ADVANCES IN TOP-DOWN AND BOTTOM-UP APPROACHES TO VIDEO-BASED CAMERA TRACKING

THÈSE N^o 3970 (2007)

PRÉSENTÉE LE 6 DÉCEMBRE 2007
À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE TRAITEMENT DES SIGNAUX 1
PROGRAMME DOCTORAL EN INFORMATIQUE, COMMUNICATIONS ET INFORMATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

David MARIMÓN SANJUÁN

Enginyer de Telecomunicació, Universitat Politècnica de Catalunya, Barcelona, Espagne
et de nationalité espagnole

acceptée sur proposition du jury:

Prof. S. Süsstrunk, présidente du jury
Prof. T. Ebrahimi, directeur de thèse
Dr A. Cavallaro, rapporteur
Prof. F. Meyer, rapporteur
Prof. J.-Ph. Thiran, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2007

To the light of my life

Acknowledgements

This thesis is the result of four years of work. There has been ups and downs but fortunately many people have been by my side when I needed their help and support.

First of all, I would like to thank Professor Touradj Ebrahimi, my supervisor. I appreciate the freedom he gave me in my research and the advices I received from him every time I was stuck in a problem. I am especially grateful for the many opportunities he has offered me to meet people from many disciplines and share my research with them. This acknowledge would not be complete if I did not add my gratitude for his trust in me. I am thankful to the members of the jury Dr. Andrea Cavallaro, Prof. Fernand Meyer, Prof. Jean-Philippe Thiran, and Prof. Sabine Süsstrunk, for accepting to be part of the committee and for the comments and challenging discussion we had during the private defence.

Along these years, many people have helped directly in the development of my research. I would like to thank in chronological order Dr. Olivier Steiger, Dr. Yousri Abdeljaoued, Dr. Yannick Maret, Dr. Ulrich Hoffmann, PD.Dr. Michael Ansorge and Dr. Frédéric Dufaux. All their insides and reflections shaped my understanding of scientific investigation.

Part of this thesis was developed in the framework of academic and industrial collaboration. I would like to thank the people involved in the *Variable environment / mobility, interaction city and crossovers* research project. From EPFL, Mr. N. Henchoz and Mr. M. Fernandez, from ECAL, Prof. C. Guignard and Prof. P. Keller, and from fabric | ch, Dr. C. Babski. Their interaction-design spirit has helped me open my mind to different points of view. I am also grateful to Mr. J-M. Nicolas for the nice experience we shared in setting the demonstrator at the Audiorama museum.

During my PhD I had the chance to supervise the diploma project of excellent students. It was a great pleasure to work with P. Berges, B. Palacios and C. Torralbo. It has also been an amazing experience to work with the members of the Signal Processing Institute.

Of course, life outside the lab did also exist. I greet those who I have encountered along this period and with whom I have shared very special moments. I am specially grateful to our mini-family of Spanish friends. You are just terrific!

There are some moments in life when it is important to look back and see where we come from. This work would not have been possible without the support of my dear family and friends in Barcelona.

Everything that is explained in this thesis is based on the existence of light. There is a light that has guided me throughout the process, during the past four years. That light is named Cecilia.

Abstract

Video-based camera tracking consists in trailing the three dimensional pose followed by a mobile camera using video as sole input. In order to estimate the pose of a camera with respect to a real scene, one or more three dimensional references are needed. Examples of such references are landmarks with known geometric shape, or objects for which a model is generated beforehand. By comparing what is seen by a camera with what is geometrically known from reality, it is possible to recover the pose of the camera that is sensing these references.

In this thesis, we investigate the problem of camera tracking at two levels. Firstly, we work at the low level of feature point recognition. Feature points are used as references for tracking and we propose a method to robustly recognise them. More specifically, we introduce a rotation-discriminative region descriptor and an efficient rotation-discriminative method to match feature point descriptors. The descriptor is based on orientation gradient histograms and template intensity information. Secondly, we have worked at the higher level of camera tracking and propose a fusion of top-down (TDA) and bottom-up approaches (BUA). We combine marker-based tracking using a BUA and feature points recognised from a TDA into a particle filter. Feature points are recognised with the method described before. We take advantage of the identification of the rotation of points for tracking purposes. The goal of the fusion is to take advantage of their compensated strengths. In particular, we are interested in covering the main capabilities that a camera tracker should provide. These capabilities are automatic initialisation, automatic recovery after loss of track, and tracking beyond references known a priori.

Experiments have been performed at the two levels of investigation. Firstly, tests have been conducted to evaluate the performance of the recognition method proposed. The assessment consists in a set of patches extracted from eight textured images. The images are rotated and matching is done for each patch. The results show that the method is capable of matching accurately despite the rotations. A comparison with similar techniques in the state of the art depicts the equal or even higher precision of our method with much lower computational cost. Secondly, experimental assessment of the tracking system is also conducted. The evaluation consists in four sequences with specific problematic situations namely, occlusions of the marker, illumination changes, and erratic and/or fast motion. Results show that the fusion tracker solves characteristic failure modes of the two combined approaches. A comparison with similar trackers shows competitive accuracy. In addition, the three capabilities stated earlier are fulfilled in our tracker, whereas the state of the art reveals that no other published tracker covers these three capabilities simultaneously.

The camera tracking system has a potential application in the robotics domain. It has been successfully used as a man-machine interface and applied in Augmented Reality environments. In particular, the system has been used by students of the University of art and design Lausanne (ECAL) with the purpose of conceiving new interaction concepts. Moreover, in collaboration with ECAL and fabric | ch (studio for architecture & research), we have jointly developed the Augmented interactive Reality Toolkit (AiRToolkit). The system has also proved to be reliable in public events and is the basis of a game-oriented demonstrator installed in the Swiss National Museum of Audiovisual and Multimedia (Audiorama) in Montreux.

Keywords camera tracking, data fusion, top-down approach, bottom-up approach, pattern recognition, histogram matching, template matching.

Version abrégée

Le suivi d'une camera avec la vidéo consiste à suivre la pose tridimensionnelle d'une caméra mobile en n'utilisant que la vidéo comme entrée. Pour permettre l'estimation de la pose de la camera par rapport à une scène réelle, une ou plusieurs références tridimensionnelles sont nécessaires. Deux exemples de références sont: des marqueurs avec une forme pré-definie, et des objets à partir desquels on a généré un modèle auparavant. En comparant ce qui est vu par la camera avec ce qui est connu géométriquement dans la réalité, il est possible de retrouver la pose de la camera qui observe ces références.

Dans cette thèse, nous investigons la problématique du suivi de la camera à deux niveaux. Premièrement, nous travaillons dans la reconnaissance de points d'intérêt. Les points d'intérêt sont utilisés comme références pour le suivi et nous proposons de les reconnaitre de façon robuste. Plus précisément, nous introduisons un descripteur de region discriminant à la rotation et une méthode efficace et aussi discriminante à la rotation pour trouver la correspondance entre les points d'intérêt. Ces descripteurs sont basés sur les histogrammes d'orientation du gradient et sur l'information d'intensité. Deuxièmement, nous travaillons au niveau du suivi de la camera et proposons la fusion d'une approche top-down (TDA) et d'une approche bottom-up (BUA). Nous combinons un traqueur de marqueurs utilisant une BUA avec la reconnaissance des points d'intérêt cherchés avec une TDA dans un filtre à particules. Les points d'intérêt sont reconnus avec la méthode décrite avant. Nous profitons de l'identification de la rotation des points pour le suivi de la camera. Le but de la fusion est de compenser les faiblesses individuelles tout en profitant des avantages de chacun. En particulier, nous souhaitons développer un système qui soit à la fois capable de s'initialiser automatiquement, de rétablir sa trajectoire en cas de perte et ayant une zone étendu à des régions sans référence connues a priori.

Des expériences ont été faites aux deux niveaux d'investigation. Premièrement, des tests ont été faits pour évaluer la performance de la méthode de reconnaissance. L'évaluation consiste à un ensemble de petites régions d'image extraites de huit images texturées. Les images sont tournées et la correspondance est estimée pour chaque région. Les résultats montrent que la méthode est capable de faire la correspondance de façon précise même après les rotations. Une comparaison avec des techniques similaires dans l'état de l'art montre une performance égale ou même supérieure avec un coût computationnel très inferieur. Deuxièmement, nous faisons aussi des expériences avec le système de suivi. L'évaluation consiste en quatre séquences décrivant des situations spécifiques différentes: occlusions du marqueur, changement d'illumination, et mouvement erratique ou rapide.

Les résultats montrent que la fusion compense les erreurs caractéristiques des deux approches combinées. Une comparaison avec des traqueurs dans l'état de l'art montre une précision competitive. De plus, les trois capacités citées avant sont accomplies par notre traqueur. Pourtant l'état de l'art montre qu'il n'y a pas d'autre traqueur publié qui couvre toutes ces capacités en même temps.

Le système de suivi de la camera a une application potentielle dans le domaine de la robotique. Il a été utilisé en tant qu'interface homme-machine avec succès et appliqué à des environnements de Réalité Augmentée. En particulier, le système a été utilisé par des étudiants de l'Ecole Cantonale d'Art de Lausanne (ECAL) pour concevoir des nouveaux concepts d'interaction. De plus, en collaboration avec l'ECAL et fabric | ch (studio d'architecture & recherche), nous avons développé le 'Augmented interactive Reality Toolkit' (AiRToolkit). Le système a été aussi testé dans des événements publics et il est la base d'un démonstrateur de type jeux installé dans Le Musée National Suisse de l'Audiovisuel (Audiorama) à Montreux.

Mots clés suivi de camera, fusion de données, approche top-down, approche bottom-up, reconnaissance de patron, correspondance d'histogrammes.

Contents

1 Introduction 1

1.1 Motivations 1
1.2 Investigated approach 2
1.3 Major contributions 3
1.4 Organisation of the thesis 4

I Region recognition 7

2 Background and state of the art 9

2.1 Introduction 9
2.2 Template and distribution descriptors 10
  2.2.1 Templates 10
  2.2.2 Distribution descriptors 11
2.3 Matching strategies 15
  2.3.1 Point correspondence 15
  2.3.2 Line-search and trust-region 15
  2.3.3 Window-search 16
2.4 Similarity measures 18
2.5 Summary 18

3 Rotation-discriminative template matching 21

3.1 Introduction 21
3.2 Rotation-discriminative region descriptor 22
3.3 Efficient rotation-discriminative matching 24
  3.3.1 Gradient magnitude matching 25
  3.3.2 Histogram matching 25
  3.3.3 Template matching 26
3.4 Computational complexity 26
3.5 Experiments 27
  3.5.1 Test set 28
  3.5.2 Evaluation of the correlation accuracy 30

3.5.3 Evaluation of the orientation accuracy 31
3.5.4 Comparison with similar techniques 33
3.6 Application to top-view visual tracking 38
3.7 Summary 39

II Video-based camera tracking 41

4 Theoretical background 43
4.1 Tracking 43
4.1.1 Tracker properties 44
4.1.2 Tracker devices 45
4.2 Camera geometry 47
4.2.1 Camera internal parameters 50
4.2.2 Parameterisation of the rotation 50
4.3 Bayesian filtering 53
4.3.1 Kalman filter 54
4.3.2 Particle filter 55
4.4 Data fusion 57
4.5 Summary 59

5 State of the art 61
5.1 Introduction 61
5.2 Bottom-up approaches (BUA) 62
5.2.1 Marker-based tracking 62
5.2.2 Markerless tracking 64
5.3 Top-down approaches (TDA) 70
5.3.1 Filtering 71
5.3.2 Data association 72
5.4 Hybrid tracking 75
5.5 Comparison of approaches 76
5.6 Summary 79

6 Camera tracking combining a TDA and BUA using markers and feature points 81
6.1 Introduction 81
6.2 System description 83
6.2.1 Particle filter 83
6.2.2 Marker cue 85
6.2.3 Feature points cue 87
6.2.4 Adding feature points 89
6.2.5 Cues combination 94
6.2.6 Dynamic tuning of the filter 98
6.3 Experiments 100

6.3.1 Methodology 100
6.3.2 Evaluation of the combination 104
6.3.3 Evaluation of the system 107
6.3.4 Comparison with other techniques 111
6.3.5 Discussion 113
6.4 Summary 113

7 Applications **115**
7.1 Introduction 115
7.2 Human-machine interfaces and augmented reality 116
7.2.1 Augmented machinery inspection 116
7.2.2 Collaboration with University of art and design Lausanne (ECAL) 118
7.2.3 Transfer of technology 121
7.3 Robot navigation and visual servoing 121
7.4 Summary 122

III Conclusions and future work **123**

8 Conclusions **125**
8.1 Summary of achievements 125
8.2 Perspectives 127

IV Appendix **129**

A Complementary results on rotation-discriminative template matching **131**

Bibliography **137**

Curriculum Vitae **149**

Acronyms

BUA	Bottom-Up Approach
TDA	Top-Down Approach
MC	Marker Cue
FPC	Feature Point Cue
DoF	Degrees of Freedom
FoV	Field of View
SLAM	Simultaneous Localisation and Mapping
MCL	Monte Carlo Localisation
AR	Augmented Reality
RDTM	Rotation-Discriminative Template Matching
KF	Kalman Filter
EKF	Extended Kalman Filter
PF	Particle Filter
PDF	Probability Density Function
NCC	Normalised Cross Correlation
HMD	Head Mounted Display

Introduction

1

1.1 Motivations

Tracking consists in the estimation of the motion trajectory of a sensor or object. When this trajectory is described in the three dimensions of space, this process is generally known as 3D tracking. Several sensors relying on different principles such as magnetic fields or acoustic waves, among others, provide accurate estimates. The drawback of these technologies is often the economic cost and the limited mobility permitted to a potential user of these trackers. The first drawback contrasts with an increasing number of tracking sensors available at a reduced cost. For instance, cameras are currently integrated in portable devices such as mobile phones or laptops. The mobility drawback is relevant in a society that is getting used to communication without wires and, most of the time, without borders. As a consequence, cheaper devices already available should be exploited and mobility limitations of such tracking sensors should be overcome.

Visual perception or vision is the ability to interpret the surrounding light information through an optical system. This interpretation is natural for human beings. However, performing the same task with a machine is not straightforward. The research area dedicated to investigate this task is commonly known as computer vision.

Video-based camera tracking, from now on simply referred to as camera tracking, is a subset of 3D tracking where the sensor is a camera and the principle is based on computer vision techniques. In this case, visual information is used to track the six degrees of freedom (DoF) of the camera pose, three for position and three for orientation. In order to estimate the pose of a camera with respect to a real scene, a three dimensional reference or references are needed. Examples of such references are landmarks with known geometric shape, or objects for which a model is generated beforehand. By comparing what is seen by a camera with what is geometrically known from reality, it is possible to recover the pose of the camera that is sensing these references.

Camera tracking drives currently a growing interest in the research and industry communities.

Indeed, cheaper equipments are available and faster CPUs enable real-time processing previously unimaginable. Older methods that could only rely on hardware implementations can now be more flexibly developed. With new possibilities, new challenges arise and hence novel approaches are needed.

Among the various problems related to camera tracking, the research community has identified the main issues that should be solved. Some of these issues are related to common problems of using video as input such as occlusions, viewpoint and illumination conditions. Indeed, the references of the real scene mentioned before must be detected in order to determine the camera pose. These problems could deteriorate the detection or even make it impossible. Other issues are related to tracking namely, automatic initialisation, automatic recovery after loss of track, or tracking beyond known references. From a user point of view, a camera tracker should be ready for use with the least possible preparation of the environment. Ideally, a tracker should be able to start tracking as soon as the reference is detected. Furthermore, if the references used by the tracker are not detected, the system should have a method to automatically re-initialise the track. Another aspect that should be addressed is that of extending the trackable area beyond references known a priori. The tracker should be able to incorporate and rely on previously unknown references.

Most researchers concentrate their efforts on particular issues either related to vision or to tracking. However, little research has been done on solving those issues at the same time on a single camera tracker.

1.2 Investigated approach

In this thesis, we have investigated how to address the problems cited before in a unique framework. Our research is divided in two areas. On one side, aspects related to tracking namely, automatic initialisation, automatic recovery after loss of track, or tracking beyond known references, are studied. On the other side, we aim to solve problems related to vision. These problems are occlusions, viewpoint and illumination conditions.

Tracking can be performed from a bottom-up or from a top-down approach. Bottom-Up Approaches (BUAs) address the problem of camera tracking by formulating the following question: *from what I see, can I estimate my position?* This means that BUAs detect references and try to infer the pose from their back-projection in the image plane. Top-Down Approaches (TDAs) address the camera pose problem by asking: *from my position, do I see what is expected?* In other words, the tracker keeps an estimate of its position and tries to detect references where it expects them to be.

Both approaches have advantages and disadvantages. In the case of BUAs, an advantage of looking for a reference with little or no knowledge of the camera pose is that the initialisation and re-initialisation problem are directly solved. A disadvantage arises when the reference is not detected. In this case, the tracker cannot even distinguish whether the reference is in front of the camera or simply outside of the camera's field of view (FoV). In the case of TDAs, an advantage of keeping a time coherent estimate of its pose is that occlusions of the reference do not break the track. When a reference is momentarily undetected, the tracker can provide an estimate of the trajectory given a motion model and previously corrected estimates. Depending on the accuracy of the motion model, chances are high that the estimate of the tracker during the occlusion of the reference is close to camera's real motion. One disadvantage of TDAs is that the pose of the camera is assumed to be known. This assumption collides with the need of automatic initialisation. Another assumption of TDAs is that the estimate of the camera pose is accurate. A problem is originated if inaccurate estimates are accumulated. In this situation, the tracker drifts and possibly ends loosing its track. Furthermore, recovery after this situation is impossible because TDAs assume that pose is known.

In this thesis, we investigate how to overcome individual problems with a fusion of top-down and bottom-up approaches. An analysis of the state of the art in camera tracking permits to identify a potential combination that can solve characteristic individual weaknesses and take advantage of the strengths. The particular approaches identified are described next. As a Bottom-Up Approach (BUA), a tracker relying on a known squared reference called marker is chosen. The tracker detects a marker and produces an estimate of the camera pose at each frame. This enables automatic initialisation and recovery. As a Top-Down Approach (TDA), a filter-based tracker relying on natural feature points is taken. The tracker searches for feature points according to their 3D position in the scene and the camera pose predicted according to a motion model. Each feature point localised contributes additively to correct the prediction. Moreover, the system is provided by a dynamic mapping of feature points in the environment. This allows occlusions of the marker and hence an extended trackable area.

Among the different possible configurations of sensor or data fusion, we focus on a low-level fusion of approaches. Indeed, we defend that a tracker can benefit from combining different cues rather than combining directly the outputs of different trackers. We tackle this low-level fusion by merging pose data into a single filter.

Let us now discuss the aspects related to vision. Feature points are used as a cue to constrain the camera pose estimation of the investigated fusion tracker. Tracking based on feature points' recognition confronts occlusion, viewpoint and illumination changes. Occlusions are handled directly by the filter. Viewpoint and illumination changes have to be addressed with computer vision techniques. In order to recognise feature points, these are generally described with the information of the neighbouring region of pixels.

In this thesis, we investigate how to robustly recognise small patches of pixels that have undergone rotations or illumination changes. More specifically, we concentrate on 2D rotations of a patch with respect to the normal of the surface they represent. Using gradient information, it is possible to determine the rotation that a patch has undergone. Once the rotation is estimated, more accurate matching can be achieved. Based on this idea, we develop an efficient method to search for feature point matches.

1.3 Major contributions

The significant contributions of the work presented in this dissertation are summarised below.

- Specification, conception, development and performance evaluation of a fusion of a top-down and a bottom-up approach for camera tracking. The tracking system proposed uses a filter to keep track of the camera's pose. This filter is updated using two cues, one given by a marker-based tracker (BUA) and another one based on feature points searched with a TDA. An experimental evaluation is conducted to prove the synergy of capabilities achieved by the fusion framework. The results show that automatic initialisation, automatic recovery after loss of track, and tracking beyond known references are possible. A comparison with the state of the art reveals that no other published tracker covers these capabilities at the same time.
- Successful use of the camera tracking framework as a human-machine interface and application in Augmented Reality environments. The system has been used by students of the University of art and design Lausanne (ECAL) with the purpose of conceiving new interaction concepts. The system has also proved to be reliable in public demonstrations.
- In collaboration with ECAL and fabric | ch (studio for architecture & research), joint development of the Augmented interactive Reality Toolkit (AiRToolkit).
- Specification, conception and development of a combination of different sorts of visual references for camera tracking. Fiducial markers and natural feature points are used as references for camera pose estimation.
- Specification, conception, development and performance evaluation of a method to map unknown feature points. Feature points unknown to the tracking system are dynamically added by estimating their 3D position in the real world using the camera motion. Experiments are conducted to show the improvement with respect to similar methods in the state of the art.
- Specification, conception, development and performance evaluation of a method to dynamically tune the filter. An algorithm to adapt the motion model of the filter to the current motion of the camera is proposed. Experiments show that tuning each dimension of the state-space independently achieves better results.
- Specification, conception, development and performance evaluation of a rotation-discriminative region descriptor and an efficient rotation-discriminative matching method. A descriptor is proposed based on gradient and grey-level pixel information. This descriptor permits to discern if the region it describes has undergone a quantised 2D rotation. A method to match the descriptor exhaustively and efficiently is proposed. Experimentation with the descriptor and the matching method shows similar or even higher accuracy with much lower computational cost when compared to similar techniques in the state of the art.

1.4 Organisation of the thesis

This thesis is organised as follows.

Part I is dedicated to region recognition. The state of the art in matching of region descriptors is first discussed in Chapter 2. Chapter 3 describes our method for matching textured regions.

Although this method is further used for feature point recognition in the camera tracking system, this chapter presents the method from a generic point of view.

Part II is devoted to the camera tracking system. Chapter 4 explains the concepts encompassed by the tracking framework. More concretely, 3D tracking technologies, camera geometry, Bayesian filtering, and data fusion are discussed. Chapter 5 surveys the state of the art in video-based camera tracking. The tracking system is fully discussed in Chapter 6. Firstly, the combination of TDA and BUA using markers and feature points is depicted. This description includes the method to map unknown feature points, as well as the method to dynamically tune the motion model. This is followed by an evaluation of its performance, which is conducted from two points of view. On one side, the fusion is faced to situations where either one or the other merged approach fails. On the other side, the different assets of the system are tested. Indeed, the region recognition method integrated in the tracking framework is evaluated. Furthermore, the mapping and the dynamic tuning are also compared to the state of the art. To finalise Part II, applications that use the proposed system and also potential applications are explained in Chapter 7. In particular, some results of the collaboration with ECAL are described.

Part III closes this dissertation. In Chapter 8, conclusions drawn from this work are summarised and future possible research extensions are proposed based on the work done.

Part I

Region recognition

Background and state of the art

2

Region recognition is the task of identifying an image region, also called image patch, inside an image. This identification is done by matching a description of the patch, with the description of patches extracted at different points in the tested image.

This chapter is devoted to analysing the information that can be extracted from image regions in order to generate a description. In addition, recognition methods developed in the research community are also discussed. In particular, those methods related to the matching algorithm proposed in the following chapter are detailed.

This chapter is structured as follows. Firstly, recognition methods are classified. Secondly, Section 2.2 explains template and distribution descriptors. Thirdly, matching strategies are dealt with in Section 2.3. Finally, a brief note on similarity measures is given.

2.1 Introduction

Recognition or matching of image information is at the core of applications such as object tracking (e.g., [ST94, Lew95, CH96, HB96, FT97, JD02a, PHVG02, CRM03, DGBD05, ARS06]) and camera tracking (e.g., [ZC93, Dav03, MDR04, PC05, SEGL05]), among others. In such applications, the information extracted from a first image is sought in a second one. For instance, a car detected at the beginning of a video sequence is to be detected along the following frames of that video sequence. Recognition has to face two main problems in general. Firstly, the viewpoint of the image from which the original information is obtained is not necessarily the same as the viewpoint of the image where it is sought. This means that the information could have changed. The second problem is illumination. Since images are representations of light, changes of the illumination conditions from one image to another might also influence the recognition process. These are the main focus, either together or separately, of the related research.

Recognition techniques can be separated in two categories: trained and non-trained.

For the trained category, classifiers are trained with a test set of positive and negative patch examples. Research is concentrated on the data set and the classification techniques. These techniques provide an excellent compromise between computational complexity and accuracy at runtime (e.g., [VJ01, LPF04]). However, the time consumed to gather or generate the training data and train the classifier is generally high. This category is not detailed here because it is not directly related to the proposed method. The reader is referred to [Bis06] for more details.

For the non-trained category, recognition is done by comparing the descriptor of a patch with the descriptors obtained at different locations in the image. This process can be described mathematically as follows. Given a patch (or region) **P** and the descriptor of this patch $f(\mathbf{P})$, the similarity of **P** with an image **I** at point (x,y) is

$$d(f(\mathbf{P}), f(\mathbf{R}_{x,y})) \tag{2.1}$$

where $f(\mathbf{R}_{x,y})$ is the descriptor of the neighbourhood region $\mathbf{R} \subset \mathbf{I}$ centered at (x,y), and $d(\cdot,\cdot)$ is a measure of similarity to compare descriptors. In most cases, **R** has the same size as **P**. In this category, attention is paid to the description $f(\cdot)$ of the information rather than the training data or the classification scheme used. The description of a region determines in great measure the robustness of a recognition process facing viewpoint and illumination changes. Consequently, most researchers concentrate their efforts on obtaining invariant descriptors. Contrary to trained methods, descriptors are generally built from a single instance of the patch to recognise. The drawback of such invariance is often a higher computational cost during the matching process.

Mikolajczyk and Schmid [MS05] classify the descriptors among the following categories: templates [ZC93, ST94, Lew95, CH96, HB96, FT97, JD02a], distributions [HSD73, HKM$^+$97, Bir98, RTG00, HGN01, PHVG02, CRM03, LC04, Low04, GM04, LSP05, BR05, ARS06], Fourier [FKCK05] and Gabor transform [BdB94], image derivatives [SM97], oriented (or steerable) filters [FA91, CJ02] and generalised moment invariants [VMU96].

Among these descriptors and the recognition strategies used in those works, some have been chosen because of their special relation to the method proposed here, and will be explained more in depth hereafter.

2.2 Template and distribution descriptors

Two descriptors have been used extensively for recognition purposes and, more specifically, in tracking applications [DGBD05]. These descriptors are based on templates and distributions.

2.2.1 Templates

Templates are ordered arrays of the pixel values of a region. Mathematically expressed, this means $f(\mathbf{P}) = \mathbf{P}$. Templates are generally compared at a pixel-wise level. In other words, the value (or values in the case of a multichannel image) of the pixels is compared one by one. A common comparison between templates is the cross correlation.

$$d(f(\mathbf{P}), f(\mathbf{R}_{x,y})) = \sum_{i=0}^{W-1}\sum_{j=0}^{H-1} \mathbf{P}(i,j)\mathbf{R}_{x,y}(i,j) = \sum_{i=-W/2}^{W/2}\sum_{j=-H/2}^{H/2} \mathbf{P}(i,j)\mathbf{I}(x+i, y+j), \tag{2.2}$$

where W and H are the width and height of the template, respectively. The second equality is valid for an odd-sized region, assuming that the divisions are given in integer values. Templates have two main advantages. Firstly, the simplicity of construction of this descriptor. Secondly, the spatial information of the region is kept. The counterpart of this advantage is the high sensitivity to viewpoint and illumination changes.

Several improvements of this simple matching technique exist in literature [SWB92, HB96, JD02a, MDR04]. Shapiro *et al.*[SWB92] introduced the *product moment coefficient* for images, which resolves the problem of illumination invariance. The idea is to compare the structure of the templates instead of the actual values and in this way, achieve invariance to linear illumination changes. This coefficient is generally known as the Normalised Cross Correlation (NCC) coefficient and can be expressed as follows

$$NCC(\mathbf{P}, \mathbf{R}_{x,y}) = \frac{\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}\left(\mathbf{P}(i,j) - \overline{\mathbf{P}}\right)\cdot\left(\mathbf{R}_{x,y}(i,j) - \overline{\mathbf{R}}_{x,y}\right)}{\sqrt{\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}\left(\mathbf{P}(i,j) - \overline{\mathbf{P}}\right)^2 \cdot \sum_{i=0}^{W-1}\sum_{j=0}^{H-1}\left(\mathbf{R}_{x,y}(i,j) - \overline{\mathbf{R}}_{x,y}\right)^2}}, \tag{2.3}$$

where $\overline{\mathbf{R}}$ is the average value of $\mathbf{R}$. By subtracting this mean value, the result is invariant to illumination changes. This is true under the assumption that the represented surface is Lambertian and that the illumination incident on the surface can be locally approximated by a constant. Nevertheless, good experimental results have been obtained on surfaces that do not fit this assumptions [RDLW95].

However, this technique still lacks viewpoint robustness. Several works explore the parameterisation of the geometrical transformation that a patch suffers, increasing in this way the robustness to rotation, translation and also scale changes [HB96, JD02a, MDR04]. Since this matching method is done in the context of object or camera tracking, a description of these works is given in Chapter 5.

2.2.2 Distribution descriptors

Distribution descriptors are arrays containing a discrete distribution of the information of a region. A widely used distribution descriptor is the *histogram*. A histogram is an array that models the true distribution by counting the occurrences of pixel values that fall into each bin (which encompasses a range of values). This can be formulated as explained next. Assuming b is a function that assigns its argument to the quantised space of bins, and δ is the Kronecker delta function, one obtains the expression for the histogram of a certain magnitude L computed from $\mathbf{P}$ in the space $\mathbb{S}$:

$$\begin{aligned} b &: \mathbb{S} \rightarrow \{0, \ldots, N-1\} \\ h_{L(\mathbf{P})}(n) &= \sum_{i,j\in\mathbf{P}} \delta(\, b\,(L(\mathbf{P}(i,j)))\,) - n\,) \quad n = 0, \ldots, N-1 \qquad (2.4) \\ f(\mathbf{P}) = \mathbf{h}_{L(\mathbf{P})} &= [h(0), \ldots, h(N-1)]_{L(\mathbf{P})}. \qquad (2.5) \end{aligned}$$

Different information can be used for histogram descriptors. For instance, $L(\mathbf{P})$ can be constructed from a gray-scale version of the patch [ARS06]. Other examples are the colour information [Bir98, PHVG02, CRM03, LC04] and the gradient [Low04, ME07b].

Histograms have opposite advantages and drawbacks when compared to templates. More concretely, histograms loose spatial information while viewpoint invariance can be achieved by construction. Several attempts at combining spatial and distribution information exist, e.g., [HSD73, HKM$^+$97, HGN01, CRM03, Low04, GM04, BR05, ARS06].

Comaniciu *et al.*[CRM03] use an isotropic kernel, with a convex and monotonic decreasing kernel profile $k(u) : \mathbb{R} \rightarrow [0,1]$ that weights the contribution of pixels to the histogram.

$$k(u) = \begin{cases} 1-u & 0 \leq u \leq 1 \\ 0 & \text{otherwise}. \end{cases} \tag{2.6}$$

One advantage of this kernel is that the influence of peripheral pixels is lessened. Peripheral pixels are the least reliable, being often affected by occlusion and background (for instance, in tracking environments) and viewpoint changes (for instance, rotations). The resulting histogram can be formulated as follows. Let $\{\mathbf{x}_i^*\}_{i\in\mathbf{L}}$ be the pixel locations of the patch $\mathbf{P}$ with respect to the center of the patch. Defining a radius r of the region described, one obtains

$$h_{\mathbf{L}}(n) = \sum_{i\in\mathbf{L}} k\left(\left\|\frac{\mathbf{x}_i^*}{r}\right\|^2\right) \cdot \delta\left(\, b\left(\mathbf{L}(\hat{x}_i)\right) - n\,\right] \quad n = 0,\ldots,N-1 \tag{2.7}$$

Georgescu and Meer [GM04] propose a similar approach. In particular, the problem of rotation is directly addressed. The authors use four kernels with an Epanechnikov profile similar to that of Equation (2.6). The difference is that each kernel is oriented at a different angle, namely 0, 45, 90 and 135 degrees depicted in Figure 2.1. This provides robustness in front of these discrete rotation angles.

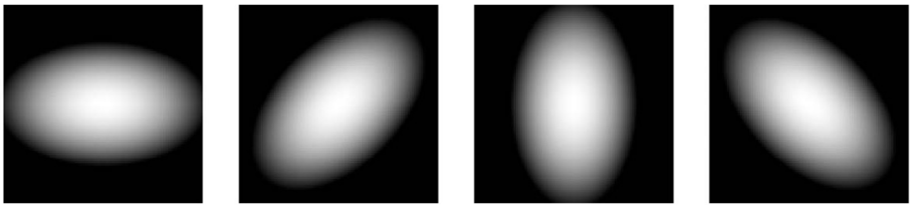


Figure 2.1: Oriented kernels to weight pixel contribution to the histogram descriptor [GM04]. Orientations at 0, 45, 90 and 135 degrees.

Lowe [Low04] uses the spatial distribution of gradient histograms in what is called Scale Invariant Feature Transform (SIFT). The construction of SIFT descriptors is complex and only a brief explanation is given hereafter. SIFT descriptors are used to represent interest or feature points of an image, i.e., points found at strong natural edges or corners. This is opposed to the descriptors explained before that can be computed for any region of an image. Firstly, a scale-space selection of feature points in $\mathbf{I}$ is performed. This gives points with strong saliency in the gradient space.

Secondly, the orientation of the region to describe is estimated. This estimation proceeds as described next. The image used for this estimation is a smoothed version of the original image. The smoothing scale is that where the interest point was detected. The orientation is computed from the gradient information of the smoothed image $\hat{\mathbf{I}}$

$$\begin{aligned} dy(x,y) &= \hat{\mathbf{I}}(x,y+1) - \hat{\mathbf{I}}(x,y-1) \\ dx(x,y) &= \hat{\mathbf{I}}(x+1,y) - \hat{\mathbf{I}}(x-1,y) \\ \nabla_m(x,y) &= \sqrt{dy(x,y)^2 + dx(x,y)^2} \\ \nabla_\theta(x,y) &= \arctan\big(dy(x,y), dx(x,y)\big), \end{aligned} \tag{2.8}$$

where $\arctan(a,b)$ is a function that returns the inverse tangent of a/b in a range $[0,2\pi[$, ∇_m is the magnitude and ∇_θ is the orientation of the gradient. A 36-bin orientation histogram is formed from the gradient orientations of points within a region around the feature point. Each sample (x,y) added to the histogram is weighted by its magnitude $\nabla_m(x,y)$ and by a Gaussian-weighted circular window $\mathbf{G}$ with a σ that is 1.5 times that of the scale of the feature point (this is equivalent to the kernel used by Comaniciu *et al.*[CRM03]):

$$\begin{aligned} b &: [0,2\pi[\rightarrow \{0,\dots,35\} \\ h_{\nabla_\theta}(n) &= \sum_{i,j\in\mathbf{P}} \mathbf{G}(i,j)\cdot\nabla_m(i,j)\cdot\delta(\, b(\nabla_\theta(i,j)) - n\,] \quad n = 0,\dots,35. \end{aligned} \tag{2.9}$$

The highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a different descriptor with that orientation. Finally, a parabola is fit to the 3 histogram values closest to the peak and the maximum of this parabola determines a more accurate peak position. This results in the dominant orientation. In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to this dominant orientation. The descriptor is built following the structure depicted in Figure 2.2. For each square subregions, an orientation histogram is built. Besides further refinements detailed in [Low04], the descriptor of a feature point is a vector containing all the histograms (one per subregion). Lowe's experimentation shows that the best results are achieved with regions of 16x16 pixels (4x4-pixel subregions) and histograms of 8 bins. The size of the vector descriptor is hence 4x4x8 = 128. One of the achievements of SIFT is a high rotation invariance, gained partially by pre-computing the dominant direction and normalising the histograms accordingly.

The descriptor of Adam *et al.*[ARS06] also relies on the idea of computing histograms for subregions of a patch. Each histogram is in this case computed from the grey-scale information. Similarly to Lowe's approach, the arrangement of the patches overcomes the loss of spatial knowledge in the histograms. A particularity of this approach is that matching is performed individually on each subregion histogram. A match for the complete patch **P** is considered when enough individual support is achieved. This descriptor is especially interesting to handle partial occlusions of the patch.

Other examples of integration of spatial information into statistical descriptors exist in literature. For instance, the co-occurrence matrices [HSD73], colour correlograms [HKM$^+$97] and

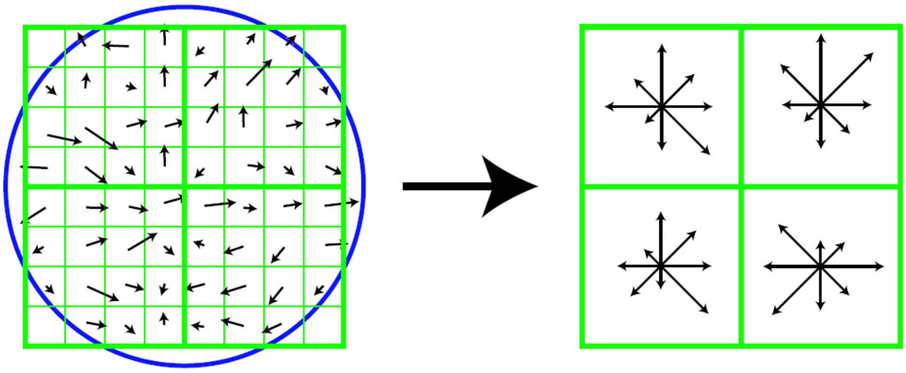


Figure 2.2: SIFT descriptor (right) constructed from gradient information (left) [Low04]. Arrows indicate gradient direction and their length, the magnitude. The circle represents the Gaussian-weighted window. This is an example for a region of 8x8 pixels (left) leading to a 2x2 8-bin histogram descriptor (right).

multi-resolution histograms [HGN01]. These are especially used as global features for image indexing and retrieval. More recently developed descriptors are the intensity-domain spin images [LSP05] and spatiograms [BR05]. The region descriptor proposed in this work also combines spatial and distribution information as described in Section 3.2.

