society Conference on, pages 500–507, Jun 1989.

[30] P.J. Huber. *Robust statistics*. Wiley New York, 1981.

[31] Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[32] Yan Ke and Rahul Sukthankar. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In *Computer Vision and Pattern Recognition, 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE.

[33] JanJ. Koenderink. The Structure of Images. *Biological Cybernetics*, 50(5):363–370, 1984.

[34] V. Lepetit and P. Fua. Keypoint Recognition using Randomized Trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, Sept 2006.

[35] V. Lepetit, J. Pilet, and P. Fua. Point Matching as a Classification Problem for Fast and Robust Object Pose Estimation. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–244–II–250 Vol.2, June 2004.

[36] Vincent Lepetit and Pascal Fua. Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. *Foundations and Trends in Computer Graphics and Vision*, pages 1–89, 2005.

[37] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision*, 81:155–166, 2009.

[38] S. Leutenegger, M. Chli, and R.Y. Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555, Nov 2011.

[39] Gabriele Ligorio and Angelo Maria Sabatini. Extended Kalman Filter-Based Methods for Pose Estimation using Visual, Inertial and Magnetic Sensors: Comparative Analysis and Performance Evaluation. *Sensors*, 13:1919–1941, 2013.

[40] Tony Lindeberg. Scale-space Theory: A Basic Tool for Analysing Structures at Different Scales. *Journal of Applied Statistics*, pages 224–270, 1994.

[41] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[42] D.G. Lowe. Local feature View Clustering for 3D Object Recognition. In *Computer Vision and Pattern Recognition, CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, 2001.

[43] J. Matas and O. Chum. Randomized RANSAC with Sequential Probability Ratio Test. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1727–1732 Vol. 2, Oct 2005.

[44] Jiri Matas and Ondrej Chum. Randomized RANSAC. In *VIENNA UNIVERSITY OF TECHNOLOGY*, pages 49–58, 2002.

[45] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, Oct 2005.

[46] Pablo Mrquez-Neila, Javier Lpez-Alberca, JosM. Buenaposada, and Luis Baumela. Speeding-up Homography Estimation in Mobile Devices. *Journal of Real-Time Image Processing*, pages 1–14, 2013.

[47] S.K. Nayar, S.A. Nene, and H. Murase. Real-time 100 Object Recognition System. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 3, pages 2321–2325, Apr 1996.

[48] D. Nister. Preemptive RANSAC for Live Structure and Motion Estimation. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 199–206 vol.1, Oct 2003.

[49] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast Keypoint Recognition Using Random Ferns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):448–461, March 2010.

[50] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[51] Youngmin Park, V. Lepetit, and Woontack Woo. Multiple 3D Object Tracking for Augmented Reality. In *Mixed and Augmented Reality, ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 117–120, Sept 2008.

[52] P. Perona and J. Malik. Scale-space and Edge Detection using Anisotropic Diffusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(7):629–639, Jul 1990.

[53] Thomas Petersen. A Comparison of 2D-3D Pose Estimation Methods. *Master's thesis, Aalborg University-Institute for Media Technology Computer Vision and Graphics, Lautrupvang*, 15:2750.

[54] Long Quan and Zhongdan Lan. Linear N-point Camera Pose Determination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(8):774–780, Aug 1999.

[55] J.R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[56] R Raguram, O. Chum, M. Pollefeys, J. Matas, and J.M. Frahm. USAC: a Universal Framework for Random Sample Consensus. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):2022–2038, 2013.

[57] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus. In *Proceedings of the 10th European Conference on Computer Vision: Part II*, ECCV '08, pages 500–513, 2008.

[58] Gerhard Reitmayr and Tom W Drummond. Going Out: Robust Model-based Tracking for Outdoor Augmented Reality. In *Mixed and Augmented Reality, ISMAR 2006. IEEE/ACM International Symposium on*, pages 109–118. IEEE, 2006.

[59] Paul L. Rosin. Measuring Corner Properties. *Comput. Vis. Image Underst.*, 73(2):291–307, February 1999.