Besides the histogram-based descriptor, another distribution shows interesting properties. Rubner *et al.*[RTG00] propose a descriptor called *signature*. The idea behind a signature is to use variable sized bins. A fixed size, as in histograms, increases the difficulty to finely describe a distribution. The common solution is to augment the number of bins. However, the counterpart of this gain in expressiveness is a decrease in efficiency. Hence, using variable-sized bins permits a better representation of the distribution space without such efficiency loss. A signature $\{s_{L(\mathbf{P})},(n)\} = \{[m(n);w(n)]\}$, represents a set of clusters. Each cluster is represented by its mean $m(n)$, and by the number $w(n)$ of pixels in $L(\mathbf{P})$ that belong to that cluster. The size of the signature is also variable and depends on the complexity of the described image. In this way, more accurate and larger descriptors are obtained for complex images. As the authors claim, the size of the clusters in the feature space $\mathbb{S}$ should be limited. Indeed, it should not exceed the extent of what is perceived as the same, or very similar, feature. A similarity measure called the Earth Mover's Distance is also proposed in this work. It is based upon the transportation problem and shows illumination invariance when the information described is based on intensity values (grey-level or colour). A histogram can be seen as an especial case of signature. In a histogram, the values that n can take are fixed a priori. Moreover, the values of $m(n)$ are usually equidistant in the space $\mathbb{S}$ and centered at the range of values of each bin. The values of the histogram $h(n)$ are equivalent to $w(n)$.

2.3 Matching strategies

The strategy to locate and match regions inside an image varies depending on the application and often also on the complexity of the descriptor or the measure of similarity used. Moreover, different descriptors can be used for each strategy. Three grouped strategies can be identified in literature, namely, point correspondence, line-search and trust-region, and window-search matching. Point correspondence deals with feature points previously detected in two images. In some cases, there is no knowledge of specific points that may match the region that is sought. In such situation, a broader search, often confined to a region of the image, is performed. This is the case of the other two groups.

2.3.1 Point correspondence

The point correspondence problem consists in establishing the connection between points in two or more images, representing the same real point (e.g., [CH96, Low04]). For convenience, only locations with high repeatability are considered. This is the case of feature points or regions such as edges and corners. For a review of interest region detectors the reader is referred to [MTS+05]. Once the detection of possible candidates (usually a large amount of points) in each image is done, a pair-wise match has to be set. Therefore, the similarity is only computed between pairs of interest points. For a large amount of points, this process is usually computationally complex. Nevertheless, methods to efficiently obtain the correct matches exist.

Cox and Hingorani [CH96] reduce the number of points to match by using a probabilistic approach. Indeed, the region in the image where the correct match may lie is inferred from previous matches. This is a common data association problem in tracking environments. More details and similar examples are described in Section 5.3.2.

Lowe [Low04] uses a nearest neighbour search in the database of feature point descriptors. SIFT descriptors are compared using the Euclidean distance between vectors. In order to efficiently search in the database, the author uses a custom method called Best-Bin-First algorithm.

2.3.2 Line-search and trust-region

Line-search and trust-region matching strategies consist in minimising a function that describes the dissimilarity between descriptors. This minimisation process is done in several iterations starting at a known position in the image. At each iteration the similarity between the patch and the neighbourhood of a given point is computed. Given this result it is possible to find another location where the expected similarity is higher. The process ends when the similarity is enough for the purposes of the application.

Line-search matching is the process of maximising the similarity $d(\cdot,\cdot)$ by searching at points in an image along a defined direction, e.g. steepest descent on the gradient. Comaniciu *et al.*[CRM03] propose a minimisation process based on the Bhattacharyya coefficient ρ [Kai67]

$$\begin{aligned} d(f(\mathbf{P}), f(\mathbf{R}_{x,y})) &= \sqrt{1-\rho(f(\mathbf{P}), f(\mathbf{R}_{x,y}))} & (2.10)\\ \rho(f(\mathbf{P}), f(\mathbf{R}_{x,y})) &= \sum_{n=0}^{N-1} \sqrt{f(\mathbf{P},n)\cdot f(\mathbf{R}_{x,y},n)}, & (2.11) \end{aligned}$$

where $f(\mathbf{P},n)$ is equal to $h_{L(\mathbf{P})}(n)$, defined previously in Equation (2.7), normalised so that $\sum h(n) = 1$. The process starts at a known position $\mathbf{x}_0 = (x,y)_0$ and evaluates this coefficient. Then a new position $\mathbf{x}_{k+1}$ is found using the derivative of the kernel defined in Equation (2.6)

$$\begin{aligned} w_i &= \sum_{n=0}^{N-1} \sqrt{\frac{h_{L(\mathbf{P})}(n)}{h_{L(\mathbf{R}_{\mathbf{x}_0})}(n)}} \cdot \delta(\, b\,(L(\mathbf{P}(\hat{x}_i)))\,) - n\,] \\ \mathbf{x}_{k+1} &= \frac{\sum_{i\in\mathbf{L}} \mathbf{x}_i \cdot w_i \cdot g\left(\left\|\frac{\mathbf{x}_k-\mathbf{x}_i}{r}\right\|^2\right)}{\sum_{i\in\mathbf{L}} w_i \cdot g\left(\left\|\frac{\mathbf{x}_k-\mathbf{x}_i}{r}\right\|^2\right)} & (2.12) \end{aligned}$$

where $g = -k'(x)$. The process continues with iterative evaluations while $\rho(f(\mathbf{P}), f(\mathbf{R}_{\mathbf{x}_{k+1}})) < \rho(f(\mathbf{P}), f(\mathbf{R}_{\mathbf{x}_k}))$. And finally stops when $\|\mathbf{x}_k - \mathbf{x}_{k+1}\| < \varepsilon$. Other examples where this strategy is applied can be found in [GM04, BR05].

Liu and Chen [LC04] propose a more generic iterative process. Instead of searching along a defined direction, the point that maximises the similarity is searched exhaustively in a region, called *trust region*. This region is centered at the position that maximised the similarity in the previous iteration. Moreover, the size of this region is adapted to the similarity achieved. The advantage of this technique as compared to the previous one is that the probability of falling in local minima is lower. In some cases, an exhaustive search is performed regardless of the previous position, this is known as a window-search strategy and is explained next.

2.3.3 Window-search

Window-search matching consists in comparing the descriptor of the patch and the descriptors computed at each point in an image. The result of this procedure is a *similarity map*. The computational power needed to built a similarity map is proportional to the size of the map. Due to this fact, it is often applied only when the descriptor at each point is computed rapidly or the size of the map is relatively small [ZC93, ST94, Lew95, FT97, Bir98, FKCK05, ARS06]. The Rotation-Discriminative Template Matching (RDTM) proposed in Chapter 3 also produces a similarity map and hence is close to this sort of matching strategy.

Efficient methods to compute this similarity map have been developed in literature. Two relevant methods are the integral image [VJ01] and the integral histogram [Por05].

The integral image $\check{\mathbf{I}}$ is a running sum of an image $\mathbf{I}$ computed as follows

$$\begin{aligned} \mathbf{S}(x,y) &= \mathbf{S}(x,y-1) + \mathbf{I}(x,y) \\ \check{\mathbf{I}}(x,y) &= \check{\mathbf{I}}(x-1,y) + \mathbf{S}(x,y), & (2.13) \end{aligned}$$

where $\mathbf{S}(x,y)$ is a cumulative row sum, $\mathbf{S}(x,-1)=0$ and $\check{\mathbf{I}}(-1,y)=0$. This computation needs only one pass over the image. The interesting property of the integral image is that any sum of the values of a region can be computed in only four memory accesses

$$\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}\mathbf{R}_{x,y}(i,j)=\check{\mathbf{I}}(x_r,y_b)+\check{\mathbf{I}}(x_l,y_t)-\left(\check{\mathbf{I}}(x_r,y_t)+\check{\mathbf{I}}(x_l,y_b)\right), \tag{2.14}$$

where, for an odd-sized region (and integer division), $x_r=x+W/2$, $x_l=x-W/2$, $y_b=y+H/2$, and $y_t=y-H/2$. This efficient computation of sums is used especially since the work of Viola and Jones [VJ01]. Nonetheless, this running sum of an image was previously applied for correlation purposes by Lewis [Lew95]. In this work, the integral image is used to compute the NCC rapidly. Indeed, it is possible to speed up an exhaustive NCC computation over an image. This is achieved by obtaining the integral image and integral of squared image and using it to calculate $\overline{\mathbf{R}}_{x,y}$ and the sum in the denominator (see Equation (2.3), p.11). Fast strategies of cross-correlation are often performed with the Fast Fourier Transform (e.g.,[FKCK05]). The FFT is faster than exhaustively computing the NCC. However, the normalisation preferred in template matching does not have a correspondingly simple and efficient frequency domain expression.

The integral histogram $\check{\mathbf{H}}$, on the other hand, could be seen as a running sum of bins. More precisely, the integral histogram is built as a set of integral images, one for each bin. The process to obtain an integral histogram is composed of two steps. Firstly, a certain information $L(\mathbf{I})$ is quantised in N levels. Secondly, an integral image of each level is computed. The value of a single bin in a histogram is

$$h_{L(\mathbf{R}_{x,y})}(n)=\check{\mathbf{H}}(x_r,y_b,n)+\check{\mathbf{H}}(x_l,y_t,n)-\left(\check{\mathbf{H}}(x_r,y_t,n)+\check{\mathbf{H}}(x_l,y_b,n)\right), \tag{2.15}$$

where $\check{\mathbf{H}}(x,y,n)$ is the integral image of the n-th level. The consequent advantage of computing the contribution of each bin in separate integral images is that a histogram can be computed with only $4N$ memory accesses.

It must be pointed out that computing the integral image or the integral histogram is also an advantage when different scales in the search process are used. Indeed, changing the scale or the size of the region for which the sum or the histogram is needed does not change the necessary computational time.

Let us now take a look at specific examples of fast rotation invariant matching with an exhaustive search, namely [FU01, UK04].

Fredriksson *et al.*[FU01] use an orientation invariant descriptor (colour histogram) to locate points with high probability of match. Although this method is faster than the commonly used cross correlation by FFT, histograms are not efficiently computed in this work.

Ullah *et al.*[UK04] use the gradient orientation $\nabla_\theta(x,y)$ (see Equation (2.8), p.13) of an image patch. The result is called an orientation code **OC**.

$$\mathbf{OC}(x,y)=\begin{cases}\frac{\nabla_\theta(x,y)}{\Delta} & |\nabla_m(x,y)|>th\\ N & \text{otherwise},\end{cases} \tag{2.16}$$

where $\Delta = 2\pi/N$ is the quantisation step. N is the number of bins used to obtain a histogram of **OC**, called orientation code histogram **OH**. The **OC** and the **OH** form the descriptor $f(\cdot)$. This descriptor is matched to another image **I** in a two step strategy. Firstly, orientation code histograms are computed at each point of an image. Assume for the moment that comparing orientation histograms allows to estimate the rotation that a patch might have undergone at each point of an image. A complete development of this idea is given in the next chapter. Now, the computation of this rotation is given by the similarity measure $D_1 = 1 - \max_k S^k$. The second term of this expression is the normalised area under the curve obtained by the intersection between the histogram of the patch **P** and that of the region $\mathbf{R}_{x,y}$ shifted left by k bins (symbolised by the superscript k)

$$S^k = \frac{1}{W \cdot H}\left(\sum_{n=0}^{N-1} \min\{h_{L(\mathbf{P})}(n), h^k_{L(\mathbf{R}_{x,y})}(n)\} + \min\{h_{L(\mathbf{P})}(N), h^k_{L(\mathbf{R}_{x,y})}(N)\}\right), \tag{2.17}$$

for $k = 0, \ldots, N-1$. Computing D_1 for each point in the image gives a first similarity map. Secondly, orientation codes are matched at the right orientation and only to the best histogram matches. This process is referred to as OCM. This independent work has similarities with the technique proposed in this thesis. However, it differentiates from our method in two main contributions. First and most important, the **OH** is built only upon the extracted patch at a single orientation achieving less invariance to rotations than our descriptor. Second, the processing time needed to produce a match is much higher. A comparison between this and other techniques discussed above is given in the next chapter.

2.4 Similarity measures

For the sake of completeness, a brief note on similarity measures is also given hereafter. As said before, the matching performance is also related to the similarity $d(\cdot,\cdot)$ used to compare descriptors (see Equation (2.1), p.10). For instance, cross correlation [ZC93, Lew95, CH96] and sum of squared distances (SSD) [ST94, HB96, JD02a] are commonly used for template matching. In the case of histogram matching, more alternatives exist. The Bhattacharyya distance [Kai67] and the Earth Mover's Distance (EMD) [RTG00] have attracted the attention of researchers because of their discriminative and illumination invariance properties, respectively. Examples of the former applied to tracking frameworks can be found in [PHVG02, CRM03], whereas the latter has been applied both to indexing [RTG00] and tracking [ARS06]. For more examples and details on distribution similarity metrics, the reader is referred to [RPTB01].

2.5 Summary

This chapter has given an overview of the necessary tools for region or patch recognition. On the one hand, there is the descriptor of the region to match. On the other, the strategy used for matching such descriptors. Among the vast number of possibilities, only those descriptors or strategies related to the method proposed in the next chapter have been discussed. This discussion has situated the background in relation to examples from the state of the art.

State-of-the-art research on this topic has not yet solved completely the problems of viewpoint and illumination invariance. Moreover, the existing applications of region recognition show that there is a growing demand of methods achieving invariance with a reduced computational cost. This is the case, for instance, of visual tracking. The next chapter presents a method devoted to address recognition problems with reduced processing time.

Rotation-discriminative template matching

3

In the previous chapter, region recognition methods have been divided in two categories, namely, trained and non-trained methods. In this chapter, we concentrate on the second category and target environments with demanding time constraints such as the camera tracking framework presented in the second part of this thesis. Therefore, the generic problems of region recognition, namely, illumination and viewpoint changes have to be solved efficiently in order to keep processing time low.

This chapter describes a recognition method that addresses these problems with a particular focus on 2D rotation changes. This description together with its evaluation is complemented with a potential application on visual tracking.

3.1 Introduction

The particular problem that is tackled here is template matching of small patches that have undergone two dimensional rotations. More precisely, an image that contains the patch has undergone a rotation with respect to the normal to the image plane. The patch in the image is deformed with a rotation very similar to that applied to the image. The scale of the patch is assumed to be preserved. A straight approach to this problem would be to generate a number of rotated versions of a patch and to correlate each one of them at each point of the tested image. This window matching process would however have a high computational cost.

Instead, we propose to estimate firstly which rotated version has the highest probability of being the adequate to maximise the level of correlation. This step is a rotation discrimination. Then, it is possible to perform a correlation only with the adequate rotated version rather than with all the rotated versions. This step is a template matching. Consequently, we call our method

Rotation-Discriminative Template Matching (RDTM). This technique is composed of a rotation-discriminative patch descriptor and an efficient hierarchical search strategy divided in three steps. Firstly, similar gradient magnitude is exhaustively searched within the image. The most similar points are sorted. Secondly, the orientation gradient histogram is matched at those points, providing a measure of similarity together with an estimate of the rotation that the patch has undergone. Again, only the most similar points are kept. Finally, template matching is performed at those points by computing the Normalised Cross Correlation (NCC) between the intensity neighbourhood of the point in the image and the patch rotated according to the orientation estimated in the previous step. In order to perform most scan operations rapidly, we take advantage of the integral image and the integral histogram, both described previously in Section 2.3.3, p.16.

The remainder of the chapter is structured as follows. Firstly, the tailored region descriptor and the method to efficiently perform template matching are presented in Sections 3.2 and 3.3, respectively. Secondly, the computational complexity of the algorithm is analysed in Section 3.4. Finally, an experimental assessment is given in Section 3.5.

3.2 Rotation-discriminative region descriptor

Section 2.1 gives an idea of the vast number of different region descriptors available in literature. As stated before, two main problems have to be addressed in recognition, namely, illumination and viewpoint changes. We tackle these two issues simultaneously by using the gradient information. On the one hand, the gradient has little sensitivity to illumination changes. Indeed, if the reflectance of the surface is considered Lambertian, a uniform change in illumination has no effect on the gradient. On the other hand, we propose a descriptor that addresses the viewpoint problem concentrating on rotation robustness and, at the same time, provides orientation information of the region it describes. This particular information is used to identify the rotation that a patch has undergone when detected in another image. This is the origin of the *discriminative* characteristic of the descriptor. Knowing the approximate rotation that a patch has undergone is exploited in the camera tracking framework presented in the second part of this thesis.

Let us first analyse the behaviour of the gradient. From a theoretical point of view, the gradient has a continuous response to a continuous and derivable function. Suppose that a gradient orientation histogram of N bins is computed from the intensity information of an image patch $\mathbf{P}$. In this case, a rotation of the patch by δ degrees changes the values in the histogram. In particular, when $\delta = n \cdot 360/N$ where $n \in \mathbb{Z}$, the histogram would be exactly equal to a perfect shift, and the shift in bins would be equal to n. However, this ideal case is not fulfilled in reality. Indeed, images are a pixelised, quantised and interpolated representation of light. Moreover, the gradient of a digital image cannot be computed from a continuous function. Consequently, even rotations od δ degrees have an impact on the gradient histogram beyond the effect of shifting the theoretic histogram.

Following the observation that histograms change with different orientations, we propose to generate rotated versions of a patch and, from these versions, create a single histogram that can deal with rotations. As mentioned before, orientation histograms repeat approximately their shape every $\Delta = 360/N$ degrees. This can be exploited by aligning the histograms of versions rotated exactly by $k\Delta$ with $k \in \mathbb{Z}$.

The histogram descriptor is obtained as explained next. Firstly, N rotated versions of the patch $\mathbf{P}$ to be matched are pre-computed with an angle of rotation of $i\Delta$ degrees with $i = 0,..,N-1$, where N is the number of bins. These versions are cropped so as to eliminate additional pixels introduced by the rotation, leading to an array of rotated versions of the patch $\overrightarrow{\mathbf{P}}$. The element $\overrightarrow{\mathbf{P}}(i)$ is a version rotated $i\Delta$ degrees. Secondly, the gradient of each of these versions is computed as in Equation (2.8), p.13. Then, each $\nabla_\theta(i)$ is quantised in N bins. In order to compact the statistical description of the patch and to reduce the effect of noise, the contribution of each point in $\nabla_\theta(x,y,i)$ to the corresponding bin is weighted by its magnitude $\nabla_m(x,y,i)$ (similar to the approach of Lowe [Low04], see Section 2.2.2). Using the generic formulation of a histogram given in Equation (2.5), p.11, one obtains the expression for the histogram of a single rotated version $\mathbf{h}_{\overrightarrow{\mathbf{P}}(i)}$

$$\begin{aligned} b &: [0,2\pi[\to \{0,\ldots,N-1\} \\ h_{\overrightarrow{\mathbf{P}}(i)}(n) &= \sum_{\overrightarrow{\mathbf{P}}(i)} \nabla_m(x,y,i) \cdot \delta\left(b\left(\nabla_\theta(x,y,i)\right) - n\right) \quad n = 0,\ldots,N-1 \qquad (3.1) \\ \mathbf{h}_{\overrightarrow{\mathbf{P}}(i)} &= [h(0),\ldots,h(N-1)]_{\overrightarrow{\mathbf{P}}(i)} \qquad (3.2) \end{aligned}$$

It is desirable that the weight of the peripheral pixels is lessened. However, applying a kernel (as presented by Comaniciu *et al.*[CRM03]) is not possible with the integral histogram approach. The effect of the kernel is approximated by giving double weight to the central part of the patch. Redefining Equation (3.1),

$$\begin{aligned} h_{\overrightarrow{\mathbf{P}}(i)}(n) &= \sum_{\overrightarrow{\mathbf{P}}(i)} \nabla_m(x,y,i) \cdot \delta\left(b\left(\nabla_\theta(x,y)\right) - n\right) \\ &+ \sum_{\overrightarrow{\mathbf{P}}'(i)} \nabla_m(x,y,i) \cdot \delta\left(b\left(\nabla_\theta(x,y)\right) - n\right) \quad n = 0,\ldots,N-1, \qquad (3.3) \end{aligned}$$

where $\mathbf{P}'$ is the central part of a patch $\mathbf{P}$. Finally, the global histogram of the patch is the mean obtained with the N histograms aligned according to their rotation.

$$\begin{aligned} \hat{\mathbf{h}}_{\overrightarrow{\mathbf{P}}(i)} &= [h(N-i),\ldots,h(N-1),h(0),\ldots,h(N-1-i)]_{\overrightarrow{\mathbf{P}}(i)} \qquad (3.4) \\ \tilde{\mathbf{h}}_{\mathbf{P}} &= \frac{1}{N}\sum_{i=0}^{N-1} \hat{\mathbf{h}}_{\overrightarrow{\mathbf{P}}(i)} \qquad (3.5) \end{aligned}$$

Figure 3.1 shows an example for 16 bins with the original patch and its rotated versions with the corresponding histogram aligned accordingly.

This average of rotated versions gives a robust descriptor when the rotation of the image is around $n\Delta$ degrees. It could be argued that for non-integer bin-wide angles higher variations will occur. However, experiments show that, with enough bins, this descriptor is reliable even around $n\Delta + \Delta/2$ degrees (see Section 3.5).

The final region descriptor is composed of the global histogram $\tilde{\mathbf{h}}_{\mathbf{P}}$, its variance $\sigma^2_{\mathbf{P}}$, its norm

$$||\tilde{\mathbf{h}}_{\mathbf{P}}|| = \sum_{n=0}^{N-1} \tilde{\mathbf{h}}_{\mathbf{P}}(n) = \frac{1}{N}\sum_{i=0}^{N-1}\left(\sum_{\overrightarrow{\mathbf{P}}(i)} \nabla_m(x,y,i) + \sum_{\overrightarrow{\mathbf{P}}'(i)} \nabla_m(x,y,i)\right), \qquad (3.6)$$

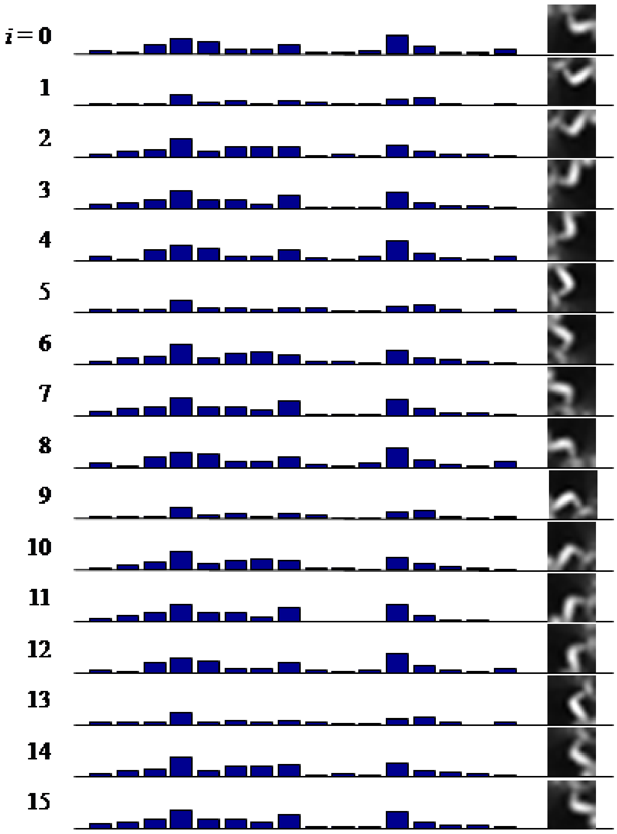


Figure 3.1: Example of histogram alignment with $N = 16$ bins. Central column: histograms aligned according to their rotation; right column: corresponding original patch and rotated versions.

and the array of rotated versions of the template $\overrightarrow{\mathbf{P}}$

$$f(\mathbf{P}) = \left[\tilde{\mathbf{h}}_{\mathbf{P}}, \sigma^2_{\mathbf{P}}, \|\tilde{\mathbf{h}}_{\mathbf{P}}\|, \overrightarrow{\mathbf{P}}\right]. \tag{3.7}$$

All of these elements are used in the matching procedure of the RDTM explained below.

3.3 Efficient rotation-discriminative matching

This section describes the efficient matching of the patch $\mathbf{P}$ to any point of an image $\mathbf{I}$, using the descriptor previously depicted. This process is divided in three hierarchical selection steps, each of them sorting out a set of most probable candidates. Firstly, an exhaustive gradient magnitude comparison is performed. Secondly, the candidates with highest magnitude similarity are kept for orientation gradient histogram matching. The similarity measure employed for histogram matching also provides an estimate of the rotation between the patch and the image. Finally, the most similar histograms together with the rotation estimated at those positions are used in the template matching process. These steps are detailed in the remainder of this section.

3.3.1 Gradient magnitude matching

The norm of the histogram $||\tilde{\mathbf{h}}_{\mathbf{P}}||$ can be used as a simple feature to rapidly scan the image for similar candidates. From the construction of the histogram it can be found that,

$$||\tilde{\mathbf{h}}_{\mathbf{P}}|| \simeq \sum_{\overrightarrow{\mathbf{P}}(i)} \nabla_m(i) + \sum_{\overrightarrow{\mathbf{P}}'(i)} \nabla_m(i), \tag{3.8}$$

for any $i \in [0, N-1]$. Following this observation, we propose to compare the global histogram norm of the patch with the gradient magnitude ∇_m of $\mathbf{I}$ in a window-search strategy. This can be efficiently performed with the integral image (see Section 2.3.3, p.16) of ∇_m. Given a neighbourhood $\mathbf{R}_{x,y}$ of a point (x,y) in the image $\mathbf{I}$, the measure used to compare the norm is

$$d_m(x,y) = \exp\left(-\alpha \cdot \left(1 - \frac{\sum_{\mathbf{R}_{x,y}} \nabla_m + \sum_{\mathbf{R}'_{x,y}} \nabla_m}{||\tilde{\mathbf{h}}_{\mathbf{P}}||}\right)^2\right), \tag{3.9}$$

where α is a factor that weights this similarity according to the variance of the histogram. This factor is fixed, upon experimentation, to $\alpha = N/(1000 \cdot ||\sigma^2_{\mathbf{P}}||)$. A set of candidates with similar histogram norm is defined

$$S_m = \left\{ (x,y) \in \mathbf{I} \;\middle|\; d_m(x,y) > 0.9 \right\}. \tag{3.10}$$

These candidates are kept for further matching.

In the worst case where similar magnitude is found all over the image, the number of candidates remains the same after this step. However, based on experiments, this simple selection criteria permits a reduction of the number of candidates by an average factor of 20.

3.3.2 Histogram matching

The gradient orientation histogram matching is applied to the set S_m. Histograms are efficiently computed with the integral histogram (see Section 2.3.3, p.16). Similarly to what is done for the descriptor of the patch, the gradient orientation histogram of a region $\mathbf{R}_{x,y}$ in the image is obtained from the contribution of the quantised $\nabla_\theta(x,y)$ weighted with $\nabla_m(x,y)$.

The similarity between the histogram of the patch $\tilde{\mathbf{h}}_{\mathbf{P}}$ and that of each candidate is computed with a custom measure to compare orientation histograms. Actually, this measure can be used with any sort of circular vector. We call it *Circular Normalised Euclidean Distance* (CNED). Not only the CNED measures the distance d between two vectors, but it also determines the circular shift $\hat{s}$ that corresponds to the minimal distance. Mathematically expressed

$$\begin{aligned}
\mathbf{CNED}(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b}) &= [\hat{s}(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b}) \;\; d_{\hat{s}}(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b})]^T && (3.11)\\
\hat{s}(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b}) &= \arg\min_s d_s(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b}) && (3.12)\\
d_s(\mathbf{a}, \sigma^2_{\mathbf{a}}, \mathbf{b}) &= \sqrt{\sum_{i=0}^{N-1} \frac{(\mathbf{a}(i) - \mathbf{b}((i+s) \bmod N))^2}{\sigma^2_{\mathbf{a}}(i)}}, && (3.13)
\end{aligned}$$

where $\mathbf{a}$ and $\mathbf{b}$ are vectors of length N, s is the shift that takes a discrete value between 0 and $N-1$, mod is the modulus function, and $\sigma^2_{\mathbf{a}}$ is the variance associated to vector $\mathbf{a}$. The result of this matching is hence a similarity score $d_{\hat{s}}$ and an estimate of the orientation of the patch $\hat{s}\cdot\Delta$ for each candidate.

Using this metric, the histogram matching step leads to a gradient histogram-based similarity map (GHSM)

$$\mathbf{GHSM}_{\mathbf{P},\mathbf{I}}(x,y) = \begin{cases} d_{\hat{s}}(\tilde{\mathbf{h}}_{\mathbf{P}}, \sigma^2_{\mathbf{P}}, \mathbf{h}_{\mathbf{R}_{x,y}}) & (x,y) \in S_m \\ 0 & \text{otherwise}, \end{cases} \tag{3.14}$$

where $\mathbf{h}_{\mathbf{R}_{x,y}}$ is the histogram computed for a region centered at (x,y) (see Equations (3.2)-(3.3)). In addition, this computation obtains the map of estimates of the orientation between the patch and the image

$$\Theta_{\mathbf{P},\mathbf{I}}(x,y) = \begin{cases} \hat{s}(\tilde{\mathbf{h}}_{\mathbf{P}}, \sigma^2_{\mathbf{P}}, \mathbf{h}_{\mathbf{R}_{x,y}}) & (x,y) \in S_m \\ \text{undefined} & \text{otherwise}. \end{cases} \tag{3.15}$$

Another subset of candidates with similar histogram S_h is defined. This set contains a fixed amount of candidates with the most similar histograms. This amount is a parameter of the method.

These candidates are kept for further matching.

3.3.3 Template matching

The magnitude and the orientation histogram discard many unrelated points but the result is still not selective enough (as seen below in Section 3.5). Spatial intensity information (template) is used as a further selection criterion.

Template matching is done using a NCC (see Equation (2.3), p.11). Templates $\mathbf{R}$ centered at those points with high histogram similarity (S_h) are compared to the corresponding template of the patch $\overrightarrow{\mathbf{P}}_{\hat{s}}$.

In order to perform this computation fast, the integral image and integral squared image of $\mathbf{I}$ are computed. In this way, the NCC can be computed in only a few memory accesses (see Section 2.3.3, p.16). Additionally, more efficiency is gained by computing $\overline{\overrightarrow{\mathbf{P}}}_{\hat{s}}$ and $\sum\sum\left(\overrightarrow{\mathbf{P}}_{\hat{s}} - \overline{\overrightarrow{\mathbf{P}}}_{\hat{s}}\right)$ prior to template matching.

A correlation map Ψ can be built using the result of matching the templates of the candidates in S_h. The map takes value 0 everywhere except at the location of these candidates, where the value $\in [0,1]$ is the NCC computed as described before

$$\Psi_{\mathbf{P},\mathbf{I}}(x,y) = \begin{cases} \text{NCC}\left(\overrightarrow{\mathbf{P}}_{\Theta(x,y)}, \mathbf{R}_{x,y}\right) & (x,y) \in S_h \\ 0 & \text{otherwise}. \end{cases} \tag{3.16}$$

3.4 Computational complexity

The processing time needed to match a region with an image determines the applicability of the technique to real-time environments. This section discusses the computational complexity of the proposed algorithm and possible ways to reduce it.

The description of a patch takes three steps: the creation of the rotated versions, creation of each individual histogram (one per version) and, finally, alignment and descriptor computation. The first step can be performed very rapidly using the processing power of a graphic card and the last two are proportional to the size of patches, which is often very small. Hence, one advantage of this type of descriptors is that they can be computed on-the-fly. Therefore, in a tracking application it would be possible to add new regions to track at run-time.

The RDTM is logically where most of the processing time elapses. Each step, separated in consecutive order, gives rise to the following cost.

- The computation of the gradient information (magnitude and orientation) and the integral histogram is done only once and is proportional to the size of the image ($W \cdot H$ pixels).
- The exhaustive magnitude comparison is performed at each point in the image and hence is also proportional to $W \cdot H$.
- The N bins-histogram is calculated in only $4N$ memory accesses and additions (independently of the size of the patch). This is the great advantage of the integral histogram over conventional methods (see [Por05] for a complete complexity derivation). This process is performed on a limited number of candidates $|S_m|$ with similar histogram norms.
- Each histogram $\mathbf{h}_{\mathbf{R}_{x,y}}$ is compared to the histogram of the patch $\tilde{\mathbf{h}}_{\mathbf{P}}$ using the CNED in $O(N^2)$ operations.
- The template matching is proportional to the size of the patch and the number of candidates $|S_h|$ kept after the histogram matching. The integral image and integral squared image is computed also once and this process is proportional to the size of the image.

To summarise, the matching step takes roughly $W \cdot H + (4N + N^2) \cdot |S_m| + (W_{\mathbf{R}} \cdot H_{\mathbf{R}}) \cdot |S_h|$ operations, where $W_{\mathbf{R}} \cdot H_{\mathbf{R}}$ is the size of the patch.

Although the number of candidates $|S_h|$ plays a role in the processing speed, our experimentation has shown that three parameters determine the rapidity of the algorithm, namely, N, W and H. As shown in Section 3.5, the number of bins determines the results of the system, whereas these are independent of W and H. In addition, $|S_h|$ is in accordance to these two parameters. Consequently, W and H should be decreased in case fast computation is needed.

In some applications, there is a rough knowledge of the area, within an image frame, where the patch may lie. In these cases, W and H can be drastically reduced. This is the case for tracking applications where the intra-frame motion can be predicted and the search region is known from the previous location. The framework presented in Chapter 6 is such an example. For 2D tracking environments, a more detailed description with additional advantages introduced by the RDTM is given in Section 3.6.

3.5 Experiments

This section describes the evaluation of the proposed method. Firstly, the test set used to assess the performance is presented. Secondly, the correlation accuracy of the RDTM is tested on its own. Thirdly, the orientation accuracy is discussed. Finally, a comparison with other similar techniques, both in terms of performance and computational cost, concludes this section. The methodology of each experiment is presented in its respective subsection.

3.5.1 Test set

The set of images used for testing is shown in Figure 3.2. The first two images are custom whereas the last six images are taken from the Visual Geometry Group database [Rob07]. It can be seen that this set has textured regions and, in many cases, similarity between these regions.

There are two key parameters in the method: the number of bins in the histogram N and the number of candidates chosen from the histogram matching step $|S_h|$. Experiments are run with 10, 16 and 20 bins to give an approximate idea of lower and upper performance bounds. The number of bins determines the value of Δ (see Section 3.2) and, hence, the performance of the method. More concretely, the RDTM is expected to work better for rotations around $k\Delta$ than around $k\Delta+\Delta/2$ (with $k = 0,..,N-1$). In order to experiment with these best and worst scenarios, the images are rotated according to each histogram length. Table 3.1 gives the tested rotation angles for each histogram length. Note that the rotation angles are not exactly equal to $k\Delta$ nor to $k\Delta+\Delta/2$. The purpose of this choice is to use common rotation angles for different histogram lengths and hence be able to observe the different performance achieved for the same rotated image. The method is also run on the original images (no transformation). The number of candidates in the set S_h extracted from the GHSM ranges from 1 to 500.

N bins	$\sim k\Delta$	$\sim k\Delta+\Delta/2$
10 ($\Delta = 36^o$)	20	70
16 ($\Delta = 22.5^o$)	10	70
20 ($\Delta = 18^o$)	10	70

Table 3.1: Rotation angles of original images according to tested histogram length.

For each one of the original images, a set of patches is extracted. The method used for patch extraction is tailored for the matching method proposed. Kadir and Brady [KB01] indicate the convenience of using the same feature(s) to detect regions and describe them. In their work, they use the entropy of the histogram as a feature of saliency. Moreover, the size of the detected region is chosen at a peak of entropy. Here a similar strategy is adopted.

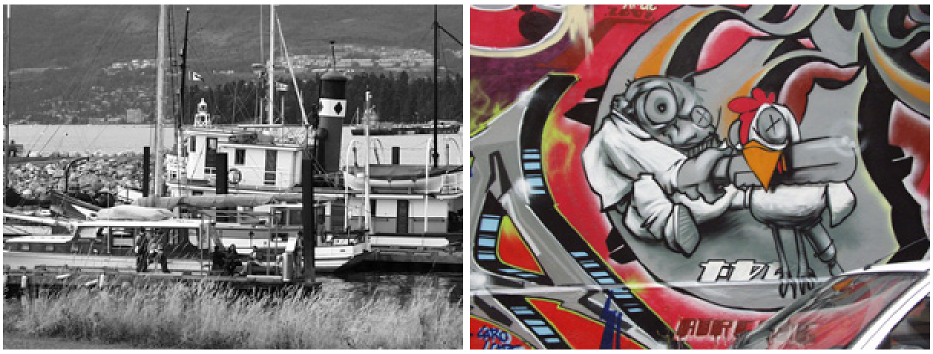
In the descriptor presented in Section 3.2, the most relevant feature is the gradient orientation histogram, as it determines in great measure the performance of the method. Actually, it is the richness of this feature that enables discrimination and orientation estimation. Consequently, regions with a rich gradient orientation histogram must be found in these images. It is desirable that the shape of a histogram has peaks and valleys and even more important, that those peaks and valleys have an heterogeneous shape. In this way, the CNED achieves the best performance. The "peakyness" η is measured directly with the values of the histogram computed at a given region $\mathbf{h_R}$

$$\eta(\mathbf{h_R}) = \frac{1}{N}\sum_i \left|\mathbf{h_R}(i) - \mathbf{h_R}(i-1)\right|. \tag{3.17}$$

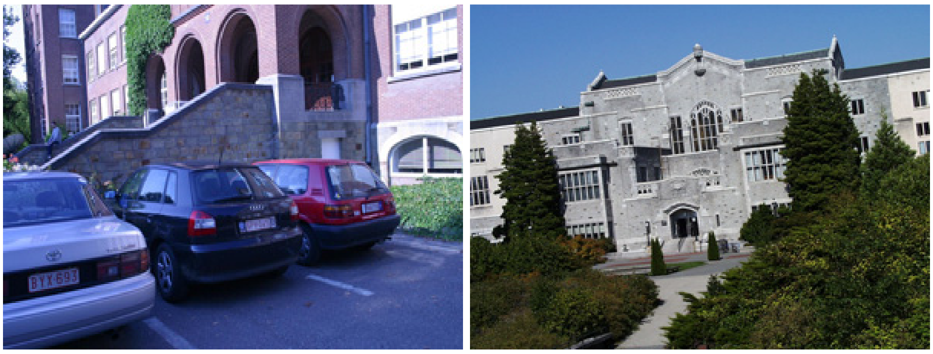
The "heterogeneousity" is indirectly enforced by limiting the variance of the histogram. This is

(a) Sunflowers, 300x225 pixels (b) Cathedral, 300x200 pixels

(c) Bark, 320x214 pixels (d) Bikes, 300x210 pixels



(e) Boat, 300x240 pixels (f) Graf, 300x240 pixels



(g) Leuven, 300x200 pixels (h) UBC, 300x240 pixels

Figure 3.2: Original images used for the experiments.

computed as follows:

$$\sigma^2(\mathbf{h_R}) = \frac{1}{N-1}\sum_i \left(\mathbf{h_R}(i) - \overline{\mathbf{h}}_\mathbf{R}\right)^2 , \tag{3.18}$$

where $\overline{\mathbf{h}}_\mathbf{R}$ is the mean of the histogram values. The procedure followed to obtain the patches is performed as described next.

1. Harris corner points [HS88] are first selected. This detector finds points with a high probability of multi-peaked orientation histograms and, essentially, with a high gradient magnitude.
2. For each point, regions are extracted at different sizes. More precisely, from a 10×10 pixels to a 20×20 pixels. This range of sizes is selected because it gives the best results.
3. The gradient orientation histogram is computed for each region. With the histogram, η is obtained.
4. The size that maximises η is kept and its patch extracted.
5. If the variance of the histogram is below $0.025/N$ (chosen upon experimentation) the candidate is ignored.

Once the patch is extracted, the descriptors can be built. After applying this extraction to the test set, 161 regions are obtained for histograms with 10 bins, 152 for 16 bins and 153 for 20 bins.

In the following evaluations, each patch is sought independently in the transformed images. Results are averaged for all the patches extracted.

3.5.2 Evaluation of the correlation accuracy

The RDTM gives a level of correlation at each point in the image. It would be possible to simply show the correlation map Ψ and let the reader evaluate its accuracy. However, showing all the maps for each patch and test image is not practical. Hence, in order to determine the performance of the proposed map, a numerical quantity that summarises all the responses is needed. This value should clearly show when a good localisation of a patch inside an image is provided. We consider that a good localisation is fulfilled when a high correlation is achieved at the ground truth position of the patch in the test image. The level of correlation and the number of good localisations among all the patches determine the capacity of the algorithm at different conditions (different test images). The accuracy of the algorithm is not only characterised by its capability of locating a patch at the right place. In addition, this metric should measure the discrimination between the right position and the remaining points. The level of correlation with points unrelated to the good location gives an idea of that distinction.

These ideas are summarised mathematically as follows. As seen before, the ground truth location of the patch inside a test image is necessary. In our evaluation, the transformation applied to the original images is known so this ground truth is available. Let us assume there is an operator $T(\mathbf{I})$ that performs a similarity transformation (rotation, translation, scaling) of an image $\mathbf{I}$. Now, suppose that a patch $\mathbf{P}$ is extracted from $\mathbf{I}$ and the centre of the patch is located at $(x_\mathbf{P}, y_\mathbf{P}) \in \mathbf{I}$.

Given an image $\hat{\mathbf{I}} = T(\mathbf{I})$, and $(\hat{x}_{\mathbf{P}}, \hat{y}_{\mathbf{P}})$ being the correspondence of $(x_{\mathbf{P}}, y_{\mathbf{P}})$ into $\hat{\mathbf{I}}$, two values are computed: the maximum correlation near the ground truth

$$\psi_{\in \mathbf{G}} = \max \Psi_{\mathbf{P},\hat{\mathbf{I}}}(x,y) \quad \forall (x,y) \in \mathbf{G}, \tag{3.19}$$

and the maximum correlation outside the neighbourhood of the ground truth

$$\psi_{\notin \mathbf{G}} = \max \Psi_{\mathbf{P},\hat{\mathbf{I}}}(x,y) \quad \forall (x,y) \notin \mathbf{G}, \tag{3.20}$$

where $\mathbf{G}$ is the region $\{\hat{\mathbf{I}}(x,y) | x \in [\hat{x}_{\mathbf{P}} - 1, \hat{x}_{\mathbf{P}} + 1]$ and $y \in [\hat{y}_{\mathbf{P}} - 1, \hat{y}_{\mathbf{P}} + 1]\}$. After image transformation and interpolation, it is possible that the original centre of the patch lies between two pixels. A 1 pixel neighbourhood is set to account for this sub-pixel location. It must be pointed out that these two values do not cover all the information that could be visualised in the map. For instance, the location of the point in the map that gives $\psi_{\notin \mathbf{G}}$ is discarded. As a matter of fact, this point could be just 2 pixels away from $(\hat{x}_{\mathbf{P}}, \hat{y}_{\mathbf{P}})$ or at a distant location. In most applications, the first case would not cause a problem whereas the second case would be much more relevant. However, by giving only a 1 pixel neighbourhood the accuracy of our method is clearly tested.

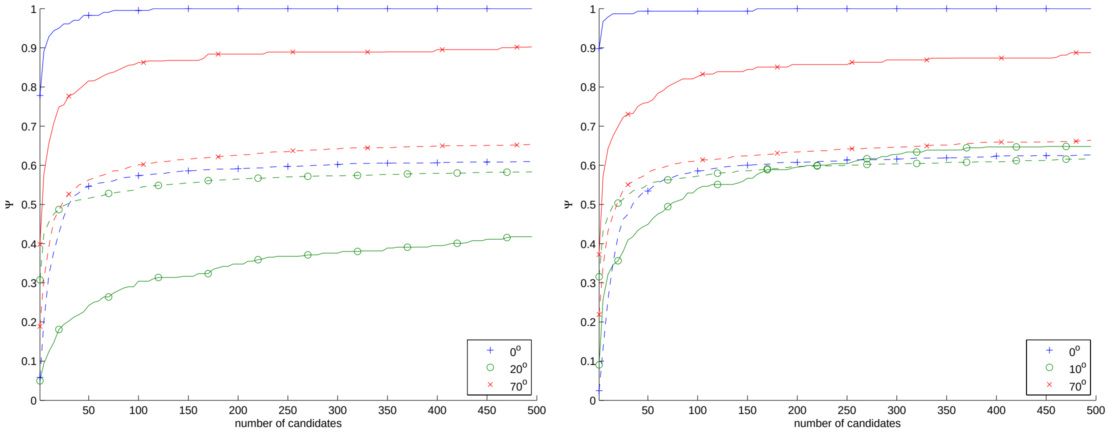
Figure 3.3 shows $\psi_{\in \mathbf{G}}$ (solid line) and $\psi_{\notin \mathbf{G}}$ (dashed line) averaged for all the patches in the test set. As it can be seen, the number of candidates kept from the histogram matching step $|S_h|$ has a direct influence on the performance. On the one hand, a small number of candidates (less than 100) is not reliable enough as in most cases $\psi_{\in \mathbf{G}}$ is small. On the other hand, for a number of candidates greater than 150 in general, several observations arise.

- The performance is improved as the number of bins in the histogram grows. This is especially visible for rotations around $k\Delta + \Delta/2$. In this particular test set, these rotations are at 20^o, 10^o and again 10^o, for 10, 16 and 20 bins, respectively (lines with circles in Figure 3.3).
- The performances achieved for rotations around $k\Delta$ (lines with '+' and '$\times$') are globally better than those achieved for $k\Delta + \Delta/2$ (lines with circles), as expected.
- The correlation inside G is larger than the correlation outside ($\psi_{\in \mathbf{G}} > \psi_{\notin \mathbf{G}}$), achieving the desired discrimination.
- A high correlation is achieved near the ground truth using 20-bins histograms ($\psi_{\in \mathbf{G}} \geq 0.7$ in the worst case, 10^o).

It must be pointed out that the number of candidates $|S_h|$ is related to the size of the images. In this case, 150 candidates is around 0.25% of the mean area of each image.

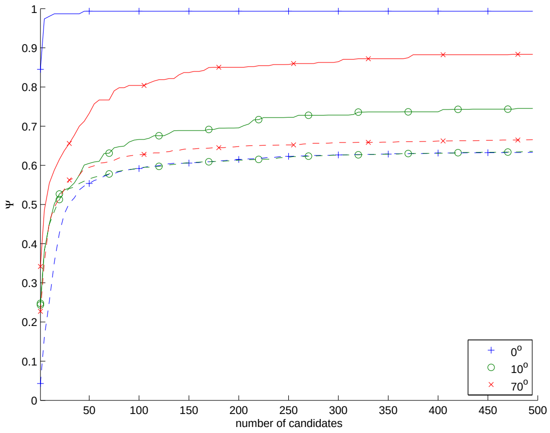
3.5.3 Evaluation of the orientation accuracy

One of the enhancements brought by the proposed method is the estimation of the relative orientation of a patch when detected in a test image. Section 3.6 described a potential application of this particular asset. As will be seen in the second part of this thesis, estimating the orientation while recognising provides an interesting added value to our method. Although directly related to Ψ, the



(a) 10 bins

(b) 16 bins



(c) 20 bins

Figure 3.3: Mean $\psi_{\in \mathbf{G}}$ (solid line) and $\psi_{\notin \mathbf{G}}$ (dashed line). Histograms computed with 10, 16 and 20 bins.

accuracy of this estimation is analysed for a more complete assessment.

The estimated orientation is given by $\Theta(x,y)$ (see Eq. 3.15) multiplied by the factor Δ to obtain the degrees of rotation. In the matching process, $\Theta(x,y)$ is used to compute the correlation $\Psi(x,y)$ (see Eq. 3.16). The validity of $\Theta(x,y)$ could then be tested with this resulting correlation. This process is not done in the RDTM to keep complexity low. However, this reasoning is used here for the evaluation of the orientation estimation. For each patch, the ground truth region $\mathbf{G}_{\hat{x}_\mathbf{P},\hat{y}_\mathbf{P}}$ in $\hat{I}$ where the patch $\mathbf{P}$ lies is selected in both $\Theta(x,y)$ and $\Psi(x,y)$. Then the orientation of each patch is computed as a weighted sum

$$\theta_{\mathbf{P},\hat{\mathbf{I}}} = \frac{\sum_{\mathbf{G}} \Psi(x,y) \cdot \Theta(x,y) \cdot \Delta}{\sum_{\mathbf{G}} \Psi(x,y)}. \tag{3.21}$$

Recalling from Section 3.3.2, the shift stored in Θ is circular by definition. This means that in the same way that a rotation is periodic every 360^o, the shift is periodic every N ($\Delta = 360/N$).

As the rotations of the test set are in the range $[-180^o, 180^o]$ it is more convenient to express the shift in the range $[-N/2, N/2]$. This fact is considered and the values of Θ inside the region G are changed accordingly. In order to compact the results, the orientation for all patches is grouped according to the orientation of the test image used. Results for all images in the test set are averaged together. Table 3.2 shows this mean estimation for a given number of bins and rotation angle of $\hat{\mathbf{I}}$ with respect to $\mathbf{I}$. The orientation accuracy is logically limited by the number of bins. Actually, the estimation is always around a multiple of Δ. These results are coherent with previous experiments. The estimation around $k\Delta + \Delta/2$ is less accurate, which is especially visible when only 10 bins are used (20 degrees of rotation angle).

	Orientation [deg]			
	0	10	20	70
10 bins ($\Delta = 36^o$)	0.64	n/a	28.08	70.54
16 bins ($\Delta = 22.5^o$)	0.65	8.35	n/a	66.95
20 bins ($\Delta = 18^o$)	0.28	12.61	n/a	70.13

Table 3.2: Orientation $\theta_{\mathbf{P},\hat{\mathbf{I}}}$ (Equation (3.21)) estimated for each combination of histogram length and orientation of the transformed image.

It is important to underline the extra achievement of this method. Histogram descriptors are built upon rotated versions of a patch (see Section 3.2). These rotations are performed with respect to the centre of the patch. However, the rotations performed on the test images are applied with respect to their own centre. This produces a distortion on the patches that is different from a rotation about the same angle from their respective centres. Therefore, our method demonstrates robustness in front of these small similarity transformations (not only rotation) of each patch.

More results of the RDTM with other similarity transformations (including small image scaling) have been reported in [ME07b]. In this case, a wide range of rotations (approximately a range of 180 degrees) is covered. Good accuracy in both the location of the patch and the orientation at each frame of a video sequence is achieved.

3.5.4 Comparison with similar techniques

This section assesses the performance of the RDTM in comparison to other similar techniques and is structured as follows. Firstly, the techniques compared are described. Secondly, the evaluation methodology is explained. Thirdly, results are depicted and discussed. Finally, the processing time of the various techniques is examined.

The matching techniques compared are listed below.

Rotation-exhaustive template matching (NCC-R) The NCC is computed at each point of the tested image and for each of the N rotated versions. Only the best result of the N correlations computed at each point is kept. Hence, this method is invariant to rotation transformations with the limitation of the number of versions used. For a fast implementation, the approach presented in [Lew95] is used (see Section 2.3.3, p.16).

Gray-level intensity histogram matching (IHM) The histogram of the intensity is compared at each point. In the intensity histogram descriptor, the central part of each patch has double weight to reduce the effect of peripheral pixels as for $\mathbf{h_P}$ (see Equations (3.2-3.3). Moreover, the descriptor of the patch that is sought is built upon the mean of intensity histograms of the N rotated versions. Although the intensity histogram of a single version should already be highly invariant to rotations, this averaging eliminates possible variations among rotated versions and hence produces a fairer comparison to the method proposed in this chapter. The similarity measure used is the Euclidean distance. Results using the Bhattacharyya distance [Kai67] showed similar behaviour and, consequently, are not shown here to avoid redundancy. This method is implemented with the integral histogram approach (see Section 2.3.3).

Gradient orientation histogram matching (GHM) The comparison of gradient orientation histograms is applied as in the RDTM (see Section 3.3.2, p.25). The difference is that the original GHSM (Equation (3.14)) is computed in this case at each point of the image and not only at the set of candidates S_m. This method is implemented with the integral histogram approach.

OH matching followed by OC matching (OH+OCM) Recalling from Section 2.3, this technique consists in a two step strategy [UK04]. Firstly, orientation code histograms (OH) are used to estimate the orientation of a patch in each point of an image. Secondly, orientation code matching (OCM) is applied only at the location of the best histogram matches and at the estimated orientation. The reason for adding this strategy to this comparison is threefold:

- to see the results for a larger data set than the one used in [UK04];
- to evaluate the improvement introduced by the proposed rotation-discriminative descriptor, which is one of the main differences with the OH+OCM method;
- to analyse the efficiency of this hierarchical search approach when compared to the RDTM.

The proposed RDTM In particular, the correlation map Ψ (Equation (3.16)).

Each one of these matching techniques is computed in a different manner targeting a broad range of possibilities. On the one hand, two sorts of information are used, either pixel intensity (NCC-R and IHM) or gradient (GHM), and in the case of OH+OCM and RDTM, a combination of both. And on the other hand, histogram matching (GHM and IHM) is compared to template matching (NCC-R) and to the mixed approaches, namely, OH+OCM and RDTM.

In the evaluation of Section 3.5.2, the correlation level is considered as a measure of performance. Another possibility is considered here. The purpose of all the compared methods is to find matches. A match is expected at a peak in the similarity. A correct match is found when the peak coincides with the ground truth. An incorrect match is found at a peak unrelated to the ground truth. These ideas are translated into the concepts of *true positives* and *false positives*, respectively. This nomenclature is often used in literature dealing with Receiver Operating Characteristic (ROC) curves [Faw06]. The performance can be given by these two values as follows: the higher the number of true positives and the lower the number of false positives, the better is the obtained performance.

A positive is defined as a point in the image for which the similarity is beyond a certain threshold. A single threshold gives a fixed value for the true positives and for the false positives. In order to obtain a curve for different values, the process consists simply in varying the threshold from maximum to minimum similarity values. Therefore, we compare all the techniques by parsing the similarity values obtain for each map independently. The true positives and false positives can now be defined mathematically. Given an image $\mathbf{I}$, a patch $\mathbf{P}$ extracted from $\mathbf{I}, \hat{\mathbf{I}} = T(\mathbf{I})$, and $(\hat{x}_{\mathbf{P}}, \hat{y}_{\mathbf{P}})$ being the correspondence of $(x_{\mathbf{P}}, y_{\mathbf{P}})$ into $\hat{\mathbf{I}}$, a *true positive* is

$$\mathrm{tp}_{\mathbf{P},\hat{\mathbf{I}},t} = \begin{cases} 1 & \text{if } \exists (x,y) \in \mathbf{G} \mid d(f(\mathbf{P}), f(\mathbf{R}_{x,y})) > t \\ 0 & \text{otherwise,} \end{cases} \tag{3.22}$$

Conversely, a *false positive* is

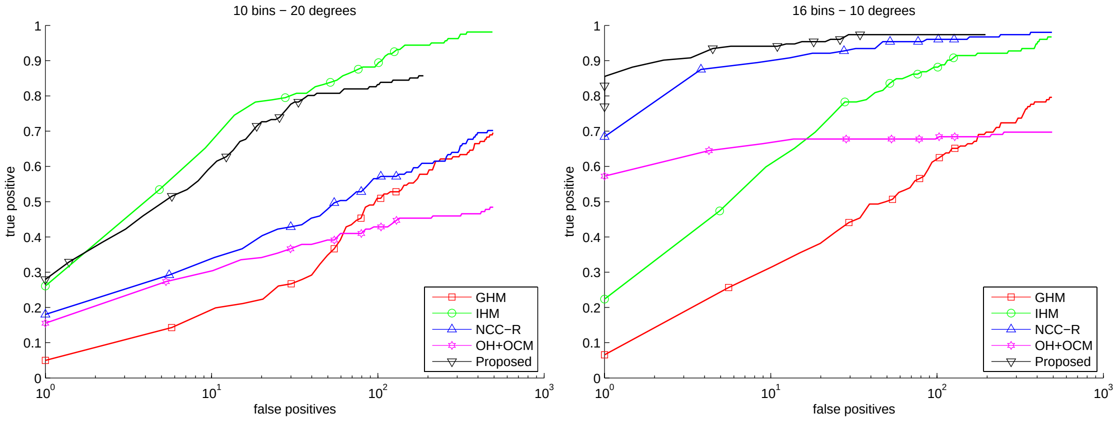
$$\mathrm{fp}_{\mathbf{P},\hat{\mathbf{I}},t}(x,y) = \begin{cases} 1 & \text{if} (x,y) \notin \mathbf{G} \mid d(f(\mathbf{P}), f(\mathbf{R}_{x,y})) > t \\ 0 & \text{otherwise,} \end{cases} \tag{3.23}$$

where t takes values in the range $[\max d(f(\mathbf{P}), f(\mathbf{R}_{x,y})), \min d(f(\mathbf{P}), f(\mathbf{R}_{x,y}))]$. Similarly to the evaluation in Section 3.5.2, these values do not give a perception of distance in pixels from the location of a false positive to the ground truth.

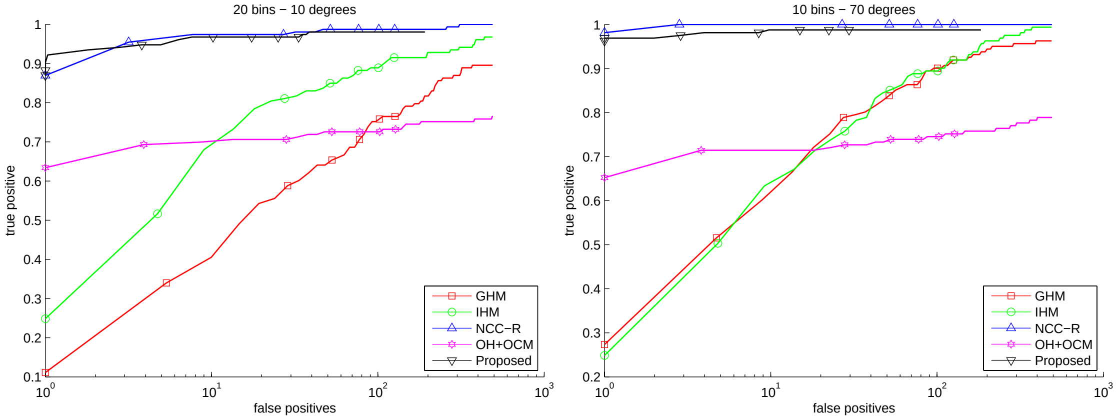
The comparison methodology is applied to the test set. The number of rotated versions used in the NCC-R method is the same as the number of bins in the histograms of the other maps. Consequently, its results may vary with different bin numbers. The response around $k\Delta + \Delta/2$ and $k\Delta$ of each similarity map is depicted in Figure 3.4. The results for the non-rotated images are given separately in Appendix A.

To give a complementary overview of the comparison, Table 3.3 shows the percentage of true positives (tp) obtained for a fixed number of false positives (fp). This number is fixed to 10.

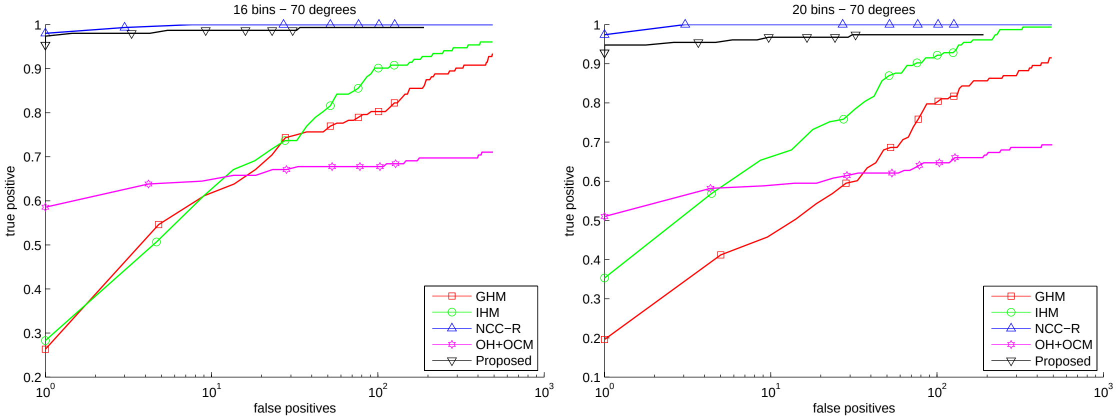
The NCC-R indicates a great performance almost independent of the number of candidates. This shows the high selectivity of this kind of map. In the case of the IHM, rotation invariance is evidenced by very similar results throughout the different cases. Moreover, for a small histogram length it achieves the best results when the rotation is around $k\Delta + \Delta/2$ (Figure 3.4a). A poorer selectivity is shown by the GHM as a large number of false positives is obtained in order to get a high probability of having a true positive. The results of the GHM are greatly improved when used as an input for further template matching as in RDTM (especially visible as the number of bins grows). Furthermore, the proposed descriptor and similarity measure achieve the desired rotation discrimination and hence accurate matching. The OH+OCM [UK04] has lower performance probably due to its non-invariant nor robust descriptor. Using only a single version to build the histogram is not enough to effectively face the variations in the histogram due to rotations.



(a) 10 bins, 20 degrees (b) 16 bins, 10 degrees



(c) 20 bins, 10 degrees (d) 10 bins, 70 degrees



(e) 16 bins, 70 degrees (f) 20 bins, 70 degrees

Figure 3.4: Average true positive (tp) and false positives (fp) among all the patches extracted at feature points. (a-c) rotation angle $\sim k\Delta + \Delta/2$. (d-f) rotation angle $\sim k\Delta$.

Rotation angle [deg]:	0			$\sim k\Delta+\Delta/2$			$\sim k\Delta$		
Number of bins:	10	16	20	10	16	20	10	16	20
NCC-R	100	100	100	34	94	97	100	100	100
RDTM	100	100	99	60	90	96	98	98	96
OH+OCM	100	100	100	30	67	70	71	65	59
GHM	87	96	93	19	31	40	62	62	46
IHM	54	64	75	67	60	69	64	63	66

Table 3.3: Percentage of true positives (tp) for a fixed fp=10, among all the patches extracted at feature points.

It could be argued that the selection of patches according to the matching method (see Section 3.5.1, p.28) biases the results favouring the RDTM. In order to provide a baseline of performance, another set of image patches has been extracted. For each image of the original test set, 15 patches are extracted randomly at different points. The size of the patch is also selected randomly with the same range as before, namely, 10×10 to 20×20 pixels. The experiment is applied to this new set of patches obtaining the results given separately in Appendix A. Table 3.4 shows the percentage of true positives obtained for a fixed fp=10 with patches extracted randonly.

Rotation angle [deg]:	0			$\sim k\Delta+\Delta/2$			$\sim k\Delta$		
Number of bins:	10	16	20	10	16	20	10	16	20
NCC-R	100	100	100	11	57	81	98	96	98
RDTM	94	92	90	30	68	79	91	91	83
OH+OCM	92	94	92	23	56	63	57	65	52
GHM	61	82	72	07	19	30	42	46	38
IHM	39	50	58	48	56	65	52	60	63

Table 3.4: Percentage of true positives (tp) for a fixed fp=10, among all the patches extracted randomly.

Contrary to what could be expected, the new set of patches does not have a big impact and the results are very similar. One possible reason is that most images in the set are textured. However, it can be deduced that patches without such texture (e.g., those extracted from the flat region of image Cathedral) induce a poorer performance almost in all the matching techniques compared.

Computational complexity

The processing time needed to produce a match for each of the compared techniques is analysed here. The time needed by any algorithm depends on memory access speed, architecture, and code optimisation, among others. Hence, variations may occur and only an approximation of the complexity is given here. With this approximation it is possible to have an idea of the order of magnitude of the processing time needed by each method. Table 3.5 shows this time (averaged for the patches in the test set) when computed with a Pentium M processor at 1700 MHz.

	Number of bins		
	10	16	20
OH+OCM	4.48	4.71	4.93
NCC-R	0.90	1.25	1.52
GHM	0.43	0.88	1.30
RDTM	0.18	0.20	0.26
IHM	0.11	0.13	0.15

Table 3.5: Average processing time in seconds for a single patch.

As it can be seen, the slowest algorithm is OH+OCM. The main reasons are the circular mask used for OC matching and, consequently, the impossibility of using the integral histogram approach. The NCC-R uses the integral image as in [Lew95]. Despite this fast matching implementation, it can be seen that comparing each rotated version of the template is inefficient. Estimating the orientation in the histogram matching step of the RDTM drastically palliates this inefficiency. Moreover, the hierarchical selection proposed here enables a processing time almost as fast as the one obtained with the most simple and efficient strategy, which is the IHM (implemented with the integral histogram). It should be pointed out that reducing the number of candidates kept from the histogram matching (fixed to 500 in these experiments) would further reduce the computational cost.

3.6 Application to top-view visual tracking

Registration of objects is one of the numerous challenges in visual tracking [YJS06]. For instance, visual recognition is necessary for top-down tracking techniques using interest points for filter update (e.g., [ZC93, CH96, Dav03, PC05]). In these works, template matching is often employed for point correspondence. When the object rotates with respect to the optical axis of the camera, correlation fails and the update is incomplete. More details are given later in Section 5.3.

However, none of these techniques exploit directly the viewpoint determined at recognition level. We describe here a possible way to use the RDTM to provide an extra input to a generic top-view 2D filter-based tracking system. We consider a top-view 2D tracker as a system using a camera situated perpendicular to a plane on which the objects describe their trajectories. This generic tracker is governed by the following motion and measurement models, respectively,

$$\begin{aligned} \mathbf{x}_k &= u(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) \\ \mathbf{z}_k &= v(\mathbf{x}_k, \mathbf{r}_k), \end{aligned} \tag{3.24}$$

where $\mathbf{x} = (\hat{x}, \hat{y})$ is the state vector, $\mathbf{z}$ the measurement vector, u and v are possibly non-linear functions, $\mathbf{q}$ and $\mathbf{r}$ are the process and measurement noises, respectively, and k is the current frame. The state vector represents the position of the target in two dimensions. The measurement $\mathbf{z}$ used for filter update is the similarity map of the target with an area $\mathbf{A}_k$ of the current frame containing the expected location of $\mathbf{x}_k$ (often called the gating region). One straight possibility is to use the correlation part of the RDTM. In that case, the likelihood of the filter

$$p(\mathbf{z}_k|\mathbf{x}_k) \propto \Psi_k(x,y) \; \Big| \; (x,y) \in \mathbf{A}_k. \tag{3.25}$$

Although the correlation map Ψ has shown good accuracy (see previous section), some false positives may still appear in the gating region which could induce erroneous updates. The question now is how to take advantage of the orientation part of the RDTM to overcome this problem. The orientation of the neighbourhood of the target is determined by the tangential direction θ of the trajectory followed by the target $\mathbf{x}_{0:k} = (\hat{x}_{o:k}, \hat{y}_{o:k})$. This direction can be obtained recursively from the previous and current frames, one has

$$\theta_k = \theta_{k-1} + \arctan(\hat{y}_k - \hat{y}_{k-1}, \hat{x}_k - \hat{x}_{k-1}). \tag{3.26}$$

If the target's descriptor is initialised at a reference orientation θ_0, then the estimated orientation Θ at the current frame k is a valid measurement to correct the filter state

$$p(\mathbf{z}_k|\mathbf{x}_k) \propto \Psi_k(x,y) \cdot \exp\left(- \left[\frac{\theta_k - \Theta_k(x,y) \cdot \Delta}{\alpha \cdot \Delta} \right]^2 \right) \; \Big| \; (x,y) \in \mathbf{A}_k, \tag{3.27}$$

where α is a parameter to tune the constrain imposed by the orientation. False positives can then be discarded in a straightforward manner using the state of the filter. This is valid for objects that move forwards, backwards or turn with respect to the camera to move in oblique directions. In the case where an object could move in oblique directions without turning, this refinement of the likelihood would not be suitable.

This idea is exploited in the camera tracking framework presented in the second part of this thesis. The idea is that the camera pose, especially its 3D orientation, has a rough correspondence to the rotation of the patch of a feature point used as 3D reference. This is exploited to refine the correction of the camera pose estimation.

3.7 Summary

An efficient method to perform 2D rotation discriminative template matching has been presented in this chapter. This is achieved with a hierarchical search that iteratively refines the candidates. Firstly, matching is done according to the gradient magnitude. Secondly, orientation histogram matching is applied to estimate the orientation of the patch that it represents. Finally, template matching is computed at the most probable locations with the corresponding orientations. The main contributions of the method are the rotation-discriminative descriptor and the efficient matching strategy.

Experimentation shows that our results are as good performing as NCC of each rotated version of a template (called here NCC-R method) but with an average speed up factor of six. In addition, performance and efficiency of our technique is superior to the most similar technique in the state-of-the-art, namely, the OH+OCM [UK04]. Furthermore, experiments on randomly selected regions show the high performance of the proposed method. This fact indicates that our RDTM is not only applicable to feature points extracted around corner points but also to any region, provided that enough texture information is available.

An achievement of the method is to bring together not only localisation information, but also its local orientation in an efficient way. A specific application of this accomplishment to refine the state correction step of top-view 2D visual tracking system has also been described. In the second part of this thesis, this matching method is used to recognise feature points that are used to estimate the position of a mobile camera.

Part II

Video-based camera tracking

Theoretical background

4

This chapter explains the concepts encompassed by the framework proposed in this thesis. This framework, as described in Chapter 1, can be summarised as a camera tracker using data fusion within a filter-based approach. Therefore, concepts related to 3D tracking, camera properties, filtering and merging of information must be discussed.

Firstly, the properties and several devices related to 3D tracking in general are depicted. Secondly, Section 4.2 deals with the parameters that relate the world coordinate system with that of the camera. Thirdly, diverse bayesian filters related to the proposed approach are detailed. Finally, the different methods of data fusion are explained in the last section.

4.1 Tracking

Tracking in general is the localisation and trailing of a certain element in the space where it moves. This high level definition is applied to a large number of scenarios depending on the type of element, the limits assumed or known from the motion space, and also on the sort of sensor(s) employed.

Among the vast spectrum of scenarios, one can first differentiate between 2D and 3D tracking depending on the motion space.

2D tracking is applied when the motion of an element in 3D space is only relevant within two directional axes. For instance, let us imagine that a camera is used to record people's motion from the ceiling of a room. In this case, only motion perpendicular to the camera is pertinent. In some other more general environments, the real motion is not only over a perpendicular plane, and features such as scale are added to the motion space. However, even in such a case tracking is addressed as a 2D problem. The interested reader is referred to a recent survey on this topic [YJS06].

3D tracking, on the other hand, describes motion in three dimensions. More specifically, the 6 DoF –three orientation axes and three positional directions– of the element with respect to a defined world coordinate frame are tracked. For instance, in virtual reality applications, the pose of the head of the user must be tracked in all directions in order to render the appropriate view of the virtual world. In 3D tracking the limitations or assumptions come either from the element itself or from the type of sensor used. This thesis concentrates on this second class of tracking.

The remainder of this section gives, firstly, specific details about performance properties of any tracker. Following this description, several sensors for 3D tracking are described. This will help to better situate the video sensor used in the proposed framework among the existing possibilities.

4.1.1 Tracker properties

The design and evaluation of a tracking system strongly depends on the final application. Several factors have to be considered and not all of them are relevant for each application. The most relevant factors have been identified by Burdea and Coiffet [BC03]:

Latency is the delay between the change of the position and/or orientation of the target being tracked and the report of the change by the tracking system [BB93]. For instance, in an Augmented or Virtual Reality environment, if the latency is greater than 50 milliseconds, it will be noticed by the user and the scene will not seem realistic.

Update rate is the number of times per second that the tracker device reports data to the computer. Depending on the trackers it can vary between 20 or less, and more than 1k updates per second. The pose update rate must be at least twice the true target motion bandwidth. Otherwise, an estimator may track an alias of the true motion. Given that common arm and head motion bandwidth specifications range from 2 to 20 Hz [FDS90], in the same example environment as above, the sampling rate should ideally be greater than 40 Hz.

Accuracy is a measure of the error in the position and orientation reported by the tracker. In trackers based on emitters and receivers, the accuracy decreases as the user moves away from the fixed reference point [BB93]. In accuracy terms, one can also define jitter and drift.

Jitter is the change in tracker output when the sensor is stationary. In the same environment as above, jitter can be observed as a rapid fluctuation of the whole VR environment or of the added virtual object with respect to the real world (AR). This can be quite annoying for a user interface.

Drift corresponds to the steady increase in tracking error with time. In robot navigation, for instance, if the tracking incrementally accumulates errors, a robot may end up finding itself in a totally different position when compared to that estimated by its tracker. Such a situation may require re-initialisation, which can be considered a failure in robot navigation.

Resolution is the smallest change in position and orientation that can be detected by the tracker. It strongly depends on the type of sensor used. An example where resolution is a critical factor is assisted or tele-operated medical surgery.

As can be seen, some properties have strong relevance in determined applications. Addressing them all at the same time is and has been the target of researchers in tracking. The next section is dedicated to the various tracking devices that have been produced by these researchers.

4.1.2 Tracker devices

Current tracking devices are based on a large number of different technologies. A short presentation of each of these sensors is given hereafter, together with a discussion of their advantages and disadvantages. This presentation is followed by a summary table. The interested reader can find more references and detailed descriptions in [ABW01, WF02, BC03, AME05].

Mechanical trackers measure displacement or bending of rigid or semi-rigid pieces. These pieces can convert their bending or mechanical pressure into electric signals. These signals can then be interpreted as relative position or rotation.

The study of body movements has increased the performance level of sport disciplines. Mechanical-based devices are often used for this purpose as well as for hand motion analysis. Different equipments are available: from body-based to hand haptic devices. Although in principle they have an unlimited range, the former may be an unpracticable solution if the user needs large free movement. Force-feedback gives the latter an additional touch perception in a mixed or virtual reality experience.

Commercial examples of such technology are the Haptic Workstation (two-handed Cyber-Force systems for virtual prototyping) and the CyberGlove, both from Immersion Corporation [Imm04].

Acoustic trackers sense sound waves. To measure a distance, a transmitter–receiver pair is needed to calculate the speed of sound. Two techniques are commonly used: time of flight and phase coherence. Commercial acoustic ranging systems use the time-of-flight of a pulse, typically of ultrasonic waves, from the source to the destination. Phase coherence systems are based on the principle that certain types of waves, i.e. ultrasonic, keep phase during their lifetime. Knowing the frequency and phase at the origin, and the shift at the end, position can be estimated from the difference within a wavelength cycle. However, isolating phase may be inconsistent as the phase difference could exceed a period when motion is too fast. As a result, acoustic sensors perform best at low motion. The main disadvantage of acoustic trackers is the slowness of sound travel, which increases latency. Also, the environment can create occlusion or wave repetitions by reflecting the signal [Val02].

An example of an acoustic tracker is the Logitech's 3D Mouse created by FakeSpace Labs, Inc. consisting of 3 ultrasonic speakers set in triangular position [Fak04].

GPS or Global Positioning System is also based on a transmitter–receiver structure. The GPS receiver calculates its position by measuring the distance between itself and three or more

GPS satellites using GPS microwave time-of-flight. It is generally used for outdoor tracking as the buildings interfere with satellite signals. Accuracy is also influenced by atmospheric effects. An enhancement of this tracker is the Differential GPS (DGPS). This technology uses a network of ground short-range transmitters that broadcast the difference between the position indicated by the satellites and their true position. DGPS has a better resolution than GPS.

Optical trackers use optical sensing to determine the real-time position and orientation of an object. They consist of two components: targets and optical sensors. A possible way to classify the optical trackers is based on the use of active or passive targets.

An example of an active target is the Infrared Light Emitting Diode (ILED). Typical sensors used to detect these active targets are the lateral-effect photodiode [WAB+90] and quad cells [KRC97]. They determine the centroid of the light on the image plane. Because of the high speed of light, these trackers have a small latency. Other advantages are the high update rate and the large range of activity. However, optical trackers require direct line-of-sight (no occlusion between emitter and receiver). A tracker using this technology is the HiBall [HiB07].

Passive targets (e.g. markers) and sensors (e.g. CCD cameras) are treated separately in the vision-based category.

Vision-based trackers use the video stream provided by a camera (e.g., CCD, WebCam) and natural or calibrated elements in the environment. The elements are passive targets with no emission of information besides reflecting ambient light. Motion between video frames or motion from a calibrated position is employed to track the camera. When motion is slow, this tracker produces very accurate estimations. Since the necessary equipment is very limited (and often not expensive), this type of tracking has attracted the interest of most researchers in the last decades. An in-depth description of this research is given in the next chapter.