[60] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515 Vol. 2, Oct 2005.

[61] Edward Rosten and Tom Drummond. Machine Learning for High-Speed Corner Detection. In *ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 430–443. Springer Berlin Heidelberg, 2006.

[62] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, Nov 2011.

[63] C. Schmid and R. Mohr. Combining Greyvalue Invariants with Local Constraints for Object Recognition. In *Computer Vision and Pattern Recognition. Proceedings CVPR '96, IEEE Computer Society Conference on*, pages 872–877, Jun 1996.

[64] S. Se, D. Lowe, and J. Little. Vision-based Mobile Robot Localization and Mapping using Scale-invariant Features. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, pages 2051–2058 vol.2, 2001.

[65] I. Skrypnyk and D.G. Lowe. Scene Modelling, Recognition and Tracking with Invariant Image Features. In *Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on*, pages 110–119, Nov 2004.

[66] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. Graph.*, 25(3):835–846, July 2006.

[67] Charles V. Stewart. Robust parameter estimation in computer vision. *SIAM Reviews*, 41:513–537, 1999.

[68] Ivan E Sutherland. Sketch Pad a Man-machine Graphical Communication System. In *Proceedings of the SHARE Design Automation Workshop*, pages 6–329. ACM, 1964.

[69] S. Taylor, Edward Rosten, and Tom Drummond. Robust Feature Matching in 2.3 μs. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 15–22, June 2009.

[70] Simon Taylor and Tom Drummond. Multiple Target Localisation at over 100 FPS, 2009.

[71] Simon Taylor and Tom Drummond. Binary Histogrammed Intensity Patches for Efficient and Robust Matching. *Int. J. Comput. Vision*, 94(2):241–265, September 2011.

[72] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 128(1):99–141, 2001.

[73] Engin Tola, V. Lepetit, and P. Fua. DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):815–830, May 2010.

[74] B.J. Tordoff and D.W. Murray. Guided-MLESAC: Faster Image Transform Estimation by using Matching Priors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1523–1535, Oct 2005.

[75] P. H. S. Torr and A. Zisserman. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, 78:2000, 2000.

[76] V. Javier Traver and Alexandre Bernardino. A Review of Log-polar Imaging for Visual Perception in Robotics. *Robotics and Autonomous Systems*, 58(4):378 – 398, 2010.

[77] B. Triggs. Camera Pose and Calibration from 4 or 5 Known 3D Points. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 278–284 vol.1, 1999.

[78] Jonathan Ventura, Clemens Arth, Gerhard Reitmayr, and Dieter Schmalstieg. Global localization from monocular SLAM on a mobile phone. *Visualization and Computer Graphics, IEEE Transactions on*, 20(4):531–539, 2014.

[79] Etienne Vincent and Robert Laganiere. An Empirical Study of Some Feature Matching Strategies. *Proc. Conf. Vision Interface, Calgary, Canada*, pages 139–145, 2002.

[80] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose Tracking from Natural Features on Mobile Phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 125–134. IEEE Computer Society, 2008.

[81] Daniel Wagner, Dieter Schmalstieg, and Horst Bischof. Multiple Target Detection and Tracking with Guaranteed Framerates on Mobile Phones. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '09, pages 57–64, 2009.

[82] Abraham Wald. *Sequential Analysis*. John Wiley and Sons, 1st edition, 1947.

[83] Joachim Weickert. Efficient Image Segmentation Using Partial Differential Equations and Morphology. *Pattern Recognition*, 34:2001, 1998.

[84] Changchang Wu, B. Clipp, Xiaowei Li, J.-M. Frahm, and M. Pollefeys. 3D model matching with Viewpoint-Invariant Patches (VIP). In *Computer Vision and Pattern Recognition. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[85] Guoshen Yu and J-M Morel. A Fully Affine Invariant Image Comparison Method. In *Acoustics, Speech and Signal Processing, ICASSP 2009. IEEE International Conference on*, pages 1597–1600. IEEE, 2009.