Magnetic trackers employ a magnetic field generated by a stationary transmitter to determine the real-time position of a moving receiver [RBSJ79]. A calibration algorithm is used to determine the position and orientation of the receiver relative to the transmitter. Since the magnetic field penetrates dielectrics, this tracking category overcomes the problem of line of sight. Other advantages of magnetic tracking are convenience of use and high accuracy. However this accuracy is seriously degraded by magnetic fields due to ferromagnetic or conductive material in the environment. A representative magnetic tracker is the FASTRAK from Polhemus [FAS07].

Inertial trackers calculate position and orientation using accelerometers and gyroscopes. Inertial trackers are self-dependent, i.e., there is no transmitter and receiver system. This is an advantage over acoustic or optic rangers where there is a need for a direct line-of-sight. Being chip-implementable, passive, with low latency and without environment requirements, makes them one of the most efficient tracking systems. Drawbacks arise from the double-integrator needed to calculate position and the single integrator for orientation [AME05]. A bias in the accelerometers induces drift in position estimation. At low frequency (low motion), noise in measurement is confused with acceleration. Whether there is real low motion or the user is still, the tracker could produce the same estimation [ABW01]. Gyroscopes are affected by similar problems. Their good performance in fast moving environment though, makes them a good candidate for a hybrid solution, as described further in Section 5.4. Examples of commercial products are: InterSense's InterTrax2, InterCube2, and IS-300 Pro [Int05].

Table 4.1 summarises the main features of the tracker devices described above.

Device category	Principle	Mobile	Particularities
Mechanical	Displacement or bending of semi-rigid pieces (Self-contained).	Yes	
Acoustic	Time-of-flight or phase coherence of sound waves (T-R).	No	Latency due to sound waves.
GPS	Satellite triangulation with microwave time-of-flight (T-R).	Yes	
Optical	Sensing of active light targets (T-R).	Yes	Require line-of-sight.
Vision	Sensing of passive light targets (R).	Yes	Require line-of-sight.
Magnetic	Magnetic fields (T-R).	No	Influenced by ferromagnetic or conductive material.
Inertial	Accelerometers and gyroscopes (Self-contained).	Yes	Drift with low motion.

Table 4.1: Summary of tracker devices description. Notation: T, transmitter; R, receiver.

4.2 Camera geometry

The camera geometry establishes the relation between the world in three dimensions and the plane of the image where this world is projected. This geometry is modeled mathematically to express the distortions of converting the 3D continuous space into a 2D representation. This section briefly depicts the most relevant aspects of camera geometry. Detailed descriptions can be found in the book by Hartley and Zisserman [HZ00].

The most extended model, which is used in this thesis, is the pinhole camera model. It is designed for CCD like cameras but is also valid for other imaging sensors. The model uses the back-projection of a point in space. The back-projection is the line that joints a point **X** in 3D space with the centre of the camera **C** (known as centre of projection), also in 3D. The intersection of this line with the image plane is the point **x**, which represents **X** in two dimensions (see Figure 4.1). The image plane is placed perpendicularly to the Z axis at $Z = f$, where f is the focal length of the camera. Although this plane is mathematically represented as an infinite plane, only points inside the camera's FoV have a representation in the image plane.

In order to find the coordinates of the point **x** in the image plane, one can use the structure depicted in Figure 4.2 for the y axis of the image plane. By similar triangles, one has the following

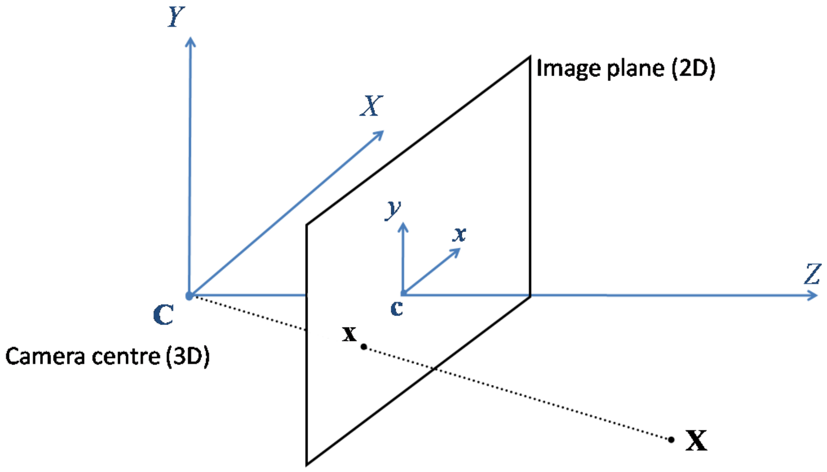


Figure 4.1: Pinhole camera model. A 3D point **X** is represented as a point **x** lying on the intersection of the image plane and the line that joints **X** with the camera centre **C**.

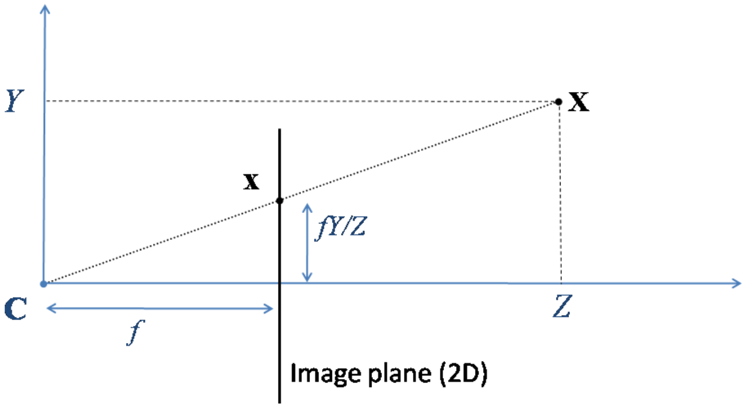


Figure 4.2: Mapping between 3D and 2D space. Representation for the Y axis.

mapping from Euclidean $\mathbb{R}^3$ to $\mathbb{R}^2$

$$\mathbf{X} = [X, Y, Z]^\top \mapsto [fX/Z, fY/Z]^\top = \mathbf{x} \tag{4.1}$$

where $\top$ is vector or matrix transpose. This mapping can be further expressed in homogeneous coordinates

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{4.2}$$

The matrix in the above formulation is referred to as *camera projection matrix* **P**. This matrix is commonly factorised into two separate matrices. In this case,

$$\mathbf{P} = \mathrm{diag}(f, f, 1)[\mathbf{I}|\mathbf{0}], \tag{4.3}$$

where diag$(\cdot)$ is a diagonal matrix and $[\mathbf{I}|\mathbf{0}]$ represents a 3x4 matrix further divided into a 3x3 identity matrix and a 3x1 column vector (in this case a zero vector). So far, the camera projection matrix has been expressed with respect to the camera coordinate system. In general, it is more convenient not to make the camera centre explicit and express the projection matrix with respect to a fixed world coordinate frame. In this case, the previous factorisation is reformulated as

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R}|\mathbf{t}], \tag{4.4}$$

where $\mathbf{K}$ is the 3x3 camera calibration matrix, $\mathbf{R}$ represents the rotation and $\mathbf{t}$ the translation of the camera with respect to the world coordinate frame. By comparing to Equation (4.3), it can be deduced that when the world and camera coordinate frames coincide, the rotation is logically the identity matrix and the translation is a zero vector. Figure 4.3 shows the spatial relation of the projection matrix factorisation. The calibration matrix $\mathbf{K}$ depends on the internal parameters of the camera such as the focal length. $[\mathbf{R}|\mathbf{t}]$ is the 3x4 matrix of external parameters and its computation is the core of all camera trackers. Estimating these parameters is commonly known as the pose estimation problem.

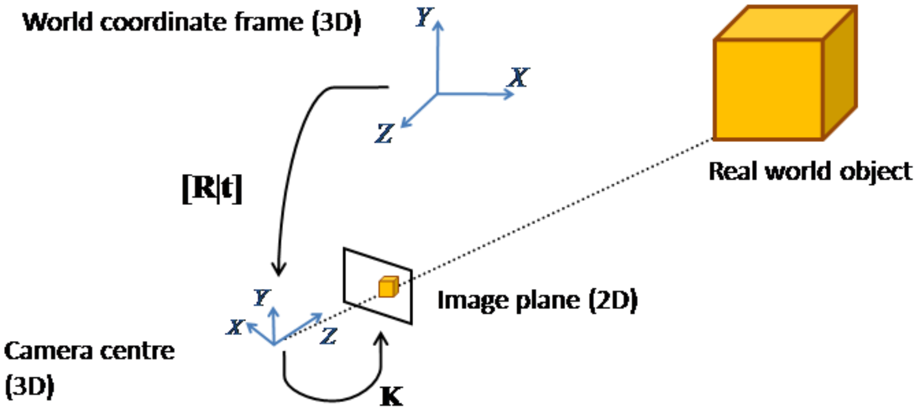


Figure 4.3: Relation between world, camera and image coordinate systems through the camera projection matrix.

Finally, it is possible to formulate in homogeneous coordinates the relation between a point in the world coordinate frame $\mathbf{X}$ with its representation in the image plane $\mathbf{x}$ using Equations (4.2) and (4.4)

$$\lambda \cdot \mathbf{x} = \mathbf{P}\,\mathbf{X} = \mathbf{K}\,[\mathbf{R}|\mathbf{t}]\,\mathbf{X}, \tag{4.5}$$

where λ is a proportion factor usually fixed so that the third component of the homogeneous vector $\mathbf{x}$ is equal to 1.

Following is a description of the internal parameters. Then, the end of this section gives a description of possible parameterisations of the rotation matrix. As for the external parameters, examples of how to compute them are given in the next chapter.

4.2.1 Camera internal parameters

The calibration matrix **K** defines the internal relation between the camera centre and the image plane. This relation is defined by physical properties such as pixel capture layout. The calibration matrix is generically expressed as follows

$$\mathbf{K} = \begin{bmatrix} fm_x & s & c_x \\ 0 & fm_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.6}$$

The focal length f is known from the previous section. Each of the remaining terms defines a different property.

m_x m_y are the number of pixels per unit distance in image coordinates, for each axis. They express the scale factor from Euclidean coordinates to image coordinates. In the previous case where $\mathbf{K} = \mathrm{diag}(f, f, 1)$, the size is 1x1 pixels per unit distance. In commercial cameras, this size is different and modeled with m_x and m_y for each axis, respectively.

c_x c_y are the image coordinates of the intersection of the optical axis and the image plane **c**, generally known as the principal point (see Figure 4.1). In many cases, the origin of the image coordinate system is placed at one corner of the image plane for implementation convenience.

s is the skew factor. It expresses the distortion introduced by non-square pixels. Therefore, this value is not zero when x and y axes are not perpendicular, which is unusual in modern cameras.

Since the projection of the real world into the image plane is a necessary computation, accurate transformations between both coordinate systems are necessary. As said before, the computation of external parameters is the main load of video-based camera trackers during run-time. Therefore, computing the camera calibration matrix off-line is beneficial for most tracking systems. Examples of calibration methods can be found in [HZ00].

4.2.2 Parameterisation of the rotation

The estimation of the pose of a camera necessarily determines the rotation that relates camera and world coordinate systems. Rotational quantities are more difficult to represent than linear quantities such as the translation vector. Several methods to express these quantities exist and not all of them are suitable for camera tracking. This section describes the different mathematical tools available to express rotations.

The first and most straightforward expression is with *Euler angles*. In the rotation theorem of Euler, he stated:

> Any two independent orthonormal coordinate frames can be related by a sequence of rotations (not more than three) about coordinate axes, where no two successive rotations may be about the same axis.

This means that one can represent a rotation with 3 numbers. A sequence of rotations around principle axes is called an Euler Angle Sequence. But there are various Euler angle conventions. Firstly, the terms of the angles change depending on the area where they are used, for instance, navigation, computer graphics, etc. Secondly, also the order in which the angles are applied changes and in mathematical terms rotations are not commutative. Thirdly, rotations can be left or right handed.

One possible specification is to use Roll (ϕ), Pitch (θ) and Yaw (ψ). These represent a rotation of the camera about the X, Y, and Z axes, respectively (see Figure 4.4). The sequence is expressed as Yaw/Pitch/Roll (ψ, θ, ϕ). This means that it starts with a rotation by ψ about the Z axis, followed by a rotation by θ about the new Y axis (i.e. axes have been modified by the previous rotation), followed by a rotation by ϕ about the new X axis.

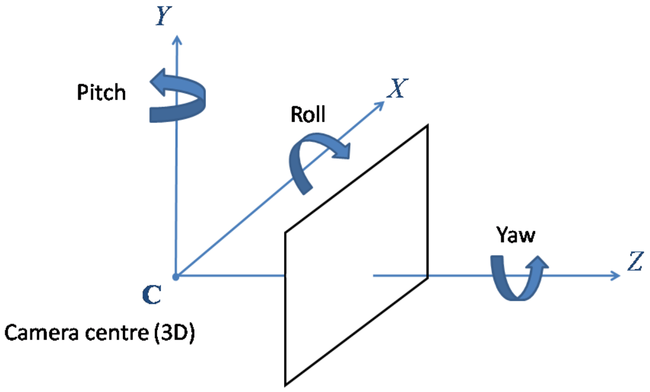


Figure 4.4: Euler Angle Sequence

These angles of rotation are expressed mathematically by concatenation of individual rotation matrices. Indeed, general rotations in 3D can be expressed as three successive rotations about different axes.

- Rotation ϕ about X axis,

$$\mathbf{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & sin(\phi) \\ 0 & -sin(\phi) & cos(\phi) \end{bmatrix} \tag{4.7}$$

- Rotation θ about Y axis,

$$\mathbf{R}_\theta = \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{4.8}$$

- Rotation ψ about Z axis,

$$\mathbf{R}_\psi = \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.9}$$

The full transformation around the three axes can be expressed as the product of these three separate transformations. A rotation in space using the Yaw/Pitch/Roll convention defined before is achieved with the following matrix

$$\mathbf{R} = \mathbf{R}_\phi \cdot \mathbf{R}_\theta \cdot \mathbf{R}_\psi. \tag{4.10}$$

As the matrix multiplication is not commutative, different order in multiplication represents different final rotations.

By multiplying the vectorial expression of a point $\mathbf{X}$ with one of these matrices, one obtains the location of that point after such rotation. Therefore, a rotation matrix is convenient for transforming points because only a simple and efficiently implemented matrix multiplication is required. However, they are inappropriate for filtering or interpolation. More concretely, the angles interact between them, which is visually perceived after multiplying the three matrices. If each Euler angle is interpolated or filtered independently, the interaction is not taken into account. In addition, one potential problem that Euler angles suffer from is what is called gimbal lock. This occurs when two of the three rotation axes align, resulting in a temporary loss of a degree of freedom as one rotation has no effect.

These problems can be solved with other alternatives such as quaternions and exponential maps. Quaternions are used in the proposed camera tracking framework. Therefore, they are hereafter explained. A description of exponential maps can be found in [LF05].

Quaternions are, together with Euler angles, the most common way to represent rotations. A *quaternion* is a four-element vector [Ham63]

$$\mathbf{q} = [W, X, Y, Z] \tag{4.11}$$

where W, X, Y, Z are reals. It can also be expressed as a sum of four elements of the form $\mathbf{q} = W + Xi + Yj + Zk$, with $i^2 = j^2 = k^2 = ijk = -1$. A quaternion can represent a rotation, expressed in radians, about an arbitrary three dimensional axis. The relation between the quaternion and this representation is

$$\begin{aligned} \alpha &= 2\arccos(W) \\ \mathbf{v} &= [X, Y, Z] \\ \mathbf{q} &= \left[\cos(\alpha/2), \frac{\mathbf{v}}{\|\mathbf{v}\|}\sin(\alpha/2)\right] \end{aligned} \tag{4.12}$$

where $\mathbf{v}$ is the vector along the axis of rotation $Xi + Yj + Zk$, and α is the rotation angle.

Quaternions have several advantages when compared to Euler angles. First of all, as mentioned before, they resolve the problem of gimbal lock. Moreover, they have a convenient expression when interpolation is required [ABW01].

However, there are also some disadvantages. Rigid body dynamics represented by quaternions cannot be characterised by linear equations. When filtering quaternions for tracking purposes, a non-linear model must be used. Filtering with non-linear equations needs more complex operations, and the processing time will increase in some way. Filtering is the topic of the next section, covering the linear/non-linear difficulties just mentioned.

4.3 Bayesian filtering

Bayesian filtering is extensively used for camera tracking. The main reasons are that it enforces time coherence, error estimates can be obtained from covariance measures, and feature positions can be predicted in incoming frames. In this section, an introduction to the concepts behind Bayesian filtering and example filters relevant to this thesis are given. Complementary information can be found in [May79, DdG01, AMGC02, Che03].

Before getting into the equations that govern Bayesian filtering it is necessary to define some concepts. Since filtering is used here in video-based environments, the time line is indexed with frames.

Filtering is an operation that consists in the estimation of a certain quantity at frame k. This estimation is computed using data measured along frames up to and including k [Che03]. It is divided generally in two steps: *prediction* and *correction*.

Prediction is an *a priori* form of estimation. Its aim is to derive information about what the quantity of interest will be like at some frame $k+\triangle k$ in the future (where $\triangle k > 0$) by using data measured up to and including frame k [Che03]. In this thesis, prediction refers to prediction of one-frame ahead ($k+1$).

Correction or Update is an *a posteriori* form of estimation. Its aim is to rectify the prediction for frame k, predicted in frame $k-1$, given the measurement obtained at frame k.

Let us now get into the mathematical part. Assume the following stochastic filtering model in a dynamic state-space form

$$\begin{aligned} \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) & (4.13)\\ \mathbf{z}_k &= g(\mathbf{x}_k, \mathbf{r}_k). & (4.14) \end{aligned}$$

Equation (4.13) is the process or state equation, where $\mathbf{x}_k$ is the state vector at frame k and $\mathbf{q}_k$ is the process noise. This equation characterises the process or *transition prior model* $p(\mathbf{x}_k|\mathbf{x}_{k-1})$. Equation (4.14) is the measurement or observation equation, where $\mathbf{z}_k$ is the measurement vector and $\mathbf{r}_k$ is the measurement noise. This equation characterises the *measurement noise model* $p(\mathbf{z}_k|\mathbf{x}_k)$, also known as *likelihood*. f and g are possibly non-linear, time-varying functions. Sometimes, these functions have another parameter called the control vector. This is common in robotics, for instance, where the motion is mechanically controlled and hence the filter is constrained not only by measurements. This parameter is not included here without loss of generality. $\mathbf{q}$ and $\mathbf{r}$ are generally known as the hyper-parameters of the system model.

Using this mathematical formulation, it is possible to redefine filtering as follows. Filtering is an operation that consists in the estimation of the *posterior density* $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. This estimation is computed using an initial density $p(x_0) \equiv p(x_0|z_0)$ known as *prior density*, the prediction at frame k given the previous state $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, and the likelihood of the incoming measurements according to the current prediction $p(\mathbf{z}_k|\mathbf{x}_k)$. This definition can be explained via the Bayes rule [Che03]

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{1:k}) &= \frac{p(\mathbf{z}_{1:k}|\mathbf{x}_k)\,p(\mathbf{x}_k)}{p(\mathbf{z}_{1:k})} && (4.15)\\ &= \frac{p(\mathbf{z}_k,\mathbf{z}_{1:k-1}|\mathbf{x}_k)\,p(\mathbf{x}_k)}{p(\mathbf{z}_k,\mathbf{z}_{1:k-1})}\\ &= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1},\mathbf{x}_k)\,p(\mathbf{z}_{1:k-1}|\mathbf{x}_k)\,p(\mathbf{x}_k)}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})\,p(\mathbf{z}_{1:k-1})}\\ &= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1},\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{z}_{1:k-1})\,p(\mathbf{z}_{1:k-1})\,p(\mathbf{x}_k)}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})\,p(\mathbf{z}_{1:k-1})\,p(\mathbf{x}_k)}\\ &= \frac{p(\mathbf{z}_k|\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}, && (4.16) \end{aligned}$$

where the normalising constant is

$$p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{z}_{1:k-1})\,d\mathbf{x}_k, \qquad (4.17)$$

and the prediction from previous measurements is related to the transition prior model as follows

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) &= \int p(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{z}_{1:k-1})\,p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})\,d\mathbf{x}_{k-1}\\ &= \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\,p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})\,d\mathbf{x}_{k-1}. && (4.18) \end{aligned}$$

Note that the second step of this equation is possible because the transition dynamics are modeled as a Markov process of order one.

The recurrence of Equation (4.18) followed by Equation (4.16), form the basis of optimal Bayesian filtering. The problem of this recurrence is that in general it cannot be determined analytically. Solutions to this problem exist under certain conditions. Depending on the linearity or non-linearity of functions f and g, there are several solutions proposed in literature. Furthermore, the properties of the transition and measurement probability distributions, determine the type of filter that can be applied to the tracking problem at hand. The remainder of this section describes the most common linear Gaussian-based filter, the Kalman Filter (KF), as well as the opposite case, where no assumption is made on the densities, and the functions are not necessarily linear. This is the case of a Particle Filter (PF).

4.3.1 Kalman filter

The Kalman filter (KF) assumes that the posterior density is Gaussian and its analytical computation is possible when the functions f and g are linear. In this case, the stochastic filtering model can be rewritten as

$$\begin{aligned} \mathbf{x}_k &= \mathbf{F}_k\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1} && (4.19)\\ \mathbf{z}_k &= \mathbf{G}_k\,\mathbf{x}_k + \mathbf{r}_k. && (4.20) \end{aligned}$$

The covariances of $\mathbf{q}_{k-1}$ and $\mathbf{r}_k$ are $\mathbf{Q}_{k-1}$ and $\mathbf{R}_k$, respectively. The system, measurement and noise matrices are allowed to be time varying.

The recursive relation previously described, can be specified for the KF as follows

$$\begin{aligned} p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) &= \mathcal{N}(\mathbf{x}_{k-1};\mathbf{m}_{k-1|k-1},\mathbf{P}_{k-1|k-1}) && (4.21)\\ p(\mathbf{x}_{k}|\mathbf{z}_{1:k-1}) &= \mathcal{N}(\mathbf{x}_{k};\mathbf{m}_{k|k-1},\mathbf{P}_{k|k-1}) && (4.22)\\ p(\mathbf{x}_{k}|\mathbf{z}_{1:k}) &= \mathcal{N}(\mathbf{x}_{k};\mathbf{m}_{k|k},\mathbf{P}_{k|k}), && (4.23)\end{aligned}$$

where $\mathcal{N}(x;m,P)$ is a Gaussian density with argument x, mean m and covariance P,

$$\begin{aligned} \mathbf{m}_{k|k-1} &= \mathbf{F}_k\,\mathbf{m}_{k-1|k-1} && (4.24)\\ \mathbf{P}_{k|k-1} &= \mathbf{F}_k\,\mathbf{P}_{k-1|k-1}\,\mathbf{F}_k^T+\mathbf{Q}_{k-1} && (4.25)\\ \mathbf{m}_{k|k} &= \mathbf{m}_{k|k-1}+\mathbf{K}_k\,(\mathbf{z}_k-\mathbf{G}_k\,\mathbf{m}_{k|k-1}) && (4.26)\\ \mathbf{P}_{k|k} &= \mathbf{P}_{k|k-1}-\mathbf{K}_k\,\mathbf{G}_k\,\mathbf{P}_{k|k-1}. && (4.27)\end{aligned}$$

And, finally,

$$\begin{aligned} \mathbf{S}_k &= \mathbf{G}_k\,\mathbf{P}_{k|k-1}\,\mathbf{G}_k^T+\mathbf{R}_k && (4.28)\\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\,\mathbf{G}_k^T\,\mathbf{S}_k^{-1} && (4.29)\end{aligned}$$

are the covariance of the innovation term $\mathbf{z}_k-\mathbf{G}_k\,\mathbf{m}_{k|k-1}$, and the Kalman gain, respectively. A more extended description and mathematical derivation can be found in [ABW01].

Under the Gaussianity assumption, the Kalman filter is the best filter to solve the optimal solution. This fact has made the KF the most common approach to tracking environments where Gaussian densities can be assumed. In cases where the process and/or measurement equations are not linear, other variants exist to approximate the optimal solution. These are, among others, the Extended Kalman Filter (EKF)[ABW01] and the Unscented Kalman Filter (UKF)[JU97][WVdM01]. The former is used as a first approximation to non-linear f and g functions. Whereas the latter is used for cases where these two functions are highly non-linear and the EKF gives poor performance.

4.3.2 Particle filter

The analysis of general dynamic models involves a sequence of posterior distributions corresponding to the subsequent stages of the dynamic model. In the absence of a Gaussian structure, numerical integration schemes are required [Mue92]. Particle filters (PF) are sequential Monte Carlo methods based on point mass (or *particle*) representations of probability densities, which can be applied to any state-space model and which generalise the traditional Kalman filtering methods [AMGC02]. They have been named differently in diverse fields. The statistics community, where they originated, knows them as *particle filters*. In the Artificial Intelligence community, they are called *survival of the fittest* and in computer vision, *ConDensAtion* [IB98].

The idea underneath particle filters is to partition the state space. Instead of a continuum, the space is discretised in order to have a tractable integration in the Bayesian statistics. Particles represent the Probability Density Function (PDF) of the state, and evolve following the state equation.

Their distribution is done according to the PDF so that the higher the probability, the higher the density of particles. By sampling the state space one gets a finite number of particles. The higher the number of particles, the more precise the approximated PDF, and consequently the state estimated.

Mathematically, this is represented as follows. The posterior density is approximated with a weighted sum of N_p discrete samples drawn from the posterior space

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \approx \sum_{n=1}^{N_p} w_k^n \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^n) \tag{4.30}$$

where n indexes the samples $\mathbf{x}_{0:k}^n$ and w_k^n are called *importance weights* and are normalised to sum 1. In general, it is impossible to sample from the true posterior distribution as it is not available. The idea is to draw samples from a known distribution. For this purpose a *proposal distribution*[*] $q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ is introduced. It is possible to approximate the posterior distribution by assuming that the weights can be defined up to proportionality

$$w_k^n \propto \frac{p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})}{q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})}. \tag{4.31}$$

If the importance density is chosen to factorise such that

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})\, q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \tag{4.32}$$

then one can obtain samples $\mathbf{x}_{0:k}^n \sim q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ by augmenting each of the existing samples $\mathbf{x}_{0:k-1}^n \sim q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})$ with the new state $\mathbf{x}_k^n \sim q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$ [AMGC02]. By derivation of Equation (4.16) using Equation (4.18), one finds that

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}, \tag{4.33}$$

A key factor in filtering is the ability to form a prediction from the previous state, recursively. Using the previous equation into Equation (4.31), one can find a sequential form of calculating the importance weights

$$\begin{aligned} w_k^n &\propto \frac{p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})}{q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})\, q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})} && (4.34) \\ &= w_{k-1}^n \frac{p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathbf{x}_{k-1})}{q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})}. && (4.35) \end{aligned}$$

Moreover, it is commonly assumed that the proposal distribution is only dependent on the the previous state x_{k-1} and the current measurement $\mathbf{z}_k$. In this case, one can discard the path $x_{0:k-2}$ and the history of observations $z_{0:k-1}$, leading to $q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$ and

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathbf{x}_{k-1})}{q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)}. \tag{4.36}$$

Finally, the sequential approximation with particles to the posterior density is

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{n=1}^{N_p} w_k^n \delta(\mathbf{x}_k - \mathbf{x}_k^n) \tag{4.37}$$

---

[*] Also known as importance density or important function.

The research in particle filters has produced a numerous quantity of variants to this generic idea. For instance, choices of proposal distribution as well as the analysis of the sampling used has brought many insights. Following are the algorithmic details of the PF used in this thesis, namely the Sampling Importance Resampling (SIR) filter, also called Bootstrap filter.

Sampling Importance Resampling (SIR) filter

A known problem of sequentially sampling the proposal distribution and re-weighting the particles is that the distribution of importance weights become increasingly concentrated. Hence, after some iterations, only a few particles have non-zero importance weight. This is known as the weight degeneracy problem. A possible solution is to generate many replicates of those particles with high weight and discard those with low weight. This step is called *resampling* and a graphical example is shown in Figure 4.5.

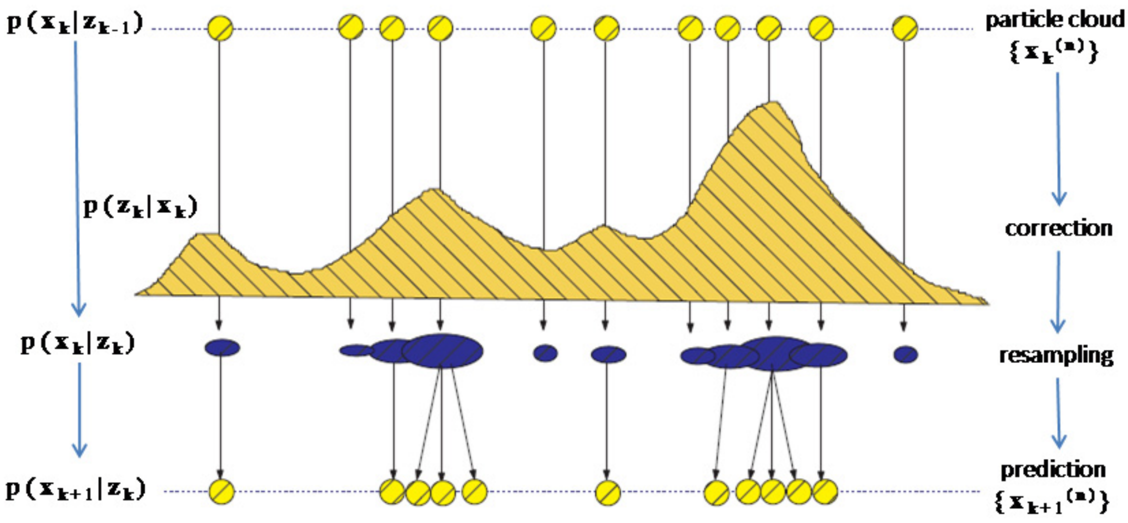

$p(\mathbf{x}_k|\mathbf{z}_{k-1})$
particle cloud
$\{\mathbf{x}_k^{(n)}\}$
$p(\mathbf{z}_k|\mathbf{x}_k)$
correction
$p(\mathbf{x}_k|\mathbf{z}_k)$
resampling
$p(\mathbf{x}_{k+1}|\mathbf{z}_k)$
prediction
$\{\mathbf{x}_{k+1}^{(n)}\}$

Figure 4.5: Graphic example of a generic Sampling Importance Resampling filter [Che03].

A common proposal distribution is the transition prior model $q(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{z}_k) = p(\mathbf{x}_k|\mathbf{x}_{k-1})$. With such distribution, the computation of the weights is straightforward since they only depend on the measurement model. Indeed, the importance weights are in this case

$$w_k^n \propto w_{k-1}^n \; p(\mathbf{z}_k|\mathbf{x}_k^n). \tag{4.38}$$

To conclude this section, Algorithm 4.1 specifies the steps that are needed for an iteration of a SIR particle filter using the transition prior model as proposal distribution.

4.4 Data fusion

Data or sensor fusion is the process of combining multiple inputs into a single output or decision. Fusion has been historically used for tracking purposes, especially linked to military applications

Algorithm 4.1 SIR filter with the transition prior as proposal distribution

```
Initialisation: for $n = 1,\dots,N_p$, sample $\mathbf{x}_0^n \sim p(\mathbf{x}_0)$, $w_k(\mathbf{x}_k^n) = \frac{1}{N_p}$.
loop
  1. Importance Sampling: for $n = 1,\dots,N_p$, draw samples $\hat{\mathbf{x}}_0^n \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^n)$ and set $\hat{\mathbf{x}}_{1:k}^n = \{\mathbf{x}_{1:k-1}^n, \hat{\mathbf{x}}_k^n\}$.
  2. Weight update: calculate the importance weights $w_k^n = p(\mathbf{z}_k|\mathbf{x}_k^n)$.
  3. Normalise the importance weights: $\tilde{w}_k^n = \frac{w_k^n}{\sum_{n=1}^{N_p} w_k^n}$.
  4. Resampling: for $n = 1,\dots,N_p$, sample an index $j(n)$ distributed according to the discrete distribution with $N_p$ elements satisfying $\Pr\{j(i) = l\} = \tilde{w}_k^{(l)}$ for $l = 1,\dots,N_p$. Then, for $n = 1,\dots,N_p$, $\mathbf{x}_{1:k}^n = \hat{\mathbf{x}}_{1:k}^{(j(n))}$ and $w_k^n = \frac{1}{N_p}$.
end loop
```

[HL01]. An overview of possible fusion architectures is given here in order to better contextualise the fusion framework proposed in this thesis. The following description of data fusion is applicable to any sort of signal or tracker.

As Thomas Bak stated in [Bak00], observational data may be combined, or fused, at a variety of levels. Three levels can be identified: high-level, mid-level and low-level fusion. Depending on the level of fusion, the data combination is done at an earlier or later stage of the tracking platform.

High-level fusion N trackers are designed to generate an estimate of the camera pose by processing the video frame with a different approach. A decision-module is used to choose the optimal tracked pose among the N available. The quality of the estimation of each of the trackers varies depending on the motion and references used. Therefore, the decision-module continuously computes and selects the best estimation.

The main problem with this kind of data combination is that there will be jumps in the final pose estimation. This happens each time the decision-module determines to change from the output of one tracker to another. The problem is due to the independency of fused trackers.

An example of such fusion scheme can be found in [OKS03].

Mid-level fusion N trackers are designed to generate an estimate of the camera pose by processing the video frame with a different approach. A fusion module is used to weight the contribution of each tracker. The weighting is done in accordance to a confidence value provided by each tracker together with the pose estimate. This fusion can be implemented, for example, with a bayesian filter that corrects its prediction using all the N tracker estimates. Each one of these measurements might have different measurement noise statistics that would weight their influence. In the case of KF, different measurement noise statistics are modeled with different covariances $\mathbf{R}_k$ for each tracker.

This solution overcomes the drawback of a high-level fusion. This means no jumps are produced because there is a continuous output.

Examples can be found in [Fox96, FN03].

Low-level fusion N cues provide different sorts of measurements, not necessarily in the state space of the final output. A fusion module is used to integrate all cues and generate a single estimate.

> This can be implemented, for instance, with a bayesian filter that has a unique process model but N different observation models (see Equation (4.14), p.53). In this case, each observation model would be tuned to a different cue.
>
> The framework proposed in this thesis is an example of a low-level fusion.

Another possible way of classifying data fusion is into loose or tight coupling. These terms are often used in computer architecture design. Loose coupling is referred to a design where the interdependence across modules or components is lessened. This way, changing one module has little or no impact on other modules. This approach is the most flexible. Tight coupling is on the other end, where each module relies on the other ones. The previous classification does not have a one-to-one mapping with loose/tight coupling. Indeed, trackers or cues can be independent (loose coupling) or dependent (tight coupling) but this relation might exist regardless of the level of fusion. Actually, the coupling depends strongly on the particular implementation.

4.5 Summary

This chapter has given an overview of the main background notions of the tracking framework, which will be presented further in Chapter 6. Firstly, tracking technologies as well as attributes that define the performance of a tracker have been discussed. Secondly, properties of the geometry of camera and its relation to world and image coordinates have been depicted. Thirdly, a conceptual description of Bayesian filtering with an emphasis on particle filters has been given. Finally, a classification of architectures for data fusion has been presented.

State of the art

5

The previous chapter has established the mathematical basis for camera tracking. In this chapter, a look at the state of the art in camera tracking using those mathematical tools is given. Among the vast literature on camera tracking, only those systems that use visual input are reported here.

This chapter is structured as follows. The conceptual approaches to the camera tracking problem are first introduced in Section 5.1. These approaches are then developed in Sections 5.2 and 5.3. The report of tracking techniques is complemented in Section 5.4. Fusions of approaches are described, as well as hybrids of video and other types of sensors. Section 5.5 gives a final comparison of methods to give the reader a better global perspective.

5.1 Introduction

In the last decade, there has been a growing interest in video-based camera tracking due to the increased accuracy, low economic cost of commercial cameras and higher CPU power. Indeed, accuracy is achieved thanks to a long history of visual geometry modelling research. Low-cost cameras such as WebCams are increasingly common in our daily life. Increased resolution and image quality makes them a feasible sensor for tracking. As opposed to other sensors seen in the previous chapter, most of the tracking is done through algorithms that interpret the imaged scene. The computational power is a key issue as more complex methods can be executed and in the last few years this has even reached real-time performance.

Video-based camera tracking relies on the registration of objects and/or mapping of environments. Among the possible classifications that can be made of the vast amount of available techniques, we have chosen to group camera tracking into two approaches: bottom-up and top-down approaches. For Bottom-Up Approaches (BUAs), the six DoF, 3D position and 3D orientation, are obtained from low-level 2D features and their 3D geometric relation (e.g., a plane, a square, or the edges of a model). BUAs rely on the accurate representation of the tracked target. For Top-Down Approaches (TDAs), the 6 DoF are obtained from top-down state space filtering techniques relying on a motion model of the target.

These two approaches are the origin of very broad research focusing on different scenarios and inputs, and hence dealing with specific issues. The goal of the following sections is to dive into these different techniques to better situate this thesis in the context of video-based camera tracking research.

5.2 Bottom-up approaches (BUA)

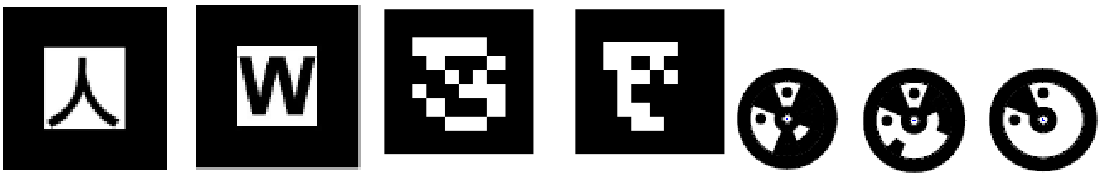
BUAs use geometric relations of certain detectable features in space to induce the pose of the camera. In this process, two main steps have to be fulfilled, namely tracking of references and camera extrinsic parameters estimation [OKS04]. The first step consists in detecting features such as interest points or edges in the image plane. If the three-dimensional relation of these points or edges is known, it is possible to estimate the camera pose using the two-dimensional coordinates of their back-projection onto the image plane.

Depending on the sort of features used, two different classes exist in BUAs: marker or fiducial-based tracking, and markerless tracking. Marker-based tracking uses specific patterns (called markers) with known geometry placed in the scene. This class introduced a leap for Augmented Reality (AR) applications. Augmented Reality consist in the overlay of virtual information such as 3D objects or text on top of a real video stream. Markerless tracking, on the other hand, uses no particular pattern but natural features whose geometric layout is either detected or fitted to a model. This second category deals logically with less prepared environments and attempts to remove the space limitations of marker-based tracking. In this section both classes are described.

5.2.1 Marker-based tracking

Fiducial markers such as squares or circles are easily detected thanks to their contrasted contours. Pose of the marker itself can be recovered from their geometric properties. Another advantage of fiducial markers is their coding capability. Indeed, pattern recognition algorithms can be applied once perspective distortion is recovered from imaged shapes. This enlarges the possibilities as each pattern can be associated with a different information, such as a virtual object to augment the scene or a localisation for a robot positioning system. Figure 5.1 shows various examples of fiducial markers. These are the patterns used in some of the tracking techniques explained hereafter.

The most extensively used marker-based tracker was presented by Kato and Billinghurst [KB99] as part of a framework for an Augmented Reality conference system. This tracker is now part of the ARToolKit [ART07] library. The tracking works as follows. Firstly, the image is binarised and regions similar to squares (surrounded by four lines) are selected. These regions are then normalised and matched to off-line registered patterns. To resolve the normalisation problem, the algorithm estimates the camera projection matrix P using the imaged position of the four corners and the four lines. Since this tracker is used in the framework proposed in this thesis, a detailed



(a) ARToolKit[KB99], 2 examples. (b) ARTag[Fia05], 2 examples. (c) Intersense[NF02], 3 examples.

Figure 5.1: Examples of square and circular fiducial markers.

description is given in Chapter 6.

Liu *et al.*[LSM$^+$03] employ colour square markers. This work targets architecture and urban planning using a round table. Since the collaborative work is over a plane (the table), homography-based pose estimation is used for each marker. A homography is the geometric transformation between two planes [HZ00]. In this case, the homography from the plane of the camera to the plane of the table is sufficient to estimate the camera pose. Additionally, this technique presents re-configurable markers based on colour segmentation.

Colour is also exploited by Neumann and Park [NP98] who present a framework to track coloured circular markers attached to an object. Camera tracking is achieved by examining all the possible estimates given trios of markers and selecting the most probable one. In addition, new markers can be added at run-time. To achieve this goal, the authors propose a recursive filtering approach to compute their 3D position (which is necessary for further pose estimation).

Naimark and Foxlin [NF02] describe the creation of coded circular markers and compare them to square ones. It presents the formers to be more robust to template matching especially for a large set of markers. The approach consists of two steps: acquisition and tracking. Acquisition deals with new detected markers, recognising their pattern and hence their code. Tracking uses only the centroid position of markers already recognised.

Marker-based tracking has proven to be an easy-to-use tool in AR applications, especially in indoor environments. Examples of collaborative and mobile applications as well as new gesture interfaces can be found in literature (e.g., [BK02, WS03, BVBC04]). Besides the flexibility and opportunities brought by markers, some drawbacks exist from a tracking point of view. For instance, in most of these techniques fiducial markers must always be in view in order to recognise their pose. The result being a direct failure with occlusions in cases where the marker partially disappears or when illumination conditions are poor. A consequence of this limitation is that the area of user's motion must be filled with markers in case loss of track is unacceptable for the application.

Beyond the representative examples described above, two recent works, [CF04] and [Fia05] stand out for their robustness to illumination changes and partial occlusions. Claus and Fitzgibbon [CF04] concentrate on the first problem taking advantage of machine learning techniques. In particular, a classifier is trained with a set of markers under different conditions of light and viewpoint. However, no particular attention is given to occlusion handling. Fiala [Fia05] uses spatial derivatives of grey-scale images to detect edges, produce line segments and further link them into squares.

This linking method allows the localisation of markers even when the illumination varies from one edge to the other. Although occlusion is partially handled, the edges must still be visible enough to produce straight lines that cross at the corners and hence enable marker detection.

5.2.2 Markerless tracking

Markerless tracking overcomes the space limitations of markers by using only natural features. Natural features are physical structures that are highly detectable from an optical point of view. This is the case for the corners of a table or the contour lines of a box. Their contrast with respect to the surroundings can be exploited for tracking. Since this type of features are present in a real scene without being artificially created, they are defined as natural features. By tracking the 3D position of these natural features it is possible to recover the pose of a camera.

Though all techniques in this area point to unprepared environments, geometrical constraints or a few references such as models or pre-calibrated views of a scene are necessary for registration. Techniques can be classified based on the constraints taken from the environment, namely, planar structures, models and geometrically unrestricted scenes. The remainder of this section gives some examples of these classes.

Planar structures

Several techniques estimate the camera pose using 3D *planar structures* in the real scene [NY99, SFZ00, PXC02, JD02a]. In some cases they are highly accurate and almost jitter-free. As a counterpart, they often require greater computing power as compared to marker-based approaches. This makes them unsuitable for real-time applications.

The algorithm presented by Neumann and You [NY99] detects natural features such as points and regions which group co-planar areas in the scene. Camera motion is then derived from 2D optical flow tracking of the features. It consists in minimising a least-square error function of image transformations from frame to frame. However, this motion is limited to an affine model and does not estimate the full perspective determined by the 6 DoF.

As described previously, a homography maps two different views of the same real 3D plane [HZ00]. Its computation uses directly the 2D point correspondence and the only restriction is that points lie in the same 3D plane. Several works have identified the advantages of homography computation as a mean to obtain camera pose from feature points (e.g., [SFZ00, PXC02, JD02a]).

Simon *et al.*[SFZ00] assume that a planar region is always visible, which is generally the case for indoor environment (e.g., ceiling or floor are commonly visible). The pose estimation starts with an initial detection of a plane either selected manually or automatically (after at least two frames). Feature points in the plane are tracked and the homography between two consecutive frames is computed from point correspondences. In order to avoid outliers (due to moving objects or incorrect point correspondences), the RANSAC robust estimator [FB81] is used. As it can be seen, camera motion is accumulated along time from the initial plane selection. The drawback of this method is the drift introduced by errors in the computation of the homography, which are propagated from frame to frame. Additionally, the authors propose a "hand-off" method to switch between planes so that tracking is not interrupted when the initial plane is no longer visible. This work was extended to multiple plane tracking in [SB02] (see Figure 5.2).

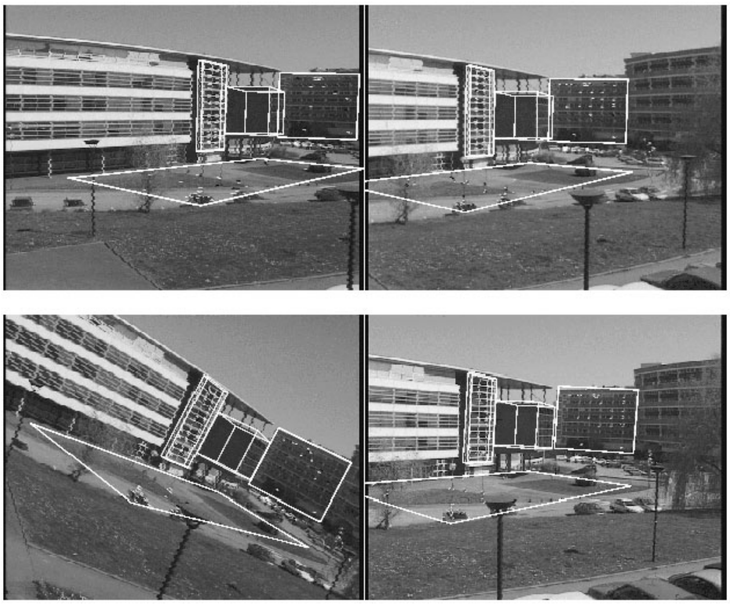


Figure 5.2: Tracking of multiple planes using feature points [SB02]. Planes' correspondence can be identified among the four camera poses.

Prince *et al.*[PXC02] propose a similar approach but compute the absolute camera pose at each frame. To achieve this, a picture of the surface to track is pre-registered as a reference (see Figure 5.3). The picture is equivalent to a marker in the scene. The difference is that tracking is performed from natural feature points on a frame-to-frame basis.

Jurie and Dhome [JD02a] use the optical flow approach in the following manner. Pose is estimated by minimising a least-square function that parameterises the homographic warping between a reference image and the image at the current frame. In order to compute this minimisation robustly, the authors propose a hyperplane approximation of the warping function assuming that the motion is small. However, this computation has a high computational cost. This problem is addressed by learning the hyperplane approximation in an off-line stage. This parameterised homography determines the pose of the plane. Real-time performance with robustness to illumination changes and occlusion has been shown in [JD02b] (see Figure 5.4).

Model-based

Another solution to register the scene is to use a rough Computer-Aided Design (CAD) model of parts of the real environment, generally an object, and to fit this model to edges or points detected in the image. This is known as *model-based tracking*. In this case, camera pose computation is relative to the center of mass of the object. Therefore, it is sometimes also referred to as 3D object tracking.

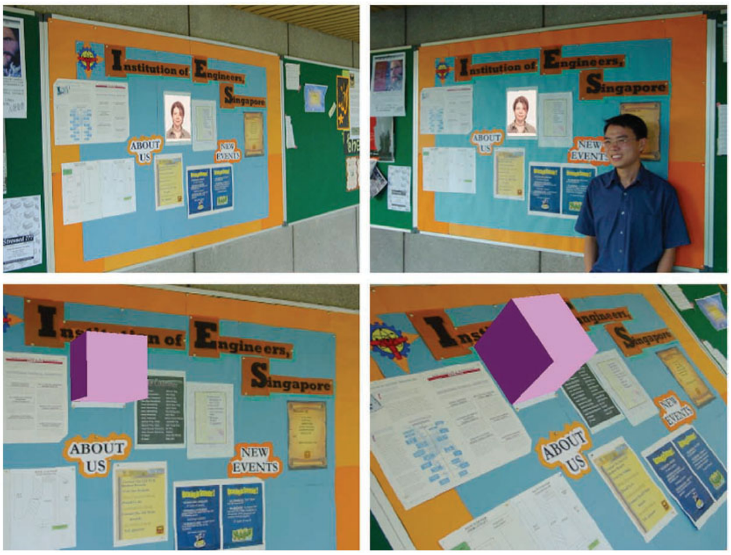


Figure 5.3: Pose computation from a reference plane image (taken from the real clipboard) and feature point frame-to-frame tracking [PXC02]. In the bottom row, the scene is augmented with a virtual 3D cube.

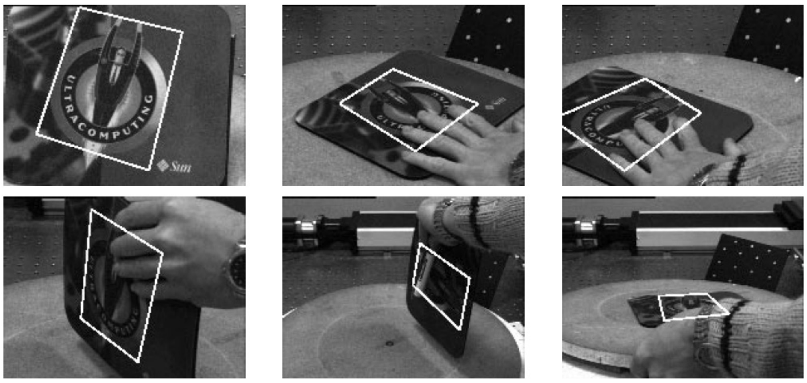


Figure 5.4: Homography parameterisation of image transformation for plane tracking [JD02b].

The *edges* of an object are often easily detectable in gradient space. This makes them a suitable reference for tracking. The model is fitted to the detected edges by minimising the reprojection distance between both. Depending on the processing of gradient information two different categories can be considered.

In a first category, the idea is to look for strong gradients in the image (which are possible candidates of edges of the object) that are close to the current pose estimate (e.g.,[Har92, SLB99, DC02, VLF04a]). This method, first introduced by Harris [Har92], consists in establishing a set of 3D points lying on the edges of an object (called control points) and match them with points with high gradient magnitude. More concretely, control points are searched at the perpendicular direction of the edges according to the pose estimated in the previous frame. Once control points are matched, the new object pose is estimated (see Figure 5.5).

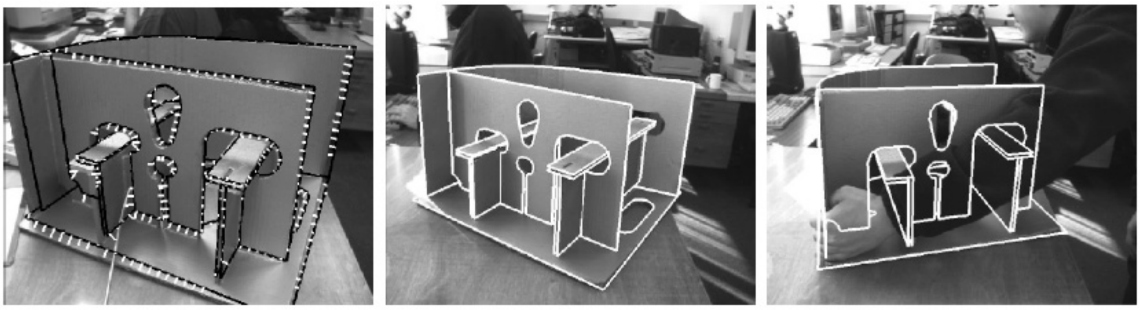


Figure 5.5: Model-based tracking using control points lying on object's edges [DC02].

In a second category, the idea is to first detect primitives such as line segments and to fit them to the model's contour. Lowe [Low92] uses this approach. For this purpose, a Bayesian framework is employed to find line segments (see Figure 5.6). The goal of this computation is to reduce the search areas (gain efficiency) of potentially correct image line segments and deal with multiple candidates. In this work, distance and orientation of detected line segments are compared to those of the model.

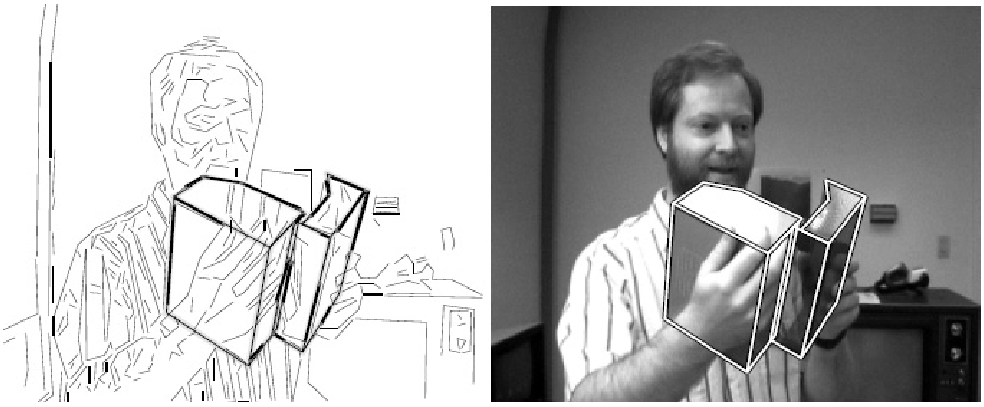


Figure 5.6: Model-based tracking using line segment detection [Low92].

A problem of the two categories described just above lies in the minimisation process. Two situations make the process fall in local minima: when the edges of the object are confused with the background; or different edges are confused due to the object's shape. Drummond and Cipolla [DC02] address the first situation of detecting multiple edges within the search area. In their work, the influence of the control points is inversely proportional to the number of edge strength maxima visible within the search path. Vacchetti *et al.*[VLF04a] addresses both problems with a robust estimator to handle multiple pose hypotheses.

Another cue used for object tracking is the position of *interest points* or *patches* lying on the object's textured surface (e.g., [UK95, RDLW95]). Uenohara and Kanade [UK95] track patches of the object's surface and use their texture to recognise the location in the image. A reference of each patch is kept for further matching. To compensate for viewpoint changes, those patches are warped according to object's pose. Ravela *et al.*[RDLW95] propose to match patches using 2D rotation-invariant cross-correlation. A large set of object's surface patches is stored and later accessed depending on the estimated pose. A table stores the subset of patches that is visible at each pose and the tracking system uses this set to match the model. A KF is used to smooth the output.

In real sequences, the right edges and feature points are not always easy to detect alone. For instance, when the scene has cluttered background many strong edges and feature points may confuse the tracker. On the other hand, when the object is poorly textured, edges might be visible but feature point-based methods would probably fail. Nevertheless, it is possible to combine both cues to produce a more robust tracking algorithm (e.g., [VLF04a, SLB99]). For instance, Simon *et al.*[SLB99] combine the model fitting of points, lines and curves, with the epipolar constraint between consecutive frames. Point correspondences of two images are related by the Fundamental matrix through the epipolar constraint [HZ00]. The idea is to locate the back-projection on the image plane of the same 3D feature point for various frames of a video sequence. From the relation of their different 2D back-projection, it is then possible to compute the motion of the camera. In [SLB99], feature points are tracked from frame to frame and the Fundamental matrix is computed from the set of pair-wise correspondences. In this way, the model-based pose estimation is enforced with feature points beyond those lying on the object's surface. However, a drawback of computing the Fundamental matrix between consecutive frames is that the camera centre must translate and that the computation can be unstable.

Another complementary feature used for model-based tracking is based on *keyframes*. Keyframes are frames for which the viewpoint is known. Instead of minimising a reprojection error, object pose can be estimated by referring to those known viewpoints. The epipolar constraint is enforced, in this case, between wide-angle views of the object, by matching feature points lying on the same object. Using keyframes avoids the drift problem as the camera pose is always computed as an absolute transformation. Moreover, the computation of the Fundamental matrix does not encounter the unstable situations cited before because the viewpoint of the keyframes is usually different enough from that of the current frame. Examples of this category are [LVTF03, VLF04b]. Vacchetti *et al.*[VLF04b] propose to extend this idea by generating new viewpoints during tracking to deal with large scale changes. Nevertheless, a problem arises when new viewpoints are created while the object is occluded.

Geometrically unrestricted scenes

In the two precedent cases, geometric constrains have been used to determine camera pose. Fiducial markers, planes, and object models are used with impressive results for tracking. However, in some scenarios or applications, none of them are available. For instance, in outdoor environments it is difficult to prepare the scene by placing markers or defining a model (even rough) of the scene. Even in indoor conditions it is not always possible to prepare the environment beforehand.

Several works have exploited the *epipolar constraint* to deal with scenarios where geometric restrictions cannot be imposed. This approach to camera registration has been studied in the last decade with excellent results [TK92, PK97, PKV99, HZ00]. Most of the ideas of these seminal works are now part of commercial products such as Boujou®[Bou07] for post-production movie edition. However, it is only recently that this approach has been applied to real-time camera tracking systems [CCP02, NNB04, KKSES05].

Chia *et al.*[CCP02] propose a system that tracks feature points and computes the epipolar constraint between the current frame and keyframes. Actually, it uses one or two off-line calibrated keyframes. The calibration consists in computing the transformation from the camera to the world coordinates using the fiducial marker-based system of [KB99]. Tracking is then performed by estimating feature point localisation from consecutive frames and computing the Fundamental matrix between the current and one or two reference images. Therefore, camera tracking is assisted in this case by the two and three-view constraints [HZ00], depending on the number of key images used. Using keyframes in this case has the same benefits as those achieved for model-based tracking. An additional benefit and extension of the system of Chia *et al.*is that by using a database of reference images it is possible to broaden the tracking area. By exploiting the current camera position it would be possible to know which reference image(s) can be used. As identified by the authors, using epipolar constraint based on feature point matching at each frame introduces jitter in the pose estimation. A KF is used to smooth the jittered output.

Nistér *et al.*[NNB04] present a framework based on the Essential matrix computation (similar to the Fundamental matrix but with calibrated cameras) [HZ00]. This framework combines the epipolar constraint between consecutive and non-consecutive frames with perspective methods based on environment 3D mapping. The system works as follows. Feature points are tracked (using a fixed search region and Normalised Cross Correlation (NCC)). For several frames, the track of features is kept. The process starts with the estimation of the relative poses between three frames using point correspondence. With the camera pose and the 2D point tracks it is possible to estimate the 3D position of these points. The process can then continue by using the 3D points and the 3-point algorithm [HLON94]. In order to avoid drift, the system is forced to recompute, from time to time, the 3D position of feature points, according to more distant frames. All the geometric computations use the RANSAC [FB81] robust estimator to discard outliers. This framework has been tested on a real path of 600 metres with a very small drift.

A great advance in markerless tracking, however, has come more recently with the introduction of *tracking-by-detection* techniques [SL04, LLF05]. In the techniques discussed before pose estimates are accumulated along time or points are tracked at consecutive frames. As opposed to this, in tracking-by-detection the idea is to compute pose and point correspondence at each frame of a video stream, regardless of the previous estimate. More precisely, points are detected in a reference frame and their correspondence with points detected at each frame is computed with independence of previous correspondences. Since the pose of the tracked scene with respect to the camera can be arbitrary, the point correspondence is in this case a problem of *wide baseline matching*. The generic problem of wide baseline matching and related research is detailed in Chapter 2. Here, only research related to wide baseline matching for camera tracking is presented.

Skrypnyk and Lowe [SL04] present a fully automated framework for object modelling and tracking. Point correspondence is achieved through a robust process of feature point detection and matching using the Scale-Invariant Feature Transform (SIFT) presented in [Low04]. Pose is computed as a global optimisation on camera parameters (calibration, rotation and translation) and 3D location of feature points. This optimisation process is dynamically regularised in order to reduce jitter when motion is small but avoiding to lag behind when motion is fast. Although very accurate tracking is produced, the drawback of this method is the processing time needed to generate and match SIFT point descriptors.

Lepetit *et al.* [LLF05] break this bottleneck with a fast strategy for feature point description and matching based on machine learning techniques. The idea is to load the computational cost on the training stage and not at runtime. The authors propose to train a classifier where each class contains multiple views of the patch that describes a feature point. Generating different views can be achieved by image warping. Thanks to an efficient implementation of the classification engine using only pixel intensity differences, the tracking achieves real-time performance.

Notice that tracking-by-detection is similar in spirit to marker-based trackers. In both cases, pose is estimated at each frame. More precisely, a reference is calibrated off-line and then searched in the current frame. The advantage of using feature points over markers is that occlusions can be handled seamlessly. Indeed, only a subset of all the feature points is needed to estimate the pose, whereas the complete marker has to be detectable in order to produce and estimate. The price that has to be paid is more complex algorithms either at run-time (e.g.,[SL04]) or off-line (e.g.,[LLF05]).

5.3 Top-down approaches (TDA)

As discussed in the previous section, BUAs rely on geometrically known features in the scene, such as markers and objects. Moreover, pose estimation is done by accumulating inter-frame motion estimates or using keyframes. As opposed to BUA, Top-Down Approach (TDA) rely on the context and induce the geometry of the scene from this context. More concretely, motion models are used in bayesian filters to predict the pose of the camera and, from this prediction, references in the scene are sought. With the detection of these references, the prediction is corrected and additionally, the geometry of the environment can be deduced. Indeed, the 3D position of references such as edges or points and their 2D back-projection in the image plane are related by geometric constraints in the camera pose (see Equation (4.5), p.49). Such constraints induce the necessary correction of the predicted pose. Therefore, the camera is tracked along time by performing a prediction/correction cycle at each frame (see Section 4.3, p.53. At the same time, references have to be added online to enable extended tracking (this is generally known as *mapping*).

In the prediction/correction cycle two different issues have to be addressed, namely, filtering and data association. Filtering is related to the motion model used and the limitations of assuming such model. Data association deals with the localisation of the reference(s) according to the predicted pose, for further pose correction. The remainder of this section describes how researchers have dealt with these two aspects of top-down approaches.

5.3.1 Filtering

Recalling from Section 4.3, the state of the filter is updated with incoming measurements. The state of the filter **x** in bayesian tracking is a representation of the pose of the camera. In other words, the state expresses the transformation **T** between the camera's reference system and that of the world. This is equivalent to $[\mathbf{R}|\mathbf{t}]$ (see Section 4.2, p.47). Generically, this state vector can be formulated as

$$\mathbf{T} = [t_X, t_Y, t_Z, \phi, \theta, \psi], \tag{5.1}$$

where t are the translations and ϕ, θ, ψ are the angles of rotation in respect of the X,Y and Z axes. The possible representations of these angles have been treated previously in Section 4.2.2. In some cases, the translational or angular velocity are also added to the state vector.

Bayesian tracking can be coarsely classified in two categories depending on the type of filter used. On the one hand there would be those camera trackers that assume Gaussian motion models and exploit the benefits of the well-known KF and its variants [SSC87, KKR$^+$97, Dav03, MDR04, YWC04, CPMCC06]. On the other hand, some scenarios are badly modelled with Gaussian noise and hence other solutions are needed. In this case, the common approach is to use PF. This approach is generally known as Monte Carlo Localisation [DBFT99, SG99, MTKW02, QC04, PC05, SEGL05, PC06a, PC06b]. Another category can be found in the robotics literature, namely grid-based Markov localisation (e.g.[BFHS96][Mur99]). In this case, only an interesting part of the state space is discretised and used for localisation purposes. The drawbacks of this method are the a priori commitment to a limited state space and the precision of its grid. This category is not directly related to this thesis and hence no further details are given here.

Most of the original research on TDA comes from the robotics community. The first framework to combine KF navigation and feature uncertainty mapping was presented by Smith *et al.*[SSC87]. The authors proposed a KF-based framework where features' location are modeled with their mean and covariance. The idea that underlies this framework is that the observations of features as the camera moves are highly correlated in space. In other words, that although the exact location of a single feature in world coordinates is difficult to determine, the spatial relation between features can be deduced accurately from camera motion. This idea has been the basis for research in Simultaneous Localisation and Mapping (SLAM) since this seminal work.

In some cases, camera motion is difficult to model with the Gaussianity assumption of the KF. This is the case of erratic motion. In addition, such filters cannot deal with multi-modal posterior densities that arise when multiple hypothesis are found. In such situations, PF become an interesting alternative. Dellaert *et al.*[DBFT99] defined what is called Monte Carlo Localisation (MCL) method based on this alternative. In this case, a robot is positioned using the 2D image of the ceiling. Montemerlo *et al.*[MTKW02] tried to combine the advantages of KF for mapping with those of PF for localisation. More specifically, the problem of simultaneously estimating trajectory and building a map is decoupled (factorised) into two separate estimation procedures. The idea is to reduce the sample space (that usually contains both motion and feature locations) by applying Rao-Blackwellisation according to the relation Pr{motion,map} = Pr{map|motion}·Pr{motion}. If

Pr{map|motion} can be computed analytically, only Pr{motion} needs to be sampled. The sampling is hence only needed for the positioning, and this is done similarly to MCL. The analytical part is computed with a KF for each feature given the state of each particle.

As said before, most of this seminal work is oriented to robot navigation, where motion is mechanically controlled and often over a 2D plane. In such environments, odometry provides an additional input for tracking correction. In the following, we concentrate exclusively on research about purely vision-based TDA derived from these seminal works on robot SLAM.

Among the KF-based approaches, Koller *et al.*[KKR$^+$97] propose a tracking framework where the filter is updated with the 2D positions of the corners of calibrated markers (known 3D position) placed in the scene. Although this could be viewed as marker-based tracking, corners are located from a TDA using the filtered pose estimate. Nevertheless, the benefits of this particular KF+markers framework have been surpassed by more recent research on marker-based tracking.

Davison [Dav03] presented a real-time system based on an EKF updated with feature point locations. As a natural evolution from the previous cited work, this framework is much more flexible and proves good accuracy results in indoor environments. Starting from a calibrated position, the system tracks feature points in 2D according to the estimated 3D pose. The 3D position of the feature points in the world coordinate frame restrains the camera pose in the update step. An important contribution of this work is the online addition of new 3D feature points present in the scene, to the set of feature points used for camera tracking.

During the development of this thesis, Monte Carlo techniques have been applied to real-time 3D camera tracking, probably thanks to higher available computational power. For instance, Pupilli and Calway [PC05] propose a framework with strong similarities to that of Davison [Dav03] but using a PF (see Figure 5.7). In [PC06a] the same authors use the junctions of line segments detected on a known 3D model instead of feature points in the scene. In [PC06b] they recover the original ideas of the Rao-Blackwellisation of Montemerlo *et al.*[MTKW02] but propose a single Unscented Kalman filter (UKF) for each feature instead of attaching a bank of KF to each particle. In this way, real-time performance is achieved. However, contrary to [MTKW02], this approach cannot deal robustly with wide area environments.

5.3.2 Data association

For correct filter update, it is necessary that data observed and re-observed is treated accurately, keeping geometrical relations. Indeed, features must be identified at different frames and associated when they represent the same data. This is commonly known as *data association*.

Any data association algorithm has to deal with multiple measurements that are candidates to match a known feature. This problem has been addressed in literature for domains beyond camera tracking (e.g., [VRB01, CH96]). Veenman *et al.*[VRB01] formulated the problem of data association as an assignment problem. They proposed to use the Hungarian algorithm [Kuh55]. Cox and Hingorani [CH96] proposed an efficient implementation of the original Multiple Hypothesis Tracking (MHT) filter [Rei79] for the tracking of feature points. These algorithms have a high

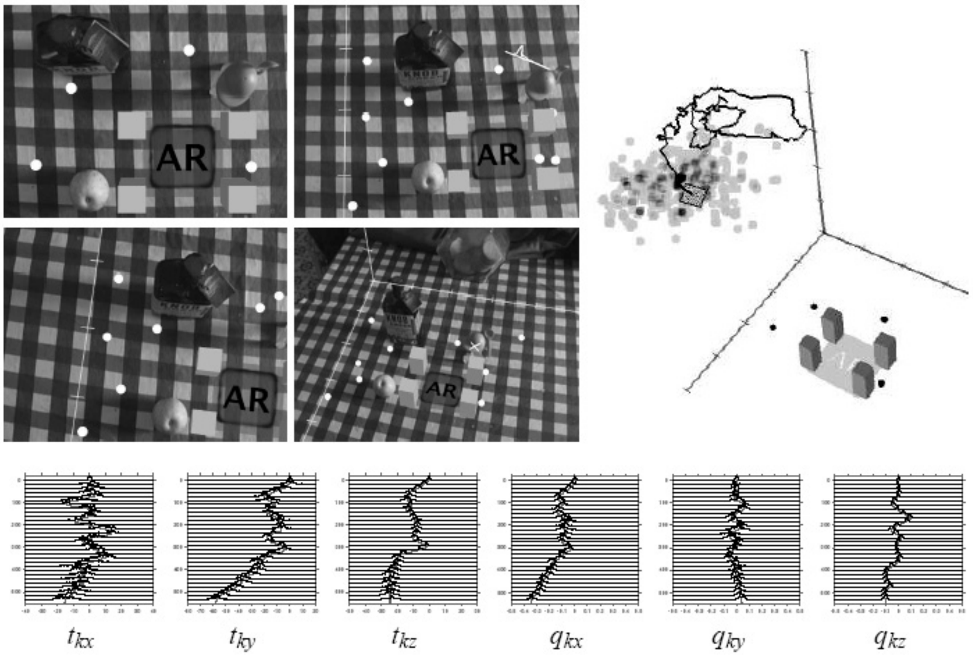


Figure 5.7: Particle filter-based camera tracking with feature points [PC05]. Top-left: example frames with projection of tracked feature points (white dots). Top-right: mean camera trajectory and particle cloud. Bottom: projections of the posterior density for each motion parameter (translation and rotation axis of the quaternion).

computational cost and are not often used in camera tracking. The interested reader can find more details and references on data association in [BSB00, YJS06]. Although data association is also a problem for BUA, this section concentrates on the solutions proposed in TDA.

Visual data association can be divided in two steps: gating and validation. *Gating* is the process of estimating a region in the measurement space in which the probability of match is high. *Validation* is a process that follows gating and consists in determining which measurement, among the ones inside the region, has the most similar visual properties when compared to the visual descriptor of the sought feature.

The gating process relies on the fact that in Top-Down Approach (TDA) the camera pose is known. This knowledge, together with an available map of 3D features, constrains the region in the image plane where the back-projection of each feature may lie. Most filter-based systems also know the uncertainty on camera pose and hence can project this uncertainty in image plane coordinates. This uncertainty determines a 2D space called *gating region*. The gating process has advantages and disadvantages. On the one hand, the computational cost of the search for measurements is alleviated. On the other hand, it can also be a source of errors. Indeed, if camera pose is incorrectly estimated, the gating region might not include the correct measurement. If the feature is then associated to a wrong measurement, the filter will be updated wrongly. If this process is repeated, the error is propagated and camera estimates drift. This problem is often solved by setting a large gating region that allows computation efficiency but at the same time keeps the probability of match high.

As said before, the validation process relies on visual properties of the features to associate. For instance, the solution for feature point-based trackers is often to use the image patch around the feature point as descriptor. Then, the validation consists in computing the NCC. More precisely, the template is correlated with the gating region, or the template is correlated with a template extracted from the neighbourhood of specific feature points detected inside the gating region. Examples of this solution are [Dav03][PC05]. Nevertheless, a problem arises when the viewpoint at which the descriptor was extracted and the current viewpoint of the same feature is different. Although a degree of rotation and scaling is accepted [DM02], beyond certain angles no valid match is found and hence no possible camera constrain for that feature are available. A straight solution to this problem would be to update the template of the feature after a positive validation (done in a previous frame). However, a problem could arise easily if poor positive validations are concatenated and the descriptor is no longer related to the original feature. In that case, the camera estimates would also consequently drift.

Several works have addressed this problem in a more elaborated way [MDR04, SEGL05, CPMCC06]. Molton *et al.*[MDR04] present a solution to overcome the problem of the correlation validation for locally planar surfaces. The normal of planar surfaces is detected with gradient-based image alignment. By warping the template according to the current state of the filter and the normal of the surface, it is possible to get a version that is closer to the appearance of the region in the current frame. In this way, correct matches are obtained even when viewpoint changes more drastically. This approach has however several disadvantages. Firstly, features might be detected at edges (not locally planar) and the normal to an edge changes rapidly inducing unstable tracking. Secondly, when an occlusion starts, the estimation of the normal changes to compensate and this again induces unstable tracking.

In the last two years, the application of invariant descriptors for correct validation has brought important improvements to camera tracking [SEGL05, CPMCC06]. Sim *et al.*[SEGL05] use SIFT features [Low04], which have high scale and rotation invariance enabling accurate tracking. However, the extraction and description of SIFT features makes the mapping more complicated. Indeed, the gating process cannot be used in a straightforward manner because the descriptors are scale invariant and hence the features have many different scales. Therefore, the association is done by traversing all the list of feature descriptors. This process has a large computational cost and the overall system runs at 11.9 seconds per frame. The authors propose an alternative kd-tree for association. Although this alternative is more efficient, the authors identify a degradation in data association performance. Chekhlov *et al.*[CPMCC06] propose a multi-resolution descriptor based also on SIFT. The approach differs from [SEGL05] in that the extraction of feature points is done at a fixed scale. In order to be scale invariant, several SIFT descriptors at different scales are stored for each feature. At runtime, the scale is selected according to camera pose and 3D feature position. Once the scale is selected, the validation can be computed.

Another problem that must be addressed is the one that arises when dealing with dynamic scenes. Most of the mentioned works cope with rigid environments. This meaning that mapping is done on static features. However, if a large number of feature points is detected on a moving object, a system would have difficulties in knowing whether it is the camera that moves or it is the object. For a small number of features lying on a moving object, top-down approaches should be able to handle this problem. Indeed, if the motion of a feature is uncorrelated with the motion of the camera, the former can be detected as an outlier and, possibly, as part of a moving object. Fitzgibbon and Zisserman [FZ00] report a bottom-up framework based on epipolar geometry to estimate structure and motion with multiple moving objects. A top-down approach to address this problem is presented by Wolf and Sukhatme [WS04]. In their work two maps are updated at the same time. On the one hand, a dynamic map keeps track of the moving parts of the scene. On the other hand, a static map defines the position of static features.

5.4 Hybrid tracking

None of the tracking techniques discussed above represent the proverbial *silver bullet* [WF02]. However, their limitations or weaknesses are complementary in some ways. Synergies have proven to be necessary for some applications. These synergies are referred to as *hybrid systems*. In this section, hybrid tracking is treated at two levels. Firstly, we describe fusions of sensors, where one of the sensors is visual. Secondly, fusions of video-based trackers are described as these are closely related to the framework presented in this thesis.

Video and inertial tracking have complementary technologies [ABW01]. As described before in Chapter 4, high performance of inertial sensors is achieved for fast motion. On the other hand, in order to compensate for drift, an accurate tracker is needed for periodical correction. Trackers that compensate for this poor performance at low motion are the best candidates for a fusion. This is the case of video-based tracking which performs better at low motion and usually fails with rapid movements. In addition, video-based tracking is suitable for both indoor and outdoor mobile applications. On the other hand, other hybrid solutions such as inertial-acoustic tracking [FHP98] still have the environment preparation constraint. During the last few years, robust trackers have been implemented combining video and inertial. As presented in one of the first such combinations by You *et al.*[YNA99], the computational cost of markerless tracking can be decreased by 3DOF orientation self-tracking of gyroscopes. It reduces the search area and avoids tracking interruptions due to occlusion. Several research on this combination has followed this idea [KFTY00, FN03, KD03].

Kanbara *et al.*[KOTY00] presented a marker-based tracker. Three markers served as reference for pose estimation with a stereoscopic camera using the epipolar constraint. In addition, this system correctly rendered mutual occlusion between real and virtual objects. Virtual objects were segmented by real objects in front. However, this system has the limitations described in Section 5.2.1 (i.e. limited number of markers, and markers must be visible). This led to a hybrid solution from the same group of researchers [KFTY00], combining inertial- and marker-based tracking, which enables the algorithm to estimate the position of markers (especially when those were out of scope).

Foxlin and Naimark [FN03] developed a framework to deal with different tracking sensors in a defined timing scheme. This framework performs auto-calibration, mapping and tracking. The first sets up the system's accuracy. The second initialises the system coordinates in space. The user walks around the scene and the system maps the markers' positions. The tracking uses Kalman filtering algorithms to merge both sensors.

Klein and Drummond [KD03] fuse model-based tracking with inertial tracking. In this case, fast sampling rate of the inertial sensors is exploited to compensate blurring images during rapid motion.

A different sensor was fused with video by Agrawal and Chellappa [AC05]. In this work, a depth range finder is used together with a camera in a Rao-Blackwellised framework. Structure is obtained with the range finder while motion is estimated with a particle filter.

Besides the hybrid trackers that fuse different sensors, little attention has been given to fusing diverse techniques from the same modality in camera tracking. Nonetheless, several researchers have identified the potential of video-based tracking fusion [OKS03, SUYT03, NNK04, KD04].

Satoh *et al.*[SUYT03] use two cameras, one stationary and one mobile. The mobile camera has a marker attached to it and performs model-based tracking. The stationary camera (called bird's-eye view camera) tracks the marker of the mobile camera to reduce the uncertainty of its pose and consequently improve the model-based tracking. Nevertheless, they do not exploit this combination in the initialisation phase, which is obtained manually by situating the camera close to a predefined position. Najafi *et al.*[NNK04] presented a similar system but in this case the fusion enables automatic initialisation of the whole system (see Figure 5.8 for a schema of the fusion). During the same period, an independent work by Klein and Drummond [KD04] was published. It shows a similar fusion schema where Infra-red LEDs attached to a Tablet PC were tracked by a stationary camera. In parallel, their own model-based tracking [DC02] (described previously in Section 5.2.2) is used from a camera attached also to the Tablet PC.

Okuma *et al.*[OKS03], as opposed to those works, are the only ones to fuse motion data from a single camera. Their system combines a model-based tracker and a PF tracker. The former is based on 2D feature point tracking and the 3-point algorithm[HLON94] with known 3D points of the model. The latter is very similar to the framework of Pupilli and Calway [PC05]. The fusion is done at a high-level (see Section 4.4, p.57) by switching between the BUA and the TDA according to an error measure associated to the former. However, this combination takes limited advantage of the filtering framework and still needs the assistance of an inertial sensor.

5.5 Comparison of approaches

In order to give a clearer picture of the vast amount of camera tracking technologies developed in the last 20 years, this section compares them. In particular, the needs and performance of these technologies are summarised. This is done in accordance with the tracker properties previously described in Section 4.1.1. Furthermore, the capabilities provided by each group of techniques are also discussed.

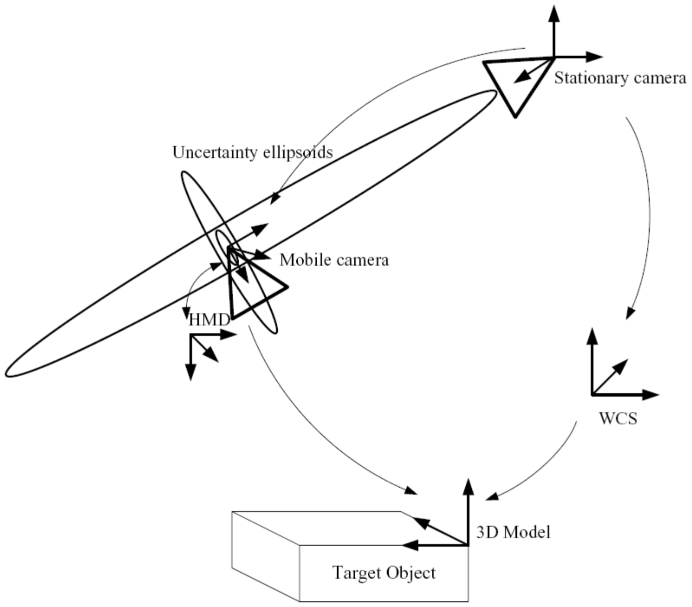


Figure 5.8: Fusion of marker-based tracking on stationary camera and model-based tracking on mobile camera [NNK04].

The most important restriction that a tracker has is the type of references that are needed and the amount of preparation that derives from these references. Table 5.1 states the information that is needed beforehand and the type of image primitives that must be visually available. In some cases, details are given for specific references. These restrictions or needs determine the applicability of each system to a different scenario. For instance, in outdoor environments it is difficult to provide the system with a reliable model or to place markers all around. On the other hand, planes are easily

Technique	Constrain	Prior information
Bottom-Up Approach		
• Marker-based	Marker	Pattern (one sample).
• Markerless		
- Planar structures	Textured plane	None or image of plane[PXC02, JD02a].
- Model-based	Edged or textured object	Model and keyframes[LVTF03, VLF04b].
- Epipolar-based	Textured scene	None or keyframes[CCP02].
- Tracking-by-detection	Textured object	Classification or description of feature points in original image.
Top-Down Approach	Textured scene	None

Table 5.1: Necessary information for different tracking approaches.

detectable in such open spaces. Nevertheless, most of the needs are fulfilled in indoor environments as texture is often available and rough CAD models can be generated somehow easily. It must be pointed out that the final objective of any of these technologies is to provide the best performance with the minimal a priori information. This is possibly the reason why marker-based approaches have such success in user interfacing applications (e.g., AR).

Measured from a less application-oriented and more technical point of view, each one of these trackers has to deal with its own limitations. Table 5.2 summarises the main failure modes in terms of accuracy, type of motion, and response to an occlusion of the reference. Update rate has not been added to this table because we could fall into an unfair comparison. Indeed, implementations are not necessarily optimised in research works. Moreover, computing power is constantly increasing so some of the seminal works proposed a decade ago could now be running on real time.

Technique	Accuracy	Motion	Occlusion
Bottom-Up Approach			
• Marker-based	Small jitter		Fail. [Fia05] handles partial occlusions.
• Markerless			
- Planar structures	Drift	Fast	[JD02a] fails
- Model-based	Drift, jitter	Fast	
- Epipolar-based	Drift without keyframes	Fast	
- Tracking-by-detection	Jitter		
Top-Down Approach			
• KF-based	Drift	Fast, Erratic	
• PF-based	Drift, Jitter	Fast	

Table 5.2: Performance and limitations of different tracking approaches.

The goal of a generic camera tracking algorithm is to deal at the same time with the following situations:

Automatic pose initialisation The tracker can start at any pose automatically.

Automatic re-initialisation after loss of track The tracker detects a loss of track, either accumulated by drift or because the reference is temporarily unavailable, and is capable of resetting its pose once the reference is available again.

Partial or complete occlusion of the reference known a priori When the reference is occluded, tracking continues by relying on references detected at run-time.

These three main capabilities are compared for each group of techniques in Table 5.3.

The comparison of approaches in terms of robustness and capabilities shows that no particular solution appears as outperforming. However, it is possible to identify compensated weaknesses and

Technique	Initialisation	Re-initialisation after loss of track	Tracking beyond known references
Bottom-Up Approach			
• Marker-based	Automatic	Automatic	No
• Markerless			
- Planar structures	Manual	No	Only if other planes can be found.
- Model-based	Manual/Automatic	Manual/Automatic	No
- Epipolar-based	Manual	No	Yes
- Tracking-by-detection	Automatic	Automatic	No
Top-Down Approach	Manual	No	Yes

Table 5.3: Capabilities of different tracking approaches.

strengths. For instance, while TDAs resist occlusions and need no prior information, none of them provides drift-free tracking. On the other hand, drift can be eliminated by using detection at each frame (marker-based or tracking-by-detection), although, in this case, a known reference must be provided beforehand. Similar identifications are possible in terms of capabilities. Some examples that benefit from these fusions have been discussed in the previous section about hybrid tracking.

5.6 Summary

This chapter has given a detailed overview of the state of the art in video-based camera tracking. The first section has dealt with those techniques that use features and deduce the pose of the camera by detecting their location in the image, called Bottom-Up Approaches (BUAs). Following this description, Top-Down Approaches (TDAs) have been described. These techniques use the temporal context and therefore have been explained from a filtering and data association point of view. Finally, hybrid techniques with an emphasis on video-video fusions have been explained. In this last type of frameworks, two compensated video-based trackers provide more robustness than individual solutions. This is the starting point of the next chapter.

Camera tracking combining a TDA and BUA using markers and feature points

6

Hybrid trackers have been proven to improve the performance. As discussed in the previous chapter, some studies have shown the viability of combining video-based camera trackers. To reach a synergy, techniques with complementary performance must first be identified. We focus our research on identifying such techniques and developing a camera tracker that combines them at two levels: the approach and the measurement level. Interesting properties of some top-down (TDA) and bottom-up (BUA) approaches have been identified in related research. Designing a framework that links these approaches is one of the goals. Both bottom-up and top-down approaches use mainly two sorts of cues (also called measurements) to update the pose estimation. Those cues are based on fiducial markers and on natural features in the scene. The other goal is hence to combine both sorts of cues.

This chapter describes a camera tracking framework developed to combine a TDA and a BUA using markers and feature points. The next section describes the goal and the targeted scenario. The system is described in detail in Section 6.2. Finally, the evaluation of the performance is dealt with in Section 6.3.

6.1 Introduction

The camera tracking framework described here proposes a fusion at two levels, namely, at the approach level and at the measurement level. Let us first concentrate on the approach combination. As discussed in Section 5.5, the goal of a generic camera tracking algorithm is to be capable of dealing at the same time with automatic pose initialisation, automatic re-initialisation after loss of track, and partial or complete occlusion of the reference (usually a marker, a pattern or an object's model). Depending on the approach, attention is usually paid on one or another aspect. Our purpose is to address these problems simultaneously by fusing approaches that solve them individually. A detailed analysis of state-of-the-art presented in the previous chapter allows the following identification of complementary techniques. On one side, BUAs such as marker-based and tracking-by-detection techniques generate frame-by-frame estimates (drift-free) and hence recover easily from an erroneous estimation. In this case, initialisation and re-initialisation is performed automatically by the tracking algorithm. On the other side, TDAs produce accurate trackers by keeping recursive and hence time-coherent estimates. Moreover, the tracking area can be extended beyond the space where the initial reference lies. Indeed, the initial reference can be occluded and tracking can continue based on dynamically added references. However, recursive estimation is prone to drift. As it can be seen, one should be able to fulfil the previous list of capabilities by using these approaches together. Among the mentioned BUAs, marker-based tracker have a low computational complexity but generally fail with occlusions, whereas tracking-by-detection systems do not fail with occlusions but are more complex either at run-time or during an off-line training. In order to keep complexity low, we choose a marker-based tracker.

Let us now move to the cue combination that we propose, namely, square markers and feature points. Fiducial markers are highly contrasted and hence easily detectable. Moreover, pose can be computed from a single instance of a marker thanks to their known three dimensional shape. Their drawback is that, in general, illumination changes or even small partial occlusions can easily cause a problem to their detection. Feature points are more flexible as they can be found individually in real scenes. Although a single point cannot generate a pose estimation, they can contribute additively to constrain the pose estimation. As it can be seen, using both cues enriches the available information and hence could augment the possibilities of achieving accurate tracking.

The analysis of complementary approaches and cues leads to the purpose of this chapter. We want to investigate the improvement achieved by fusing a marker-based tracker (BUA) and a feature point-based Bayesian tracker (TDA). This investigation is divided in two steps. Firstly, developing a camera tracking system that combines such techniques. Secondly, evaluating its performance when compared to individual tracking results.

Now that the goal is specified, an environment has to be defined in order to determine the needs, limitations and applicability of the proposed method. In this thesis, we concentrate on environments with three main characteristics. Firstly, textured regions are present in the scene. The reason being that textured regions promote the existence of feature points. Secondly, the scene is rigid. In order to estimate camera pose from feature points, these feature points must keep their location in the scene throughout the tracking. Thirdly, that a certain preparation of the environment is accepted by the potential user of this method. This preparation consists in the placement of a fiducial marker in the scene. Both indoor and outdoor environments that fulfil the first and second characteristic can be found. However, it is less common to allow placing markers in outdoor environments than indoor, which are fully controllable. Hence, we concentrate here in indoor environments without loss of applicability of the method to outdoor environments.

Another important issue of the targeted scenario is the motion that the camera follows. The most severe motion in terms of errancy is that of a hand-held camera. Other less erratic motion examples are that of a camera attached to a Head Mounted Display (HMD) typically used in Virtual Reality applications, or much less, a camera attached to a robot. The less erratic the motion is, the easier the task of filtering the camera's pose. In order to keep the proposed method applicable to a larger set of environments, we have chosen to deal with a scenario where the camera is hand-held and rapid manoeuvres can occur.

6.2 System description

The tracking system proposed uses a filter to keep track of the camera's pose. This filter is updated using two cues, one given by a marker-based tracker (BUA) and another one based on feature points searched with a TDA. The combination of cues is the particularity of the proposed framework and depends on the availability of the marker. When a marker is visible and the marker-based approach detects it, the filter is updated using this detection and the pose derived from it. This measurement is called Marker Cue (MC). On the other hand, when the detection of the marker is not available, the system relies on the feature points that are visible in the scene. In this case, the filter is updated using the localisation of these points. This measurement is called Feature Point Cue (FPC). Figure 6.1 shows the flow diagram of the system.

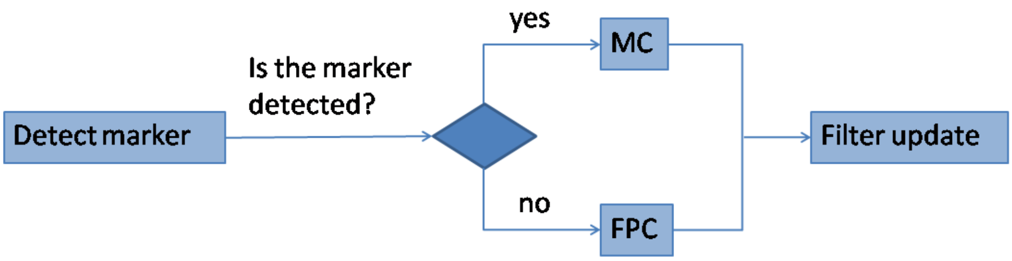


Figure 6.1: Flow diagram of the tracking system.

This section deals with the various components of the proposed system. Section 6.2.1 describes the filter and the motion model that has been chosen. The operation of the MC and of the FPC is explained in Sections 6.2.2 and 6.2.3, respectively. Section 6.2.4 describes a method to dynamically add new feature points in the scene, extending the trackable area. Then, the way in which the two cues are combined in the filter update step is depicted in Section 6.2.5. To conclude the system description, a method to dynamically adapt the motion model of the filter to manoeuvres is presented in Section 6.2.6.

6.2.1 Particle filter

As seen in Section 5.3.1, the Kalman filter (KF) has been extensively used for ego motion tracking. However, as mentioned in the introduction, we target applications where the camera is hand-held. Under such context, KF-based approaches lead to a non optimal solution because the motion is not white nor has Gaussian statistics. This fact has been identified by several researchers [RDLW95, Low92, CNHV99, AMR04]. Ababsa *et al.* [AMR04] compared the performance of a particle and a Kalman filter when tracking head motion. Although experiments are performed on synthetic data only, this work shows the higher performance achieved by particle filters using the same motion and measurement models.

Inspired by these results, we have chosen a camera tracking algorithm that uses a particle filter. More precisely, we have chosen a Sample Importance Resampling (SIR) filter (see Section 4.3.2, p.55). As discussed in Section 5.3.1, p.71, during the development of this thesis research on applying particle filters to hand-held camera tracking has been conducted successfully. This confirms the suitability of our choice. The characteristics of our filter are specified hereafter. More concretely, the space of the state vector and the prior transition and likelihood models are explained.

State space

The task of the filter is to keep an estimate of the pose of the camera in the world coordinate system. Therefore, the state space is represented by the translation and the rotation of the camera (see Section 4.2). We choose to describe the state with a seven-dimensional space. Three dimensions tackle the translation of the camera and the remaining four are used to express the rotation with a quaternion. In this way, each particle n in the filter represents a possible transformation at frame k

$$\mathbf{T}_k^n = [t_X, t_Y, t_Z, rot_W, rot_X, rot_Y, rot_Z]_k^n, \tag{6.1}$$

where $\mathbf{t} = [t_X, t_Y, t_Z]$ is the translation and $\mathbf{rot} = [rot_W, rot_X, rot_Y, rot_Z]$ is the quaternion for the rotation.

As it can be seen, velocity terms are not included in the state space. As a matter of fact, bayesian tracking with particle filters is computationally demanding due to the individual treatment of the particles (often thousands of them). In order to avoid slowing down the system, we have chosen to keep only positional terms in the state space. For particular applications, a tradeoff can be made between the number of particles, the dimension of the state space and the achieved processing time.

Filter prediction/update

For each video frame, the filter follows two steps: prediction and update (see Section 4.3). The prediction step is related to the propagation of the state from the previous frame to the current. The update step is related to the correction of the prediction with the measurements available at the current frame.

The probabilistic motion model for the prediction step is defined as follows. The transition prior $p(\mathbf{T}_k^n|\mathbf{T}_{k-1}^n)$ is modelled with a uniform distribution centred at the previous state $\mathbf{T}_{k-1}^n$ at frame $k-1$, with variance $\mathbf{q}$ (process noise vector). The reason for this type of random walk motion model is to avoid any assumption on the direction of the motion. Hence, this distribution enables faster reactivity to abrupt changes. This model, known as *max distance model* in the robotics community, is often used for legged robots [GF02]. The propagation for the translation vector is

$$\mathbf{t}_k^n = \mathbf{t}_{k-1}^n + \mathbf{u}_t \tag{6.2}$$

where $\mathbf{u}_t$ is a vector of random variables coming from the uniform distribution, particularised for the translation. The propagation for the rotation is

$$\mathbf{rot}_k^n = \mathbf{u}_{rot} \times \mathbf{rot}_{k-1}^n \tag{6.3}$$

where $\times$ is a quaternion multiplication and $\mathbf{u}_{rot}$ is a quaternion coming from the uniform distribution of the rotation components.

Now that the prediction step is defined, let us specify the update step. The weight of each particle in a SIR particle filter using the transition prior model as proposal distribution is computed using the measurement noise or likelihood (see Section 4.3.2)

$$w_k^n = p(z_k|\mathbf{T}_k^n), \tag{6.4}$$

where w^n is the weight of particle n and z is the measurement. The key role of the update step is to combine the cue provided by either the MC or the FPC.

Once the weights are obtained, these are normalised and the correction of the filter state is concluded. The corrected mean state $\widehat{\mathbf{T}}_k$ is given by the weighted sum of $\mathbf{T}_k^n$

$$\widehat{\mathbf{T}}_k = \sum_{n=1}^{M} w_k^n \cdot \mathbf{T}_k^n, \tag{6.5}$$

where M is the number of particles. This mean is used as output of the camera tracking system at frame k.

6.2.2 Marker cue

As seen in Section 5.2.1, geometric patterns such as squares provide enough information to recover the pose of the camera capturing them. Among the various existing marker-based tracking techniques, we have selected ARToolkit [ART07] because of its high detection rate and estimation speed. This marker-based tracker and some of its variants, are at the core of many applications [WS03, BK02, BVBC04].

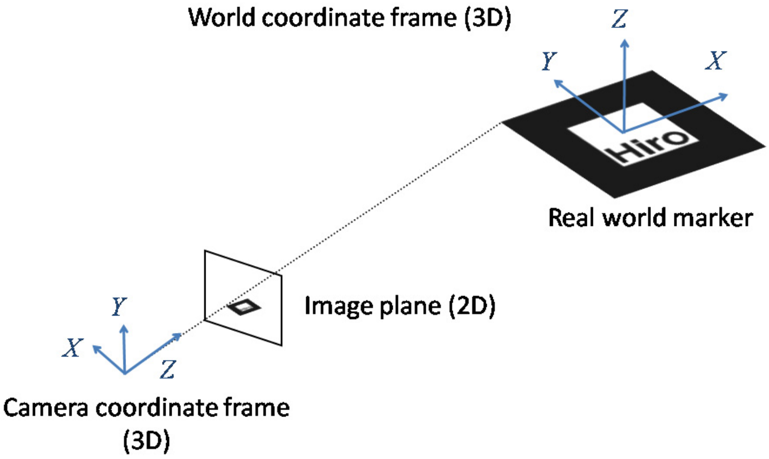
ARToolkit provides tools to calculate the transformation **T** between the world coordinate frame and that of the camera. Actually, the world coordinate frame can be arbitrarily fixed to any 3D position of the real world. We choose to fix the worlds centre $(0,0,0)$ at the centre of the marker with the Z axis pointing normal to the marker and X and Y axis parallel to the sides of the marker. Figure 6.2 shows the coordinate systems of the world and the camera, as well as examples of the world coordinate overlaid on the marker for different real viewpoints. For convenience, the state-space of the transformation provided by ARToolkit coincides with that of the particle filter.

At each frame, the algorithm searches for a square marker inside the field-of-view (FoV) and from this detection, computes the transformation from the marker to the camera $\mathbf{T}^{\mathrm{MC}}$. Although a detailed description of the algorithm can be found in [KB99] and [ART07], an overview of the method is given hereafter for the sake of completeness.

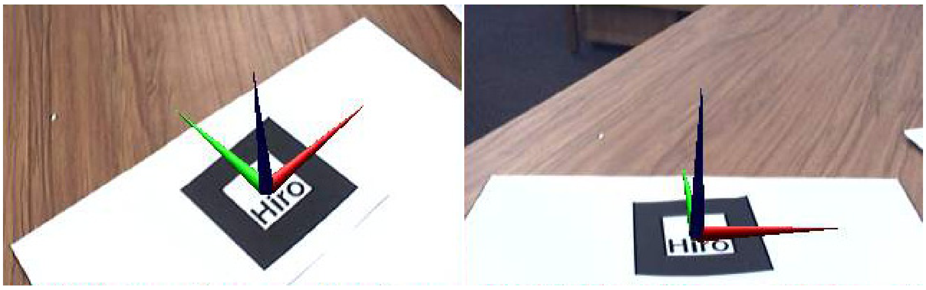
This algorithm can be divided in seven consecutive steps. The first four steps are visually depicted in Figure 6.3b-e.

1. The image frame is binarised (with a fixed threshold) and the result is a binary image used in steps 2 and 3.

2. Connected components are labelled, each one of them being a potential marker.



(a) Relation of world coordinate frame and camera coordinate frame.



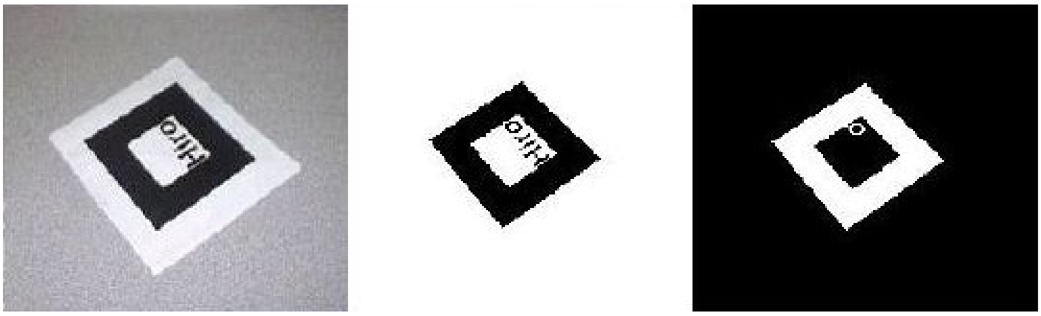
(b) Example of world coordinate overlaid on the marker for different real viewpoints.

Figure 6.2: World coordinate frame fixed to the center of the marker. Images reproduced from [ART07].

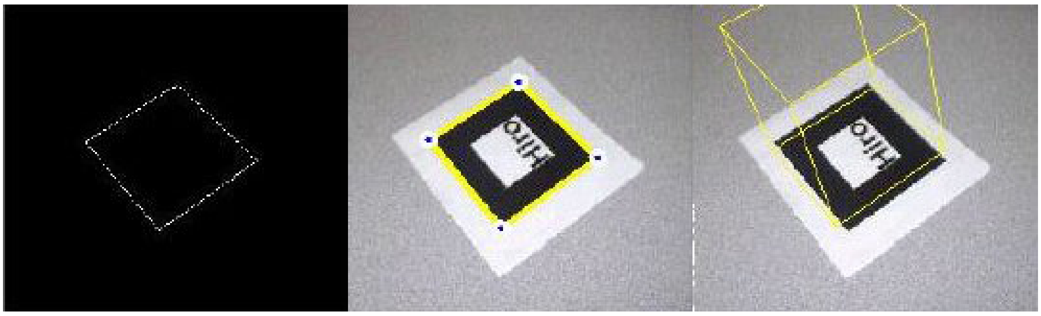
3. The contour of each component is detected and line segments estimated. Only components with four lines are kept.
4. Once the lines are parameterised, it is possible to estimate the location of its intersections (which might correspond to the real corners).
5. The homography that transforms the square into the image plane is computed. This is done using the relation between the known geometry of the square marker and the intersections of the contour lines.
6. This homography permits also the normalisation of the pattern inside the marker.
7. Template matching is used to identify the marker and hence eliminates false candidates, and to determine the exact orientation of the pattern.

The homography together with this last step leads to the desired transformation $\mathbf{T}^{\mathrm{MC}}$. The resulting transformation can be used to fit a virtual wired cube on top of the marker (Figure 6.3f).

$\mathbf{T}^{\mathrm{MC}}$ is the measurement fed into the filter for update. If a marker is detected, the transformation can be computed. As can be seen from the description of the algorithm, this computation uses



(a) Original image. (b) Thresholded image. (c) Connected components.



(d) Contours. (e) Extracted edges and corners. (f) Fitted wired cube.

Figure 6.3: Steps of the MC algorithm. Images reproduced from [ART07].

only the geometric relation of the four projected lines that contour the marker in addition to the recognition of a non-symmetric pattern inside the marker. When this information is not available, no pose can be calculated. This occurs in the following cases:

- markers are partially or completely occluded by an object;
- markers are partially or completely out of the FoV;
- or not all lines can be detected (e.g., due to low contrast that influences the binarisation).

6.2.3 Feature points cue

As discussed in Section 5.2.2, natural features such as objects, lines, edges, or points provide reliable information for visual tracking. In this work, natural feature points have been chosen as another cue to improve the robustness of the tracking system. Indeed, when feature points are detected and the 3D location of the point in world coordinates is known, it is possible to constrain the camera pose. Recalling from Equation (4.5), p.49, the 3D location of a feature point $\mathbf{X}$ and its 2D back-projection $\mathbf{x}$ are related by the camera pose as follows

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_k = \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{6.6}$$

where λ is a proportion factor, $\mathbf{K}$ is the calibration matrix, $\mathbf{R}$ is the rotation matrix formed using the quaternion **rot**, and $\mathbf{t} = [t_X, t_Y, t_Z]^T$ is the translation vector. The calibration of the camera is performed off-line using the method described in [KB99].

We propose to keep the 3D information of a set of feature points F. Then, from their detection in the image plane, we constrain the pose. Let us now analyse how the feature points contribute to the state correction. In the first part of this thesis, we have described a method to match feature points. Recalling from Chapter 3, this method is composed of a rotation-discriminative patch descriptor $f(\mathbf{P})$ (see Equation (3.7), p.24) and an efficient hierarchical search strategy. The descriptor is computed for each feature point belonging to the set F

$$F = \left\{ \left[\tilde{\mathbf{h}}_{\mathbf{P}}, \sigma^2_{\mathbf{P}}, \|\tilde{\mathbf{h}}_{\mathbf{P}}\|, \overrightarrow{\mathbf{P}}, \mathbf{X}, \sigma^2_{\mathbf{X}}, \hat{\psi}_Z \right]_j \quad j = 1, \ldots, L \right\}, \tag{6.7}$$

where $\mathbf{X}$ is the estimated 3D position, $\sigma^2_{\mathbf{X}}$ is the variance of this position, and $\hat{\psi}_Z$ is the orientation of the camera around the Z axis at the time of detection of the feature point. This orientation is used in the weighting of particles as described below in Section 6.2.5.

At each video frame, the feature points must be localised. A region is defined around the estimated location of each feature point (see Figure 6.4). Assume for the moment that those regions are known. The search is performed using the Rotation-Discriminative Template Matching method.

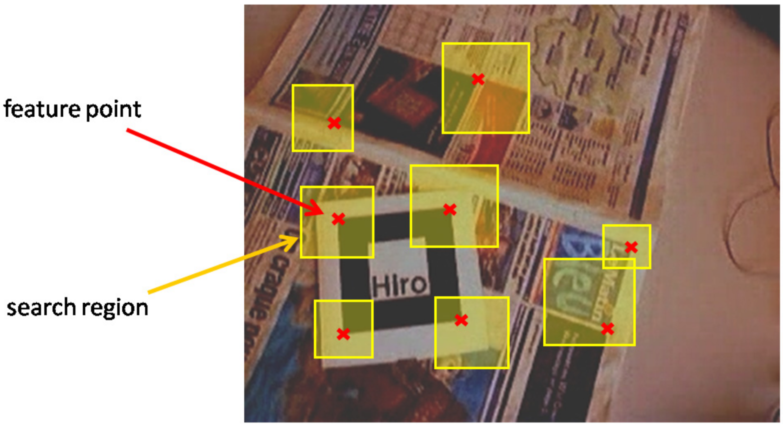


Figure 6.4: Feature points are searched within regions of a video frame.

Both the orientation and the correlation maps (see Equations 3.15 and 3.16, 26) are computed for each region. The set of correlation Ψ_j and orientation Θ_j maps is the measurement fed into the filter for update.

Each template that is positively correlated makes the filter converge to a more accurate estimate. Three points are necessary to determine the six DoF of the camera pose. With two feature points, only the translation is fully determined, whereas the rotation has one free axis. However, the filter can be updated even with only one feature point, which constrains t_X and t_Y. A reliable feature point might be unavailable in the following situations:

- the region does not contain the feature point (due to a bad region estimation);

- or the feature point is inside the region but no correlation is high enough (e.g., because the viewpoint is drastically changed).

The first situation is related to top-down camera tracking and is discussed later. The second situation is related to feature point recognition. The RDTM method is tailored to two dimensional rotations with respect to the normal of the template. When applied to 3D camera tracking, this is comparable to a rotation around the Z axis. In general, it is possible that the template that describes a feature point undergoes full 3D rotations. However, we will consider that the assumptions made for the recognition method are valid for a large number of frames. In some situations, the rotation of the camera with respect to X or Y axes is very different to that at the frame when the feature point was detected. In such situation, the 3D rotation of the template cannot be approximated by a 2D rotation. In general, the correlation might not be high enough when the visual appearance of the feature point differs too much from the original template. Besides the rotations problems mentioned above, it could even occur that a feature point lies on a point of real discontinuity (e.g., the corner of an object). The projection of its visual neighbourhood in the image plane would change completely its aspect when the camera moves. Both the 3D rotation and the discontinuity problem are addressed in the framework by relying on other feature points with appearances that can still be approximated.

Natural feature points in unprepared environment appear in objects at unknown locations. Hence, the 3D location of feature points in the world coordinate frame is generally unavailable. However, the combination framework proposed here admits a certain preparation of the environment, this is, a marker is available. Since the world coordinate frame is fixed to the marker and the real size of the marker is known, the 3D location of any point in the marker is known. We take advantage of this fact and propose to use the corners as feature points in the scene. In this case, the variance of their location $\sigma_{\mathbf{X}}$ is set to $[0,0,0]$

Although we have proved that these points provide a reliable measurement for camera tracking [MMAE07, ME07a], they might not always be available. For instance, because a corner is occluded by an object or it is outside of the FoV. In this case, it is interesting to have other feature points to rely on. As explained before, in order to constrain the camera pose, the 3D position of a feature point must be available. However, the inverse procedure can also be done. Indeed, from Equation (6.6) one deduces that the 3D world coordinates of a point can be computed if the camera pose $[\mathbf{R}|\mathbf{t}]$ is known. Since the filter keeps an estimate of this pose, it is possible to calculate the 3D position of feature points. Once this location is computed, a new feature point can be added to the set of feature points F that constrain the camera pose. This process is detailed below in Section 6.2.4.

6.2.4 Adding feature points

The camera framework proposed relies on feature points found in the scene. Besides the four corners of the marker, a method to find the 3D position of other feature points in the scene is necessary. It is then possible to continue tracking when the four corners are occluded and hence extend the tracking area. We propose a method to estimate this 3D position based on the camera pose and benefiting from its motion.

The process of adding new points to the set of references is generally known as mapping. Although a brief introduction has been given previously in Section 5.3.1, more details are given through several examples hereafter. Mapping new points is a common problem in the robotic community. A recent survey by Thrun [Thr03] identified the different techniques found in literature. Among these techniques, we concentrate here on a specific subset related to our context. More precisely, mapping is performed while the camera pose is being estimated.

Determining the 3D position of a feature point starts in general with the detection of that feature point at frame k. This determines a 2D point (x,y) in image coordinates. Since the camera pose $[\mathbf{R}|\mathbf{t}]_k$ is known, it is possible to determine the 3D position in world coordinates $B(x,y)$ of the 2D point. Using Equation (4.1), p.48, the camera internal parameters and the pose of the camera, one has

$$B(x,y) = \mathbf{R}_k^\top \cdot \begin{bmatrix} x - c_x \\ y - c_y \\ f \end{bmatrix} - \mathbf{R}_k^\top \cdot \mathbf{t}_k, \tag{6.8}$$

where $\top$ indicates matrix transpose, (c_x, c_y) are the coordinates of the principal point and f is the focal length (see Section 4.2.1, p.50). The line that intersects the camera centre $\mathbf{t}_k$ and this point determines the ray where the feature point lies. We call this the *depth line*

$$\mathbf{l} = B(x,y) - \mathbf{t}_k. \tag{6.9}$$

Figure 6.5a depicts the depth line that crosses the feature point in the image plane and the camera centre. Determining the 3D position is reduced to a problem of finding the depth point along this line. This is often done by triangulating the ray with points that match the descriptor of the feature point in the upcoming frames.

Related research

Most methods estimate the 3D position of the feature point by updating a probability distribution along the depth line. Two categories can be found namely, direct depth and inverse depth. Direct depth methods initialise a distribution in the space Z (e.g., [Dav03, PC05]). Conversely, inverse depth methods update a probability distribution in the space $1/Z$ (e.g., [LLS05, ED06, CPMCC06]).

Davison [Dav03] sets particles uniformly distributed between 0.5 and 5 metres from the camera centre. Using particles implies representing discrete depths. Each particle is back-projected to the image plane. The particles are re-weighted according to template matching. One advantage of setting a prior on the depth region is that scanning the current frame for matches can be performed efficiently. Indeed, instead of scanning all along the projection of the depth ray in the current image frame, matches are searched within ellipses (one per particle) representing the particle uncertainty in image coordinates. When the weight of one of the particles is significantly peaked, the process stops and the position of the corresponding particle is taken as the depth of the feature point. Pupilli and Calway [PC05] propose a different approach. At initialisation, particles are uniformly distributed along the ray, just as Davison (no details about depth ranges are given in [PC05]). In upcoming frames, camera particles are sampled. Each one of these camera particles is triangulated with highly correlated points in the image. Depth particles are then re-weighted according to correlation coefficients and distance from triangulation point to depth particle location.

Lemaire *et al.*[LLS05] have investigated the use of Gaussian distributions instead of particles. The authors propose prior depth distribution consisting in a sum of gaussians uniformly distributed along the depth line. The process consists in iteratively pruning unlikely hypotheses until one likely Gaussian remains.

Proposed method

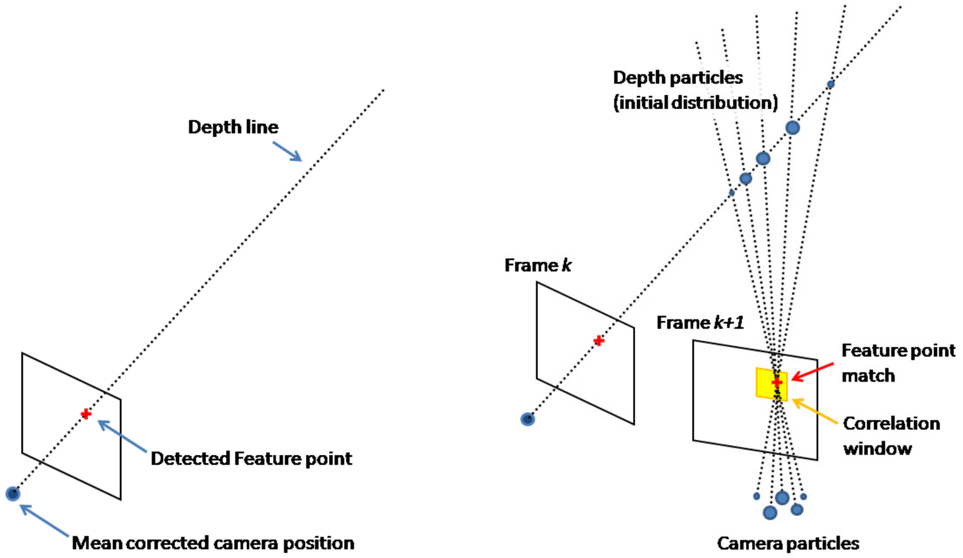
Inspired by these works [PC05, LLS05], we propose a method that builds upon them. On one side, the depth distribution is described with particles. On the other side, Gaussian propagation around each of these particles is employed to better simulate depth uncertainty.

Our method differentiates from these works in several aspects. First, the way in which depth hypothesis are initialised changes. The initialisation of these methods assumes that the range of depth is known. Accordingly, particles or Gaussians are uniformly distributed along the depth line. In our case, once a feature point is detected and the consequent depth line is established, no distribution is set. The initialisation is delayed one frame. At the following frame, particles are set at triangulated locations in the depth line. In this way, the initial depth particles have higher probability of being in the correct range. Another property of these works is that the location of the hypotheses is static during the process. We propose a fully operating SIR filter for depth estimation. More precisely, particles propagate, are weighted and then re-sampled. The propagation is performed by adding a gaussian noise to their location. The weighting is done similarly to [PC05]. The re-sampling permits to eliminate particles with small weight, representing unlikely depths and hence converging to the real depth. Notice that such a particle filter permits to refine the depth estimation iteratively. The uniform distribution of the other techniques only permits a resolution fixed at initialisation.

Let us now give an overview of the proposed algorithm. This algorithm takes several steps grouped as follows. Firstly, a feature point is detected at frame k. Secondly, the prior distribution of particles is set at frame $k+1$. During frames $k+\Delta k$, with $\Delta k > 1$, the process continues by triangulation and re-weighting of particles. These three steps are depicted graphically in Figure 6.5. As soon as the distribution shows enough convergence, the feature point is added to the set F (see Equation (6.7)). However, if this process lasts too many frames, the candidate is discarded. Details of this algorithm are given hereafter.

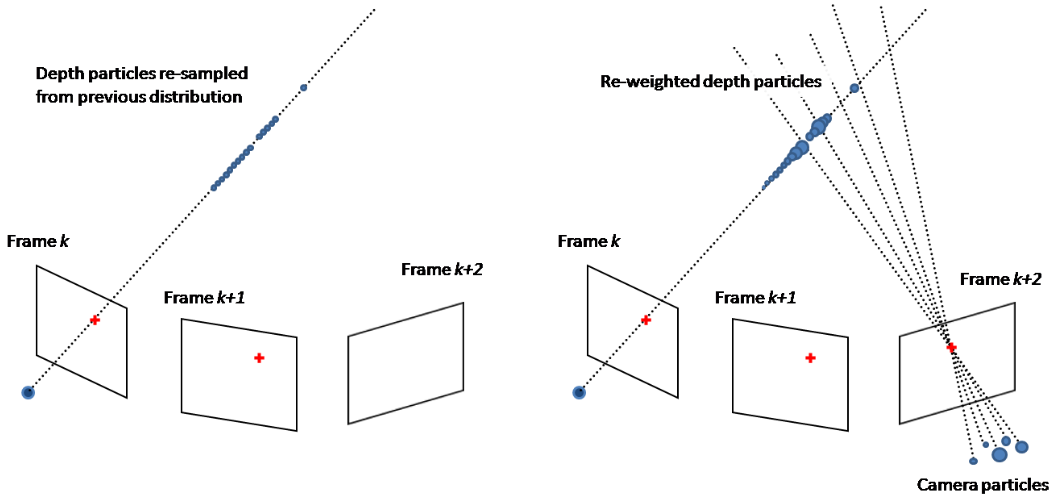
The process starts with the detection of a candidate feature point, at frame k. Feature points are searched at areas of the image where no other feature point candidate is being searched nor there is a feature point already belonging to the set F. Feature points are extracted with the method described in Section 3.5.1, p.28. This method extracts points at Harris corners but with a specific size and provided their orientation histogram has a heterogeneous shape. In this way, future feature point localisation with RDTM is prone to give better results. The template at the specified size is stored for recognition during the depth estimation process. With this first recognition the vector in the direction of the depth line can be computed

$$\begin{aligned} \mathbf{l} &= B(u,v) - \widehat{\mathbf{t}}_k \\ \hat{\mathbf{l}} &= \mathbf{l}/||\mathbf{l}|| \end{aligned} \tag{6.10}$$



(a) Frame k. Initial depth line upon feature point detection and mean camera position $\widehat{\mathbf{t}}_k$.

(b) Frame $k+1$. Initial depth particle distribution according to camera particles triangulation with correlated template.


Depth particles re-sampled
from previous distribution
Frame *k*
Frame *k+1*
Frame *k+2*
Re-weighted depth particles
Frame *k*
Frame *k+1*
Frame *k+2*
Camera particles

(c) Frame $k+2$. Depth particles are re-sampled from previous weighted distribution at frame $k+1$.

(d) Frame $k+2$. Depth particles are re-weighted upon triangulation of camera particles using the correlated template.

Figure 6.5: Mapping of new feature points. First three frames shown.

where $\widehat{\mathbf{t}}$ is the corrected mean camera translation (see Equation (6.5)).

At frame $k+1$, a NCC-based window search is performed around the previous location of the feature point. The most correlated point (p_x, p_y) is kept. If the NCC at this point is greater than 0.8 the process continues, otherwise, the candidate is erased. Then, the best M_d camera particles are selected. The lines joining each selected camera particle with the 3D point $B(p_x, p_y)$ are computed. The prior distribution can now be set. More precisely, M_d depth particles are set at the point triangulating those lines with the depth line set at the previous frame. The weight of each depth particle is equal to the weight of the corresponding camera particle.

In upcoming frames $k+\Delta k$ the update of the depth particle filter takes place. Firstly, the state of each particle $\mathbf{d} = [X,Y,Z]^\top$ propagates with the following transition model

$$\mathbf{d}^n_{k+\Delta k} = \mathbf{d}^n_{k+\Delta k-1} + e^{n^\top}_{k+\Delta k-1} \cdot \hat{\mathbf{l}} + [m,m,m]^{n^\top}_{k+\Delta k-1}, \tag{6.11}$$

where e and m are random variables with Gaussian distribution and variance $\sigma^2_{e,\Delta k-1}$ and $\sigma^2_{m,\Delta k-1}$, respectively. This propagation promotes variations along the depth line together with variations in all directions. Variations along the depth line are expected. Therefore, particles may fall at the right depth after some iterations, which is not the case of the static initialisation done in the cited works. The second additive noise models the error in the computation of the depth line. This error comes from the uncertainty in the camera pose at frame k. A precise model would consider the exact camera particles distribution. However, we consider this precision unnecessary and keep a simple model of this error. Moreover, both noises vary along time. An experimental observation shows that as more iterations are performed, the particle filter converges as expected. In order to fasten this convergence the noises are iteratively reduced. This also permits to use large initial variances $\sigma^2_{e,1}$ and $\sigma^2_{m,1}$ at frame $k+2$. This models possible errors in particle distribution initialisation at the previous frame. Secondly, the M_d triangulation lines are set just as for frame $k+1$. Thirdly, depth particles are re-weighted according to the distance between the intersection of triangulated rays and the weight of the corresponding camera particle.

The process continues while the covariance of the state vector $\mathbf{d}$ converges and until a fixed number of frames have elapsed. The convergence criteria is that the variance of the depth distribution is below a certain threshold. This threshold is proportional to $\sigma^2_{e,\Delta k-1}$ and hence is reduced at each frame. The maximal number of frames to estimate the depth is fixed to 14. After 14 frames, the feature point is discarded. If the distribution converges in a minimum of 7 frames, the estimate is considered accurate enough. A feature point is then added to the set F. The rotation-discriminative descriptor described in Section 3.2, p.22 is built. This can be performed on-the-fly thanks to the simplicity of construction of our proposed feature descriptor. This contrasts with the trained-based techniques (where description can only occur off-line) or other more complex descriptors discussed in Chapter 2. The 3D position of the feature point $\mathbf{X}$ is equal to the mean of the distribution namely, $\hat{\mathbf{d}}_{k+\Delta k} = \sum w^n_k \cdot \mathbf{d}_{k+\Delta k}$, where w^n_k are the weights of the depth particle filter. The variance of this estimate $\sigma^2_{\mathbf{X}}$ is the diagonal of the covariance matrix of the depth particle filter. If Δk is greater than a certain number of frames, the candidate is erased. This usually happens when the extraction of a feature point provides an unstable point (e.g., a point in a line).

6.2.5 Cues combination

The goal of the system is to obtain a synergy by combining both cues, the MC and the FPC. In a synergy, the combination of several items gives a superior outcome when compared to the sum of individual outcomes. This idea is materialised in the proposed method by solving individual failure modes of each cue. On one side, special attention is given to the occlusion and illumination problems in the MC. On the other side, inaccurate feature point detections lead to less accurate filter updates. Hence, during long periods relying only on the FPC, the tracker drifts (see Chapter 5). This failure mode is also addressed by the proposed fusion.

As said before, the cues are combined at the level of the filter and, in particular, when the filter is updated. More precisely, it is the weighting of the particles that uses a different likelihood depending on whether the marker is detected or not. Details on the type of likelihood function used in each case are given below.

Prior to updating the filter, however, it is necessary to initialise the whole tracking system. This initialisation happens at the first detection of the marker. This allows the automatic pose initialisation which is one of the capabilities discussed in the introduction. Furthermore, it also permits to fix the first four feature points of the set F (corners of the marker), and the state of the filter. Indeed, all particles are set to the pose given by $\mathbf{T}^{\mathrm{MC}}$

$$\mathbf{T}_0^n = \mathbf{T}_0^{\mathrm{MC}} \quad n = 1, ..., M \tag{6.12}$$

where M is the number of particles. Once the system has been initialised, camera tracking starts. As described before, the camera pose is tracked by the particle filter whose state is corrected in the update step. The cue fed into the filter is the one that gives more reliable constrains at each frame.

As a matter of fact, the MC gives highly accurate and stable results when the marker is detected. However, no estimate is produced otherwise. We take advantage of this accuracy and rely on the MC as long as the marker is detected, regardless of the cue that the feature points could provide. Consequently, the system uses the MC measurement to update the particle filter ($z_k = \mathrm{MC}$), in each frame for which the marker is detected. In this case, the likelihood is modelled with a Cauchy distribution centered at $\mathbf{T}^{\mathrm{MC}}$

$$w_k^n\Big|_{z_k=\mathrm{MC}} = p(\mathrm{MC}|\mathbf{T}^n) = \prod_i \frac{\mathbf{r}(i)}{\pi \cdot ((\mathbf{T}_k^n(i) - \mathbf{T}_k^{\mathrm{MC}}(i))^2 + \mathbf{r}(i)^2)}, \tag{6.13}$$

where $\mathbf{r}$ is the measurement noise vector and i indexes the elements of the vectors. This particular distribution choice has its origin in the following reasoning. In the resampling step of the filter, particles with insignificant weights are discarded (see Section 4.3.2, p.55). Therefore, a problem may arise when most particles lie on the tail of the measurement noise distribution (likelihood) as they would be assigned small weights. The transition prior $p(\mathbf{T}_k^n|\mathbf{T}_{k-1}^n)$ determines the region in the state-space where the particles fall before their weighting. Hence, it is relevant to evaluate the overlap between the likelihood distribution and the transition prior distribution. When the overlap is small, the number of particles effectively resampled is too small.

Figure 6.6 shows an instance of overlapping region for a generic state space $\mathbf{x}$. The transition prior $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is modelled with a Gaussian distribution without loss of generality. The measurement noise $p(\mathbf{z}_k|\mathbf{x}_k)$ is modelled with two different distributions, namely Gaussian and Cauchy. Mathematically, the support for all these distributions is the entire real space of $\mathbf{x}$. However, it must be pointed out that due to computing limits, some values fall to zero even though their real mathematical value is greater than that. In the example of this figure, there is no sufficient computed overlap for the Gaussian distribution (commonly used), whereas the tail of the Cauchy distribution covers the necessary state-space. Therefore, we have chosen the second option with a long-tailed density that better covers the state-space. It is assumed that the error in the MC is well modeled by this type of distribution.

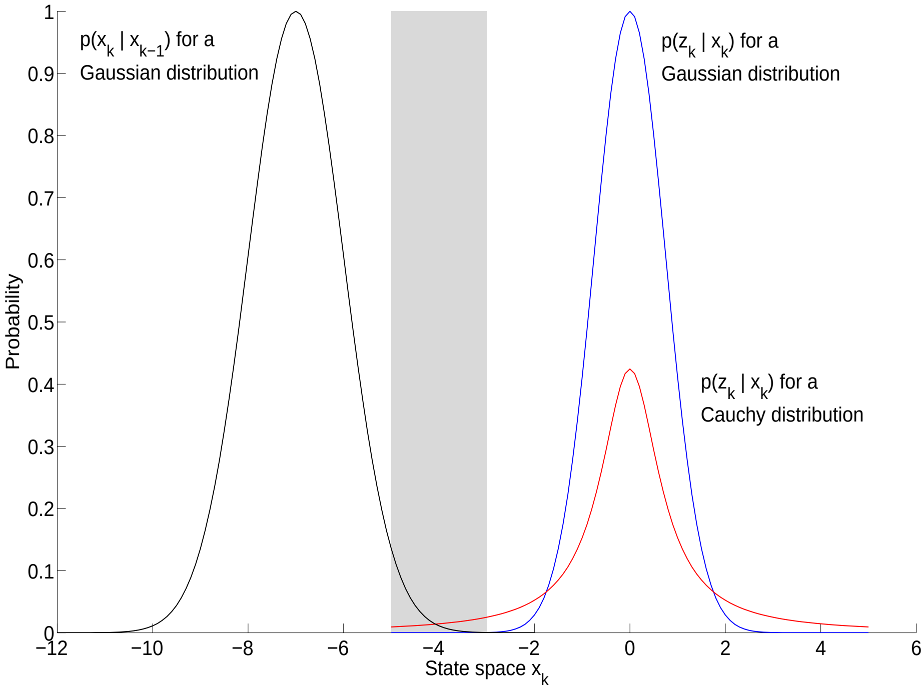


Figure 6.6: Overlap between transition prior distribution and the likelihood distribution: modelled with a Gaussian (no overlap) and with a Cauchy distribution (shaded region).

Although this cue greatly improves filter correction and particles convergence, it might not be available at each frame. The reasons for this were described earlier in Section 6.2.2). Hence, another cue is necessary in order to continue the tracking process. This alternative is the Feature Point Cue. In this case, the measurement z_k is the result of the Rotation-Discriminative Template Matching (RDTM) step, more concretely, the correlation $\Psi_{j,k}$ and the orientation $\Theta_{j,k}$ maps of the set of feature points. This cue is completely different from the MC as the measurement is not a value in the state-space $\mathbf{T}$ but a measure of localisation and viewpoint of the 3D points in the 2D image plane. The process to compute the particle weights when relying in the FPC takes several steps.

Firstly, the regions around the estimated 2D location of each feature point are computed. For each feature point, all the back-projections given the transformations $\mathbf{T}^n_{k-1}$ at the previous frame are computed (see Eq. 6.6). The region is the bounding box containing all these back-projections. The weight of the particles is considered and the bounding box is reduced so as to contain only representative back-projections. Figure 6.7 depicts this process graphically.

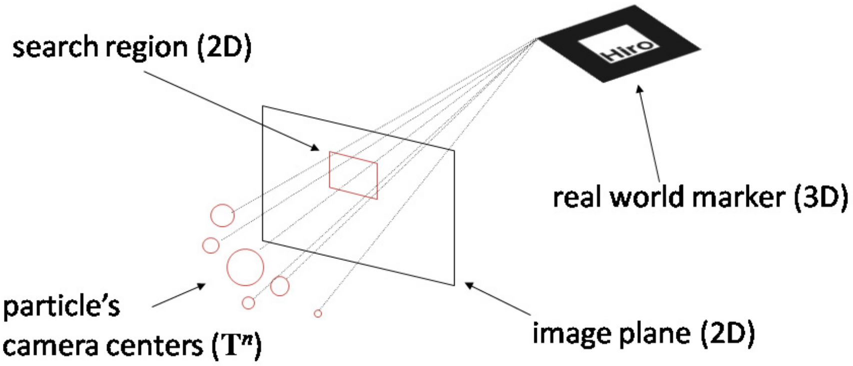


Figure 6.7: Feature point search region is determined using all the relevant back-projections given the transformations $\mathbf{T}^n_{k-1}$ and weights of the particles w^n_{k-1} at the previous frame. Weight represented with the size of the particle's camera centre.

Secondly, the set of bounding boxes is fed into the FPC and the RDTM maps are obtained in return. Thirdly, the most probable 2D locations are obtained by thresholding each correlation map $\Psi_{j,k}(x,y)$ leading to a set of 2D points

$$S_{j,k} = \left\{ \mathbf{c} = [c_x, c_y] \;\middle|\; \Psi_{j,k}(c_x, c_y) > th_{\text{corr}} \right\}, \tag{6.14}$$

where j indexes the feature points. The process continues with the particularisation for each particle n of the filter. Indeed, a subset $\widehat{S}^n_j$ is defined. This subset contains the points in S_j that are within a certain modified Euclidean distance $d_E(\mathbf{a}, \sigma^2, \mathbf{b})$ from the back-projection of the corresponding feature point $\mathbf{x}^n_j$

$$\begin{aligned}
d_E(\mathbf{a}, \sigma_{\mathbf{x}}, \mathbf{b}) &= \sqrt{\frac{(a_x - b_x)^2}{1+\sigma_x^2} + \frac{(a_y - b_y)^2}{1+\sigma_y^2}} \\
\lambda \cdot \mathbf{x}^n_{j,k} &= \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}]^n_k \cdot \mathbf{X}_j \\
\lambda' \cdot \sigma^n_{\mathbf{x},j,k} &= \mathbf{K} \cdot [\mathbf{R}|\mathbf{t}]^n_k \cdot \sigma_{\mathbf{X},j} \\
S^n_{j,k} &= \left\{ \mathbf{c} = [c_x, c_y] \in S_{j,k} \;\middle|\; d_E(\mathbf{c}, \sigma^n_{\mathbf{x},j,k}, \mathbf{x}^n_{j,k}) < th_{\text{dist},j,k} \right\},
\end{aligned} \tag{6.15}$$

where the threshold $th_{\text{dist},j}$ varies at each frame and for each feature point. Actually, it is fixed to half the diagonal of the corresponding search region. Notice that the uncertainty in the 3D position $\sigma_{\mathbf{X}}$ of feature points is considered in this subset.

Finally, the weight is computed. The weight of the particle n is proportional to the correlation $\Psi_{j,k}$ achieved in the subsets $S^n_{j,k}$. Furthermore, this is refined with the orientation $\Theta_{j,k}$ estimated by the RDTM process. This orientation should have a rough correspondence with the rotation of the camera about the Z axis. The more perpendicular is the original template to the current pose of the camera, the higher the chances of the estimated orientation being similar to the rotation about the Z axis. We take advantage of this fact. Indeed, the weights are forced to be proportional also to the difference between the orientation $\Theta_{j,k}$ and the rotation of the corresponding particles's state ψ_Z

$$w_k^n\Big|_{z_k=\text{FPC}} = exp\left(\sum_{j=1}^{L}\sum_{[x,y]\in S_{j,k}^n} \Psi_{j,k}(x,y)\cdot exp - \left(\frac{(\psi_{Z,k}-\hat{\psi}_{Z,j})-\Theta_{j,k}(x,y)\cdot\Delta}{\alpha\cdot\Delta}\right)^2\right), \qquad (6.16)$$

where L is the number of feature points, $\Delta = 360/N$ is the quantisation step of the orientation according to the number of bins N (see Chapter 3), $\hat{\psi}_{Z,j}$ is the rotation of the camera at the initialisation of the feature point (see Equation (6.7)), and α is a tunable parameter. Weighting the particles according to the correlation gives already a strong validation for the data association between feature points and the point in the image plane where they lie. Reinforcing this validation with the orientation permits to avoid confusion with points with high correlation but unexpected orientation according to the camera's pose. Therefore, α can be tuned to vary this reinforcement of the data association. In our case, this parameter is fixed to a high value ($\alpha = N/2$) as the perpendicularity of the camera with respect to the template of a feature point cannot be assured a priori. It is also possible to make this parameter vary according to the angle of rotation in X and Y axes, for instance $\alpha \propto \sum|\psi_{X,k}-\hat{\psi}_{X,j}|+|\psi_{Y,k}-\hat{\psi}_{Y,j}|$. This option is not considered for simplicity purposes.

As it can be seen, the likelihood for the FPC measurement is much less straightforward to compute than for the MC. Nevertheless, the advantage of this cue is that the weights can be calculated independently of the number of feature points recognised, whereas the likelihood for the MC is available only if the marker is detected.

Algorithm 6.1 expresses the process followed to combine these two cues. It is assumed that the filter has been initialised at the first detection of the marker. Note that the description of the marker is stored in the *pattern* variable.

Algorithm 6.1 Combination procedure

```
loop
  $vframe$ ← getVideoFrame()
  $marker$ ← detectMarker( $vframe$ )
  filter.propagate()
  if $pattern$.correspondsTo( $marker$ ) then
    $\mathbf{T}^{\text{MC}}$ ← MC.calcTransformation( $marker$ )
    $\widehat{\mathbf{T}}$ ← filter.updateFromMC( $\mathbf{T}^{\text{MC}}$ )
  else
    $regions$ ← filter.calcRegions($F$, $vframe$)
    $[\Psi,\Theta]_{j=1,\ldots,L}$ ← FPC.RDTM( $regions$ , $F$ )
    $\widehat{\mathbf{T}}$ ← filter.updateFromFPC( $[\Psi,\Theta]_{j=1,\ldots,L}$ )
  end if
end loop
```

Let us now discuss the advantages of the proposed cue combination. As explained above, the update of the filter is performed by switching between two sorts of likelihood depending on the type of measurement that is used: MC or FPC. This low-level fusion framework (see Section 4.4, p.57 for an extended definition) has several advantages:

- The most reliable cue is chosen automatically at each frame. Indeed, the FPC provides a fall-back process for the MC when the marker is not detected.
- The combination through a filter provides a continuous estimate which is free of jumps. On the other hand, in a high-level framework where the output switches between trackers (for instance, [OKS04]), the output jumps between the estimates as no common track is kept.
- The combination framework presented is a loose coupling of approaches. This increases modularity. Indeed, individual trackers do not need to be modified to be integrated in the framework. Most hybrid tracking systems have tight coupling between cues and hence give little opportunity for shaping. The likelihood switching method proposed is generic enough to be used with very different types of cues, such as model-based, or sensors, such as inertial or acoustic.

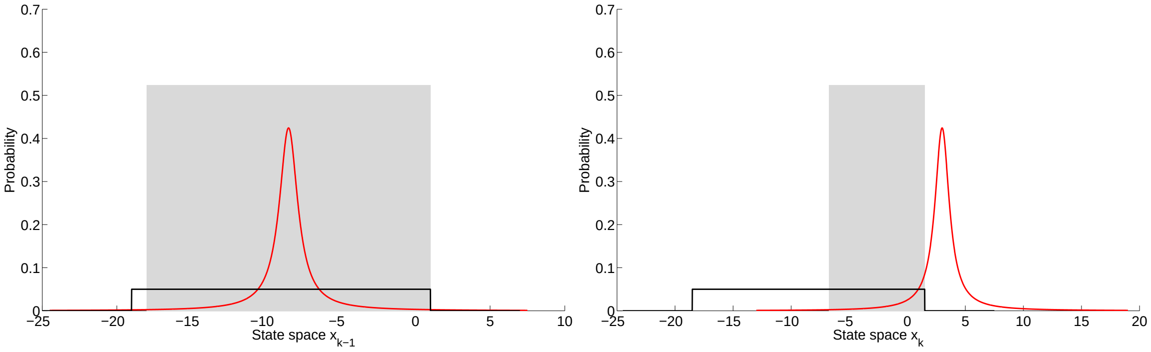
6.2.6 Dynamic tuning of the filter

Filtering techniques often suffer from the difficulty of modelling the motion with precision. More concretely, the errors in the model are usually simulated with noise of fixed variance, also called *hyper-parameter*. In practice, these variances are rarely constant in time. Hence, better accuracy is achieved with filters that adapt, or self-tune, the hyper-parameters online [May82]. The goal of the dynamic tuning presented here is to achieve better tracking accuracy together with robustness in front of manoeuvres.

With the framework presented so far, the distribution of the measurement noise is not sufficient to cover the state-space in front of rapid manoeuvres. Indeed, neither the long-tailed Cauchy-type distribution of the MC, nor the arbitrary distribution generated by the FPC can handle small overlaps between the predicted region of the state space and the measured one. Figure 6.8 shows the effect of a large manoeuvre on the probabilistic model assumed for the MC. Again, the computing limits play a role in the positive overlap of distributions.

Either the likelihood or the transition prior distributions should broaden in order to face this problem. The measurement noise model is related to the sensor. Hence, the model should only be tuned if a quality value of the measurement provided by the sensor is available. For video sensors this value could be, for instance, the number of inliers to a geometric constraint or a confidence value in the detection of the marker. This quality value is not necessarily correlated to motion and thus is of no use to face the manoeuvre problem. On the other hand, the process noise variance q is related to the motion model. We propose to tune the process noise adaptively.

Online hyper-parameter adaptation has been applied to video tracking with growing interest. Chai *et al.* [CNHV99] present a multiple model adaptive estimator for camera tracking. Each model characterises a possible motion type, namely fast and slow motion. Switching is done according to camera's prediction error. Yu *et al.*[YWC04] compute camera motion using an Interacting Multiple



(a) Slow motion

(b) Fast motion

Figure 6.8: Overlap between transition prior (uniform distribution) and the likelihood of the MC (cauchy distribution). Overlap indicated with a shaded region. When a fast manoeuvre occurs, the overlap is small.

Model (IMM) filter [BSB00]. Three filters, each implemented with an EKF, are used to describe different dynamics: a general model to handle arbitrary motion, a pure translation model, and a pure rotation one. The IMM filter provides a mechanism to compute the most probable model upon filter likelihoods. In this way, the limitations of a single motion model can be overcome. Although multiple models may seem an attractive solution, several works have identified their main drawback. When the state space is large, the number of necessary models becomes intractable and their quantisation must be fine enough in order to obtain good accuracy [Ich02]. Ichimura [Ich02] and, more recently, Xu and Li [XL06] show the advantages of tuning the hyper-parameters with a single motion model. Both have employed online tuning for 2D visual tracking purposes. Ichimura presents an adaptive estimator that considers the hyper-parameters as part of the state vector. Whilst providing good results, this technique adds complexity to the filter and moves the problem to the hyper-hyper-parameters that govern change in hyper-parameters. Xu and Li present a simpler adaptation algorithm that calculates a similarity of predictions between frames and updates the hyper-parameters accordingly. As described next, the adaptation method presented here is closer to this technique but brings a novel treatment of tuning in 3D camera tracking environments.

As said before, there are six DoF, three for orientation and three for position. Practice demonstrates that motion changes do not necessarily affect all axes in the same manner. Contrary to the adaptive estimators cited before, we propose a tuning that considers each degree of freedom independently. The process variance for any arbitrary axis $\mathbf{q}(i)$ of the process noise $\mathbf{q}$ is tuned according to the weighted distance from the current corrected mean state $\widehat{\mathbf{T}}_k(i)$ to that in the previous frame $\widehat{\mathbf{T}}_{k-1}(i)$ of the corresponding axis i

$$\varphi_i = \frac{[\widehat{\mathbf{T}}_k(i) - \widehat{\mathbf{T}}_{k-1}(i)]^2}{\mathbf{q}_k(i)^2} + \Delta_{min}$$

$$\mathbf{q}_{k+1}(i) = max\big(\mathbf{q}_k(i) \cdot min\,(\varphi_i; \Delta_{max})\ ;\ \widetilde{\mathbf{q}}_{min}(i)\big), \tag{6.17}$$

where Δ_{min} and Δ_{max} are the minimal and maximal variations, respectively, and $\widetilde{\mathbf{q}}_{min}$ is the lower bound for the hyper-parameter of the corresponding axis. A lower bound is needed to recover from stationary periods. During this periods the noise decays to very low values. When the camera moves again, tracking would fail if a minimal value is not assured. On the other hand, it is also necessary to limit the maximal variation for stability reasons.

This method permits a large dynamic range for the variance of each axis as it uses the current $\mathbf{q}_k$ to calculate the future value. In addition, it does not add complexity to the filter state vector as in [Ich02] nor to the system by means of multiple model estimation as the system proposed in [CNHV99] and [YWC04].

6.3 Experiments

This section is devoted to the assessment of the camera tracking framework proposed. Two sorts of experiments are performed. Firstly, an evaluation of the enhancement achieved by the fusion is conducted. The framework is confronted to partial and complete occlusions of the marker and to illumination changes. Secondly, experiments are concentrated on the behaviour of the system in general. In particular, the feature point recognition method, the mapping of new points and the dynamic tuning of the filter are tested. Before deepening into those experiments, the methodology is explained.

6.3.1 Methodology

Test set

The test set used for the experimentation consists of four video sequences with a resolution of 320x240 pixels. These sequences are named Renens, Desktop, Scrat and Shake, and are described hereafter.

The *Renens* sequence is generated by moving a virtual camera over a flat surface. This surface is textured with the picture shown in Figure 6.9. The centre of the image has a marker. The centre of the marker is made coincide with the virtual world coordinates $(0,0,0)$. This is set in order to make the coordinate system of the virtual camera coincide with the coordinate system imposed in our framework. In order to generate a realistic synthetic sequence, we previously store the motion estimated by ARToolkit [ART07] while moving a real marker in front of the camera. This motion is used to move the virtual camera in three dimensions. More particularly, the camera describes rotations in a range of 90 degrees about the Z axis and smaller ones $(< 20^\circ)$ about the X and Y axes.

The *Desktop* sequence is recorded with a hand-held camera performing a smooth motion. The marker is covered partially during two periods. More concretely, between frames 182 and 246, and between frames 345 and 448. Also, the marker escapes the FoV between frames 655 and 744.

The *Scrat* sequence is also recorded with a hand-held camera but the motion has abrupt changes in X, Y and Z axes. More concretely, motion changes from ~ 10 to ~ 250 mm/s. In this sequence, the marker is visible all along the sequence. Although the exact motion of the camera is unknown, it is possible to obtain a reference of its motion by storing the estimate of the MC. We will consider this estimate as the ground truth.

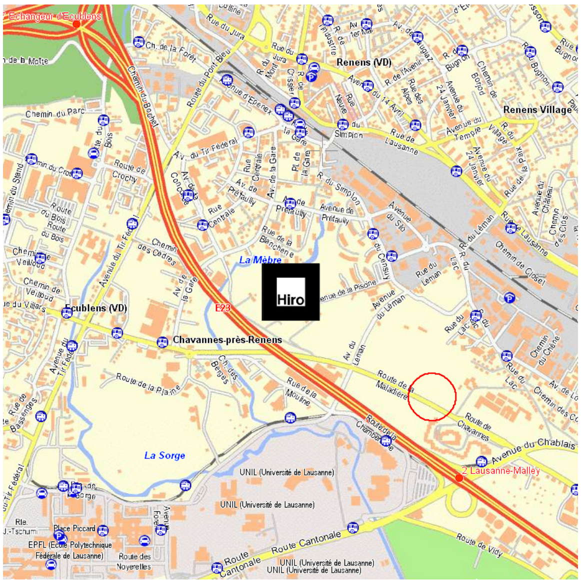


Figure 6.9: Image used to texture the virtual flat surface of the Renens sequence.

The *Shake* sequence is also recorded with a hand-held camera. In this case, the camera describes erratic motion, manoeuvres and rotations. Moreover, the marker escapes partially the FoV several times and is occluded manually.

Table 6.1 summarises the properties of each video sequence used. Figure 6.10 shows several snapshots of the test sequences.

Sequence name	Type	Number of frames	Particularities	Ground truth available?
Renens	Synthetic	492	Rotations.	Yes
Desktop	Real	1168	Occlusions of the marker.	No
Scrat	Real	591	Manoeuvres.	From MC
Shake	Real	652	Erratic motion, manoeuvres, rotations, and occlusions.	No

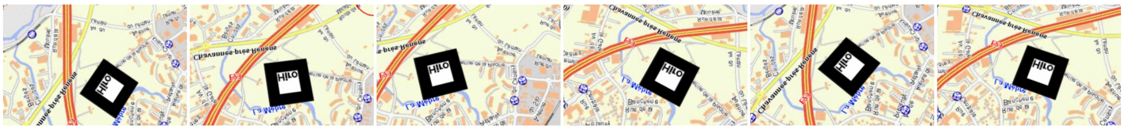
Table 6.1: Properties of the test set of video sequences.

Compared techniques

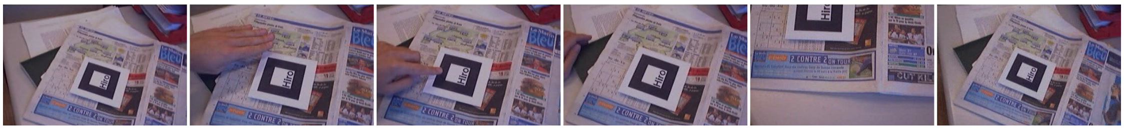
The techniques listed below are compared to our framework.

ARToolkit Camera tracker based on the ARToolkit [ART07]. It uses the detection of the marker as described in Section 6.2.2 and is equivalent to the MC.

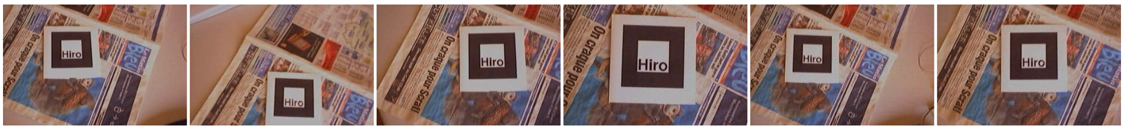
FPC-tracker Camera tracker relying only on feature points. This tracker works as if the particle filter of our framework is only updated with the FPC. In order to have a fair comparison, it



(a) Renens.



(b) Desktop.



(c) Scrat.

(d) Shake.

Figure 6.10: Snapshots of the test sequences.

is initialised with the first detection of the marker and the four corners are added as initial feature points. Note that it uses the RDTM method for feature point recognition.

PuCa-tracker Camera tracker based on the framework proposed by Pupilli and Calway [PC05]. Recalling from Section 5.3.1, p.71, this is a TDA using a particle filter updated with feature points' localisation. More specifically, the update relies on template matching using the NCC (see Equation (2.2), p.11). Hence, the descriptors contain one version of the template extracted at feature point detection.

$$F = \left\{ [\mathbf{P}, \mathbf{X}, \sigma_{\mathbf{X}}]_j \quad j = 1, \ldots, L \right\}, \tag{6.18}$$

The main difference with respect to our FPC likelihood model is that it consists in a count of inliers, provided that the correlation score is beyond the threshold. Indeed, the weight of the particle n is proportional to the number of elements ($|.|$) in the subsets S_j^n (see Equation (6.15), p.96)

$$w^n = exp\left(\sum_{j=1}^{L} C_j^n \right) \tag{6.19}$$

$$C_j^n = \begin{cases} 1 & \left|S_j^n\right| > 0 \\ 0 & \text{otherwise}, \end{cases} \tag{6.20}$$

where L is the number of feature points. Another difference with our framework and hence, with the FPC-tracker, is that it does not have dynamic tuning of the process' variance. As for the FPC-tracker, the PuCa-tracker is initialised with the first detection of the marker and the four corners are added as initial feature points.

The following parameters are fixed for all the tests conducted and filter-based techniques compared. They are chosen upon experimentation.

- The number of particles M in the particle filter of the camera is 1000.
- The transition model described in Section 6.2.1 is used.
- The output used is the corrected mean state $\widehat{\mathbf{T}}_k$ specified in Equation (6.5), p.85.

For those techniques using feature points:

- The threshold th_{corr} (Equation (6.14)) is fixed to 0.7.
- Unless stated otherwise, feature points are mapped with our method. The number of particles in the depth estimation is 100. The variances of the propagation of particles in the depth estimation are fixed to $\sigma_e^2 = 5$ and $\sigma_m^2 = 0.5$ (see Equation (6.11)), respectively.

In particular, for the FPC-tracker and our framework:

- The number of bins of the RDTM descriptor of the feature points N is 20.
- The variation bounds Δ_{min} and Δ_{max} of the dynamic tuning of the filter are fixed to 0.5 and 2, respectively.

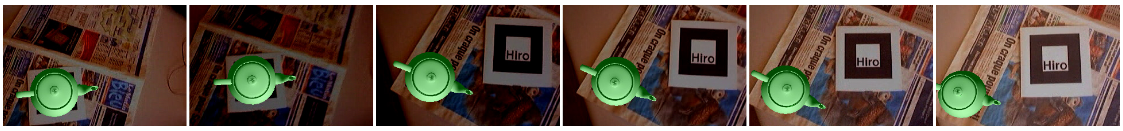
Evaluation criteria

The evaluation criteria depends on the availability of a ground truth. Indeed, if the ground truth is available it is possible to obtain quantitative results of the performance of our and other trackers. The quantitative measure used here is the Root Mean Square Error (RMSE) computed on each axis of a camera state vector T

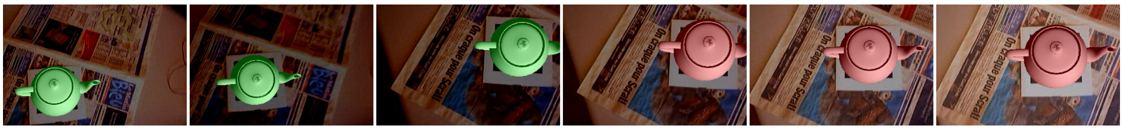
$$RMSE(\mathbf{T}_1(i), \mathbf{T}_2(i)) = \sqrt{\frac{1}{V} \sum_{k=1}^{V} \|\mathbf{T}_{1,k}(i) - \mathbf{T}_{2,k}(i)\|^2}, \tag{6.21}$$

where i indexes the axes and V is the number of video frames in the sequence. The error for the translation axes is measured in millimetres, whereas the error for the rotation is measured in degrees. Note that the rotation is expressed in Euler angles converted from the rotation quaternion. The mean corrected estimate $\widehat{\mathbf{T}}$ (see Equation (6.5), p.85) is the output evaluated for the filter-based trackers (Pupilli, FPC-tracker, and our fusion) and the MC is the output for ARToolkit.

In the opposite case where the ground truth is not available, a qualitative measure is used. When the camera position with respect to the world coordinate frame is known, it is possible to add virtual objects at a 3D position in the world coordinate space. As described in the previous chapter, this is generally known as Augmented Reality (AR). If the alignment between a virtual object and the real scene is fixed, the object should move accordingly to the cameras motion as if it was placed in the real world. A qualitative measure is found by observing how static a fixed virtual object is with respect to the real world. In our experiments, a virtual teapot is always added on top of the marker. In this way, this qualitative measure can always be analysed regardless of the availability of the ground truth. Figure 6.11 shows two examples of a correct and an incorrect alignment of a virtual teapot covering the marker.



(a) The virtual teapot starts aligned, then the tracker fails and the alignment is incorrect.



(b) The virtual teapot is correctly aligned all along the sequence.

Figure 6.11: Snapshots every 15 frames of an augmented video sequence. The alignment of the teapot is a qualitative measure of the accuracy of the camera tracker.

For some tests, the evolution of the estimate vector of the tested tracker(s) is shown. This complements the quantitative or qualitative evaluation. Computing the RMSE or showing the evolution curves have some assessment limitations. The reason being that only the output of each tracker is considered. Another possibility would be to also analyse the evolution of the likelihood distribution of these filters. The uncertainty in an estimate is related to the covariance of a filter's state. In some cases, the tracker is lost but the corrected estimate might coincide with the ground truth by chance. In these cases, the likelihood is sparse, showing high uncertainty. However, it is already possible to detect if a tracker is lost by observing the evolution of the corrected estimate. Therefore, we do not show the evolution of the distributions to improve readability.

6.3.2 Evaluation of the combination

This section explains several tests conducted in order to assess the improvements brought by the combination of approaches and cues. Recalling from the introduction, the framework is designed as a combination at two levels, namely the approach level and the measurement level. At the approach level, the original trackers that are being combined are: a marker-based (BUA) and a feature point-based Bayesian tracker (TDA). Our goal with this assessment is to prove that the fusion framework proposed is capable of solving individual failure modes of these trackers. Moreover, our goal is to obtain a synergy by taking advantage of the individual strengths.

Occlusions

An experiment is conducted to analyse the tracking performance in front of occlusion of the marker. As stated before, one of our goals is to cope with the loss of track of the MC when the marker is occluded. In our framework, tracking can continue by using the FPC.

Two techniques are compared in this case. On the one hand, ARToolkit, which is equivalent to use the MC alone. On the other hand, our framework combining MC and FPC. The Desktop sequence is used to evaluate the effect of occlusions.

Figure 6.12 shows one instance of the evolution of the camera pose. As it can be seen, while the MC is available (green dots), the output of our framework takes advantage of this cue. During marker occlusions, indicated with shaded regions, the system keeps tracking the camera, whereas ARToolkit produces no output. During these periods, the qualitative evaluation with the aligned

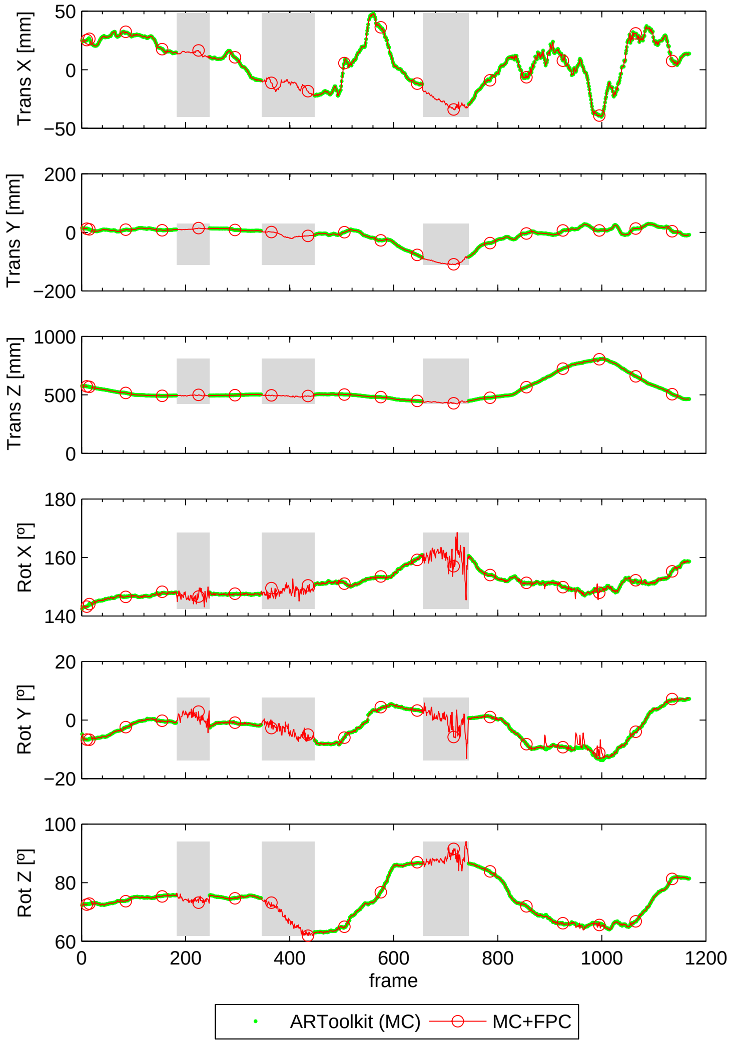
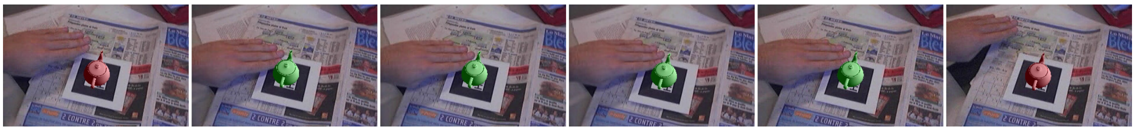
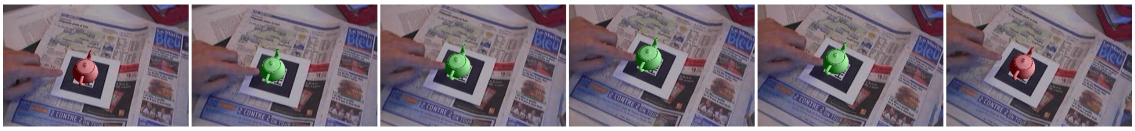


Figure 6.12: Experiment with occlusions. Comparison between ARToolkit (MC) (green dots) and our framework (red line) facing occlusions. Translation and rotation in X,Y and Z axes (Desktop sequence). Shaded regions indicate the periods of marker occlusion.

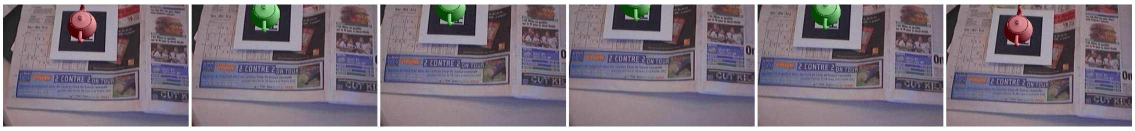
virtual teapot can be used. Snapshots from several frames of the augmented sequence are shown in Figure 6.13.



(a) Manual occlusion. Snapshots around frame 200.



(b) Manual occlusion. Snapshots around frame 400.



(c) The marker is escaping the field of view. Snapshots around frame 700.

Figure 6.13: Experiment with occlusions. A virtual teapot is placed on the marker to show correct alignment. When the teapot is red, the framework uses the MC, whereas when it is green, the framework relies on the FPC.

Illumination changes

Another experiment is conducted to analyse the tracking performance in front of illumination changes. As stated earlier in Section 6.2.2, the MC uses a fixed threshold for binarisation and further marker identification. When the illumination changes considerably, the contrast becomes too low in the contour of the marker and the detection algorithm fails. On the other hand, the FPC is illumination-invariant because the templates are normalised with respect to their luminance means in the NCC computation of the RDTM method.

The Renens sequence is used in this test. In order to simulate illumination changes, an offset varying between -100 and $+100$ is added to the RGB channels of the video. Two techniques are compared to the ground truth. On the one hand, the estimate provided by ARToolkit (same as the MC). On the other hand, the output of our proposed fusion of MC and FPC.

Figure 6.14 shows one instance of the absolute error of the camera pose. Results show that the MC fails completely to detect the marker when the offset is below -50 (approximately). When the offset is above $+50$ the marker is detected but the pose provided by the MC deviates from the ground truth. This is especially visible for the translation in the Z axis. The reason for the inaccuracy is that the definition of the contour of the marker changes with the illumination offset. As for our fusion, it can be observed that it succeeds in providing a continuous estimate regardless of the illumination changes. The RMSE achieved by the compared techniques is given in Table 6.2. Note that the RMSE with ARToolkit is only computed at those frames where there is an output.

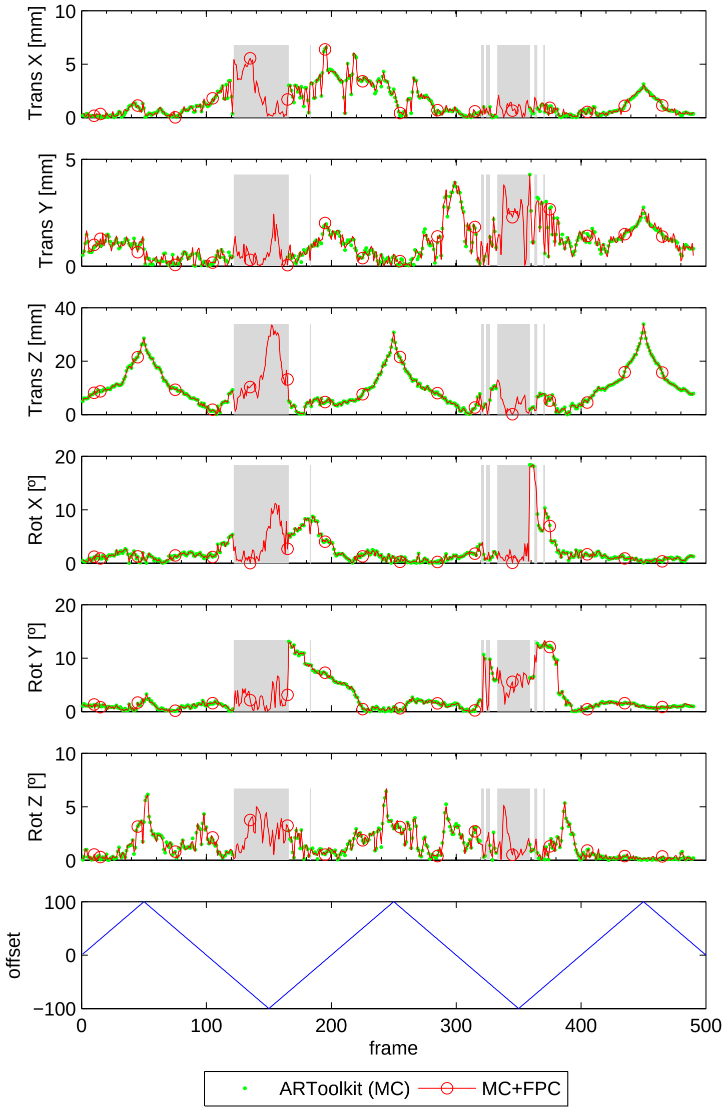


Figure 6.14: Experiment with illumination changes. Comparison between ARToolkit (MC) and the fusion (MC+FPC). Absolute error of the translation and rotation in X,Y and Z axes (Renens sequence). Shaded regions indicate the periods where the MC produces no output.

6.3.3 Evaluation of the system

The combination framework is designed to fuse two cues. Nonetheless, the framework is composed also of several particular assets that increase the robustness. These assets are the recognition method for feature points, the method to initialise new points, and the dynamic tuning of the filter to face abrupt motion changes. Several tests are conducted in order to evaluate the performance of each one of these assets individually.

	Translation [mm]			Rotation [deg]		
	X	Y	Z	X	Y	Z
ARToolkit (MC)*	1.14	0.94	12.04	3.13	4.05	0.43
MC + FPC	1.12	0.94	12.36	3.44	4.13	0.52

Table 6.2: RMSE achieved by the fusion and by ARToolkit for the Renens sequence with additive illumination changes. (*) Note that the RMSE for ARToolkit is only computed for those frames where there is an output.

RDTM for feature point-based camera tracking

In Chapter 3, the RDTM method to recognise regions is described. This method is tailored to detect the rotation that the template of a point has undergone. In that chapter, experiments have shown the accuracy of the method on several images rotated over the perpendicular axis (2D rotations). We want to evaluate here the improvement brought by the RDTM when compared to a simpler but commonly used method.

Two feature point matching techniques are compared namely, the PuCa-tracker and the FPC-tracker. In the PuCa-tracker, the recognition is performed with the NCC of the templates. As discussed in Section 5.3.2, this is a common approach for feature point-based camera tracking. In the FPC-tracker, matching of feature points is done with our RDTM method. The experiment is conducted with the Renens sequence.

Figure 6.15 shows one instance of the absolute error of each axis of the compared techniques. Matching with NCC fails as soon as a large rotation around the Z axis occurs (around frame 50). As a consequence, the PuCa-tracker looses all references and starts to drift. On the other hand, the rotation-discriminative method allows a continuous track of the feature points and hence accurate camera pose estimation. Indeed, the RMSE achieved for the Z axis is very low: 0.79 degrees.

Adding new feature points

The method proposed for mapping is evaluated here. The test consists in comparing the 3D position estimated for each feature point when this is added to the set F.

Our method is compared to the method of Pupilli and Calway [PC05]. As described earlier in Section 6.2.4, this method sets particles uniformly distributed along the depth line at the initial frame and then updates particles upon triangulation. The purpose of this comparison is to show the improvement brought by the propagation and resampling scheme proposed. The decision rules that consider that the distribution of the estimated position has converged are not specified in [PC05]. Therefore we use our own decision rules. For this experiment, the Renens sequence is used. The estimation for each point is compared to its 3D ground truth position. This ground truth is available because the virtual world created in the Renens sequence is especially tailored for this purpose. Indeed, the marker inside the image (see Figure 6.9) measures exactly 80x80 square pixels. At the same time, the description of the marker used by the system state that the size of the marker is 80x80 mm^2. In this way, any point in the image has a known X and Y position. Since the image is

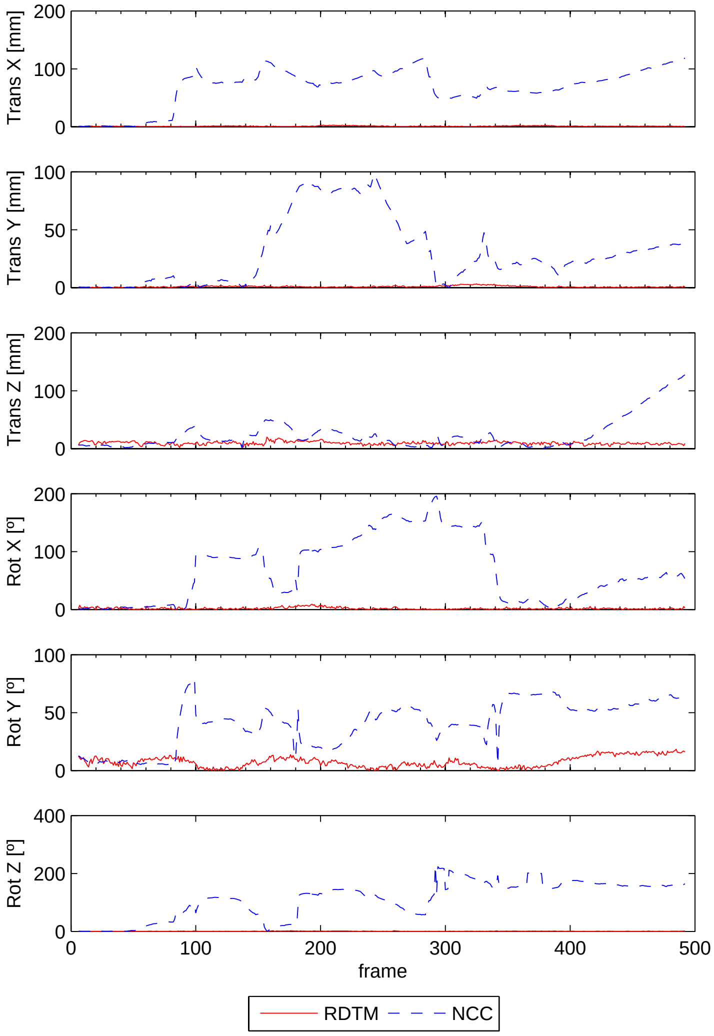


Figure 6.15: Experiment with different feature point recognition methods. Comparison between NCC and RDTM. Absolute error of the translation and rotation in X,Y and Z axes (Renens sequence).

placed $Z = 0$ all points must have this depth.

The evaluation criteria is the Euclidean distance between the ground truth and the corresponding estimated position. This distance is averaged for all the added points during several runs. In order to give a fair comparison, the same camera pose distribution is used. The filter updated with the MC is used as camera pose distribution.

The resulting average error is 31.35 mm (standard deviation 34.47 mm) for the uniform distribution method and 26.81 mm (standard deviation 18.33 mm) for the proposed method. Table 6.3 shows the variance $\sigma^2_{\mathbf{X}}$ of the descriptor of the feature point added to the set F. As described earlier, this variance is composed of the diagonal values of the covariance matrix of the depth distribution at the frame where the this is considered to have enough convergence. As it can be observed these results show that our method is in average more accurate. This test has one main limitation namely, the sequence used and consequently the motion of the camera. Indeed, this motion plays a substantial role in the accuracy of the estimation. Large translations in the Z axis or rotations around the X and Y axes make stronger constrains during the triangulation process. Therefore, this test does not give the baseline accuracy that these methods can achieve.

	$\sigma^2_{\mathbf{X}}$		
	X	Y	Z
Initial uniform distribution (based on [PC05])	3.44E-03	6.54E-03	1.00E-01
Proposed	7.29E-06	3.00E-05	1.56E-04

Table 6.3: Average values of each axis of the variance vector $\sigma^2_{\mathbf{X}} = (X, Y, Z)$.

Response to manoeuvres

The dynamic tuning of the filter is designed to adapt to different types of motion. Indeed, a motion model is not expected to perform accurately when it does not fit the current motion. In order to experience different sorts of motion, the Scrat sequence is employed.

The performance of our dynamic tuning is compared quantitatively to the framework without dynamic tuning and with a similar method. Among the similar techniques described in Section 6.2.6, we have selected the closest in spirit, which is the method presented by Xu and Li [XL06]. The authors propose an adaptation method for 2DoF that can be extended in terms of Equation (6.17), p.99 to 6DoF as follows

$$\varphi = exp\left(-0.5\sum_i \frac{[\widehat{\mathbf{T}}_k(i) - \widehat{\mathbf{T}}_{k-1}(i)]^2}{\widetilde{\mathbf{q}}(i)^2}\right)$$
$$\mathbf{q}_{k+1}(i) = max\left(min\left(\widetilde{\mathbf{q}}(i)\cdot\sqrt{1/\varphi}; \widetilde{q}_{max}(i)\right); \widetilde{\mathbf{q}}_{min}(i)\right), \tag{6.22}$$

where $\widetilde{\mathbf{q}}_{max}$ and $\widetilde{\mathbf{q}}_{min}$ are the upper and lower bound vectors and $\widetilde{\mathbf{q}}(i)$ is the nominal value for axis i. Note that $\widetilde{\mathbf{q}}(i)$, $\widetilde{\mathbf{q}}_{max}(i)$ and $\widetilde{\mathbf{q}}_{min}(i)$ are fixed off-line to a single value for the three rotation axes and another single value for the three translation axes. Also remarkable is the fact that the prediction error in one axis affects all other hyper-parameters, as φ is unique for all the axes. Consequently, the model of process noise in one axis may grow even when the real error in that axis is small. Moreover, the dynamic range of the variance of each axis is limited off-line whereas our proposed dynamic tuning has only a lower bound to insure recovery from almost static motion.

In order to compare the proposed method to this approximation of Xu and Li's approach, the upper bound is fixed to the maximal value achieved by our technique in the Scrat sequence. The lower bound is the same for both. The variances in our method are initialised with the nominal values $\widetilde{\mathbf{q}}(i)$. The variation bounds Δ_{min} and Δ_{max} are fixed to 0.5 and 2, respectively.

The most relevant motion changes are for the translation in the X,Y, and Z directions. Table 6.4 presents the RMSE calculated for these three axes. These quantitative results show the superiority of the proposed adaptive tuning method.

	Translation [mm]		
	X	Y	Z
Without dynamic tuning	1.24	1.67	1.45
Xu and Li [XL06] (extension to 6 DoF)	0.98	0.28	0.52
Proposed dynamic tuning	0.46	0.16	0.13

Table 6.4: RMSE without adaptive tuning, the extension of [XL06] and the proposed method.

6.3.4 Comparison with other techniques

The experiments conducted in the previous sections have evaluated the performance of our tracker. In some cases, the performance has been compared to individual trackers. The reason being that each comparison is tailored to assess a certain aspect of the tracking. Another experiment is conducted here to observe the response of all the techniques described in Section 6.3.1 in front of various problems at the same time. More precisely, erratic motion, manoeuvres, rotations, and occlusions. For this test, the Shake sequence has been used.

Figure 6.16 shows the evolution of the estimate for each tracker. The RMSE achieved by each one of the compared techniques is given in Table 6.5. Since there is no ground truth available, the output of ARToolkit (MC) is used as reference due to its high accuracy. Consequently, the RMSE is computed considering only those frames where the MC is available.

	Translation [mm]			Rotation [deg]		
	X	Y	Z	X	Y	Z
PuCa-tracker	42.53	30.15	60.02	105.34	15.45	15.95
FPC-tracker	22.55	36.58	41.17	15.33	9.41	10.73
MC+FPC	11.05	15.63	4.20	0.24	1.66	3.11

Table 6.5: RMSE between each tracker's output and the MC considering only those frames where the MC is available.

These results denote several aspects of the compared techniques. First of all, the BUA ARToolkit is unable to provide an output when the marker is occluded (shaded regions) which has already been discussed previously. The other techniques provide a continuous output thanks to their Bayesian tracking approach. Second, TDAs such as PuCa and FPC-tracker are able to handle erratic motion occurring around frame 200. This is especially visible in the translation on the *Y* axis.

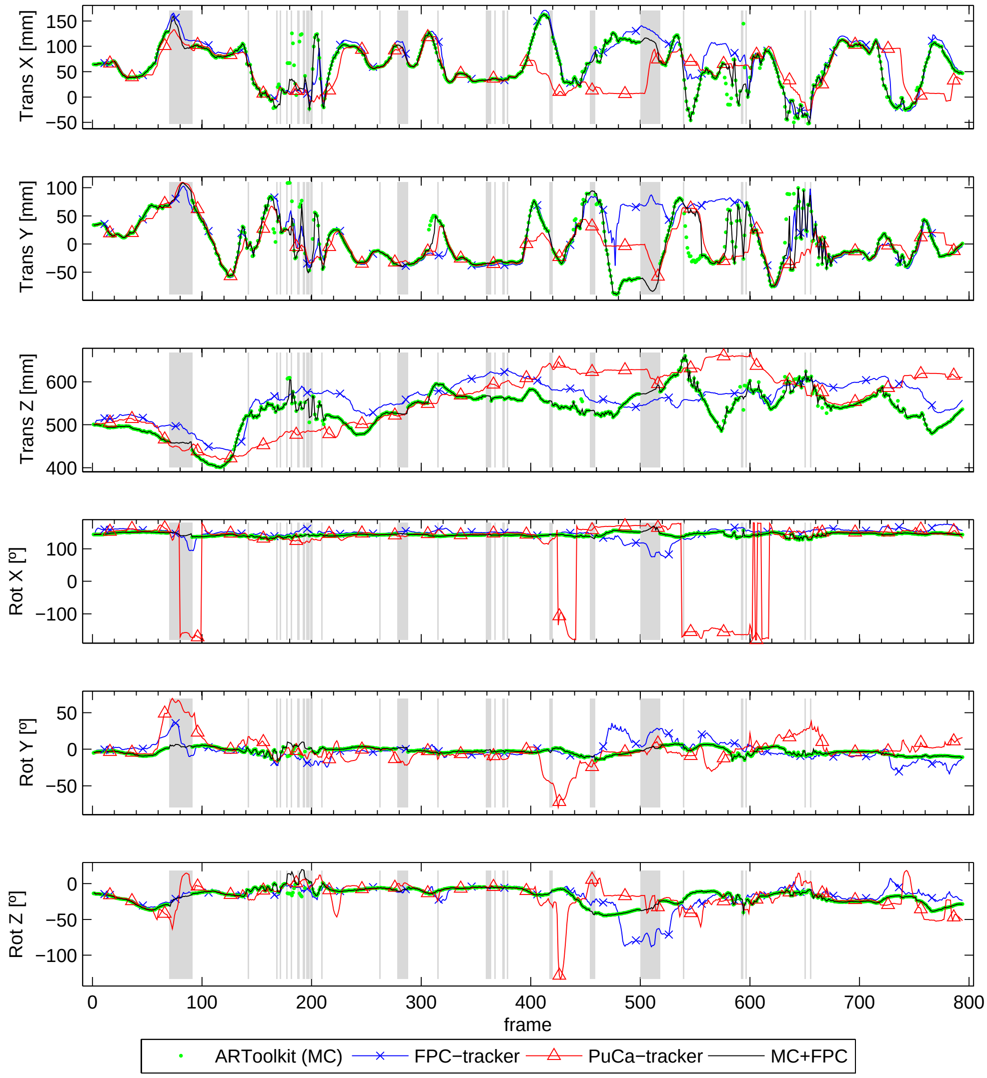


Figure 6.16: Experiment with erratic motion, manoeuvres, rotations and occlusions. Comparison between ARToolkit (MC), the FPC-tracker, the PuCa-tracker and our proposed framework (MC+FPC). Translation and rotation in X,Y and Z axes (Shake sequence). Shaded regions indicate the periods where the MC produces no output.

However, failure to recognise feature points either during marker occlusions (which also occlude some feature points) or fast manoeuvres produce a loss of track. This can be observed around frame 400 for the PuCa-tracker and frame 450 for the FPC-tracker in X and Y translation axes. Third, the only tracker that produces a continuous output and that is capable of recovering from the loss of track is our proposed framework. It can be deduced that relying alternatively on a BUA and a TDA compensates individual weaknesses and contributes to a higher overall robustness.

Let us now discuss the limitations of this comparison. Firstly, this comparison does not pretend to be generic but tailored to the environment defined in the introduction of this chapter. Much lower errors could be achieved by any of the compared techniques including the ones we propose, FPC-tracker and MC+FPC, in smoother motion conditions. Secondly, complete frameworks are not compared. PuCa and FPC-trackers rely on feature points from a TDA. Therefore, an automatic initialisation process is not viable. Pupilli and Calway [PC05] propose an initialisation consisting in holding the camera parallel to a black rectangle of known dimensions. The four corners of the rectangle are selected by hand. This gives the framework four initial feature points and the pose at frame 0. Therefore, the framework presented in [PC05] has less capabilities than the PuCa-tracker itself and also than the fusion proposed here. The same happens for the FPC-tracker as it has no automatic initialisation process on its own.

6.3.5 Discussion

Tests conducted indicate the robustness and superiority of the proposed framework. Let us recapitulate the main achievements in terms of capabilities and synergistic combination.

- Automatic initialisation and recovery from loss of track are possible thanks to the BUA implemented with the Marker Cue.
- Partial and complete occlusion of the marker are possible thanks to the TDA. Indeed, keeping a recursive estimation of the pose and mapping new feature points extends the trackable area.
- The framework is able to provide continuous and accurate tracking. This is a synergy of the accuracy of the BUA and the continuity of the TDA.
- The framework proves the viability of combining markers and feature points in a single filter-based tracker.

It is interesting to observe that our fusion relies on the MC even if its estimate is inaccurate. See for instance the results revealed by Figure 6.14 regarding the translation in the Z axis. Indeed, the framework holds strongly on to the MC and considers the FPC as a fallback cue. This could be identified as a drawback of the proposed fusion algorithm. A possible solution would be to compute both cues at each frame and combine their outputs. Unfortunately, this would increase the computational load of the system. Consequently, we still believe that the robustness shown by the MC in most situations justifies the likelihood switching method proposed.

6.4 Summary

This chapter has presented a camera tracking framework. It fuses Top-Down and Bottom-Up Approaches combining two sorts of cues, the Marker Cue and the Feature Point Cue. The system is based on a particle filter updated with two different likelihoods depending on the detection of a marker in the scene. When the marker is detected, the estimation provided by the MC is used, whereas in the opposite case, the filter is updated using the FPC. Furthermore, the Rotation-Discriminative Template Matching described in the first part of this thesis is integrated in our FPC.

In order to extend the trackable area, we introduce a method to map natural feature points. This method estimates the 3D position of previously unknown corners or edges in the scene. In addition, an algorithm to dynamically tune the motion model of the filter is presented. This algorithm permits to face tracking in front of manoeuvres of the camera.

An experimental assessment of the framework demonstrates that our two goals are achieved. Firstly, the viability of combining two approaches from the same modality, namely video-based. Secondly, a system that is capable of automatically initialising, recovering from loss of track and occluding the prepared reference. The MC is responsible for the first two capabilities. The FPC resolves partial and complete occlusion of the marker (reference) by relying on natural feature points.

The assessment is complemented with specific experiments to showcase the benefits of the proposed system. Indeed, the mapping and the dynamic tuning of the filter are tested and the improvement is shown. Moreover, the RDTM shows higher performance when compared to NCC template matching for feature point-based camera tracking.

Applications

7

Localising a camera in three dimensions with respect to a reference in a real scene is necessary for a large number of applications. This chapter is dedicated to describe developed and potential applications of the camera tracking framework proposed. We focus on those applications that take advantage of the performance and capabilities of our combination of approaches.

Firstly, applications related to human-machine interfaces and Augmented Reality (AR) environments are described in Section 7.2. In particular, we explain the results of our research collaboration with University of art and design Lausanne (ECAL) and the transfer of technology to industry. Secondly, potential applications related to robotics are dealt with in Section 7.3.

7.1 Introduction

Transferring the fundamental research developed in this thesis into real applications, demands a certain shift in the point of view. The concerns of a potential exploiter of our system often deviate from pure performance evaluation common in fundamental research. Indeed, final users concentrate on serviceability factors. In order to obtain a better usability assessment, we have contacted users with quite a large range of ages and with very different profiles such as interaction designers, architects and journalists, among others. This multidisciplinary exchange of ideas has taken place along the development of our system and has permitted to reshape particular aspects of its design.

The question that may arise is: why would an application take advantage of our framework? As discussed in the introduction of the previous chapter, it is rare to find a tracking framework that fulfills at the same time automatic initialisation, re-initialisation after loss-of-track, and tracking beyond a priori known references. However, our framework has proved to fulfil these capabilities (see Section 6.3.5). Therefore, the proposed tracker benefits any application that requires them. Of course, this benefit can be achieved provided that the environment covers the properties defined in

Section 6.1. Firstly, textured regions must be present. Secondly, at least one marker must be placed before tracking starts. Fortunately, such environment can be found in a vast variety of scenarios as explained below.

7.2 Human-machine interfaces and augmented reality

Interacting with the environment by taking advantage of camera trackers has arised a growing interest in the past decade [LF05][AME05]. Examples of such interactions are collaborative tasks and tangible interfaces, among others. In these environments, a visual reference is often used. As a consequence this reference could probably be substituted by a marker such as the ones used in our framework. However, a known problem of interfacing with markers is that these are often occluded by user's arms or hands (e.g., a shared design table). Our framework could deal with this by switching to the feature point-based cue, providing continuous tracking.

7.2.1 Augmented machinery inspection

Several applications of Augmented Reality (AR) have been described in literature. For instance, architecture [WFM$^+$96], telerobotics [MRG95] and historical heritage [SK01], among others. More examples can be found in the surveys by Azuma *et al.* [Azu97][ABB$^+$01]. Among the various examples identified in these surveys, we recognise machinery repairing or inspection, and medical environments as a potential application of our framework.

Repairing of industrial machinery is a complex task. Assisting this task with augmented annotation of the machine in front of the technician could ease this work. Caudell and Minzell pioneer this area with a demonstrator developed for Boeing [CM92]. The goal is to address the complexity of aircraft manufacture and assembly. Indeed, this process includes many manual processes which change from one design to another. The authors propose an AR system as a solution to this problem. Workers wear a Head Mounted Display (HMD) and the diagrams of the machinery are overlaid on specific wires and connectors. More recently, Navab [Nav04] describes the CyliCon software which uses existing industrial drawings and floor plans to produce the augmentation of a waste water plant. In this way, information already available about an industrial plant is exploited for monitoring and control process. The user interface allows the worker to have in-place easy access to engineering and maintenance data. Figure 7.1 depicts different augmented views using images and floor plant drawings. Another example of industrial augmentation is presented by Zhang *et al.*[ZGN01]. Their framework is composed of a mobile computer with a camera connected via wireless network to a server. The system is applied to a large environment inside an industrial plant. Markers are used as a map of the plant to guide the user and also to give extra information of the surroundings of the user (e.g. the description of the machine in front). In this system, the user can browse the location's relevant information.

Computer-aided surgery and medical imaging assisted by augmented reality are areas recently increasing the attention of researchers [BFO92, SLG$^+$96]. Bajura *et al.* [BFO92] pioneered this area presenting an augmented breast surgery using a hybrid magnetic-video tracker (see Sections 4.1.2, 45, and 5.4, 75). Figure 7.2 shows a snapshot of the augmented surgery.

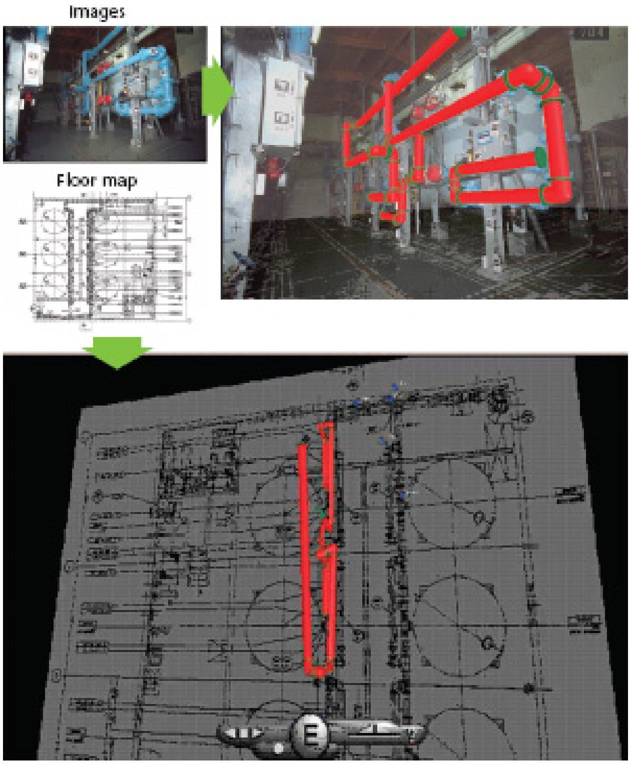


Figure 7.1: Waste water plan augmentation [Nav04].

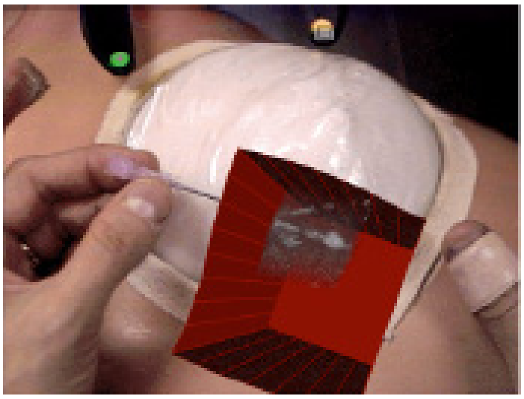


Figure 7.2: Image-guided breast biopsy using augmented reality [SLG+96].

As it can be deduced, our framework has a potential to be used in these kind of environments. In the experiments described in Section 6.3, we have shown the correct alignment of virtual objects in real scenes using our tracking framework. This showcases that augmented reality can be achieved.

We describe hereafter a specific application using our camera tracker. Taking the specific example of repairing, one notices that machinery inspection relying on markers can be awkward if the user has to consciously avoid occluding markers. That is the reason why continuity provided by our framework would improve the user's experience. A toy example of augmented machinery inspection using our system is presented next. More precisely, the back of a personal computer (PC) is inspected. Firstly, a marker is attached to it. Then, the relative position of some parts with respect to the centre of the marker are annotated. This simple procedure takes less than 5 minutes and requires only tape and a ruler. The set of positions is written on an external text file with the following content:

```
5
-75   15 -10 Printer
-85  155   0 Power
-80  -15 -10 Mouse
-150 -35 -10 Keyboard
-20 -115 -10 Screen
```

where the first figure indicates the number of elements in the set, and the other lines indicate the 3D position in millimetres and the corresponding label. The next step is simply to take a camera and observe the back of the PC. Some snapshots of an augmented sequence are shown in Figure 7.3. This example also shows how to provide handy 3D information without the need for a CAD model, just a few straightforward annotations.

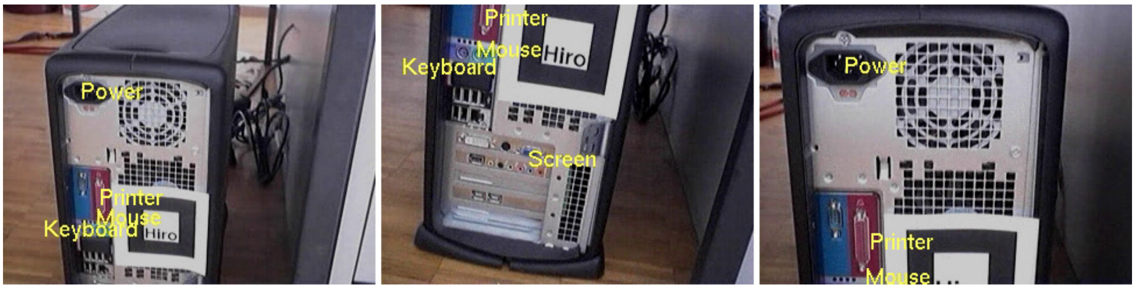


Figure 7.3: Machinery inspection assisted with augmented reality. The plugs of a Personal Computer are indicated. Augmented information is provided despite the occlusion of the marker.

7.2.2 Collaboration with University of art and design Lausanne (ECAL)

Research collaboration with the University of art and design Lausanne (Ecole Cantonale d'Art de Lausanne, ECAL) started with our group in the form of a workshop with their students. This first step revealed the potential for further collaboration between our respective research groups. At a higher level, the partnership between ECAL and the Swiss Federal Institute of Technology (EPFL) originated a research project called *Variable environment / mobility, interaction city and crossovers* [EE07]. This project analyses aspects related to spaces, the city, the interaction and the crossovers among different disciplines. This research is also developed in partnership with other designers, architects, and universities outside of Switzerland.

Our particular contribution within this collaboration is twofold. Firstly, we were involved in the lecture given at ECAL in the faculty of Media & Interaction Design. In addition to that, we have also jointly developed an interaction demonstrator called AiRToolkit. These contributions to the ECAL-EPFL partnership are explained hereafter.

Workshop with students of Media & Interaction Design

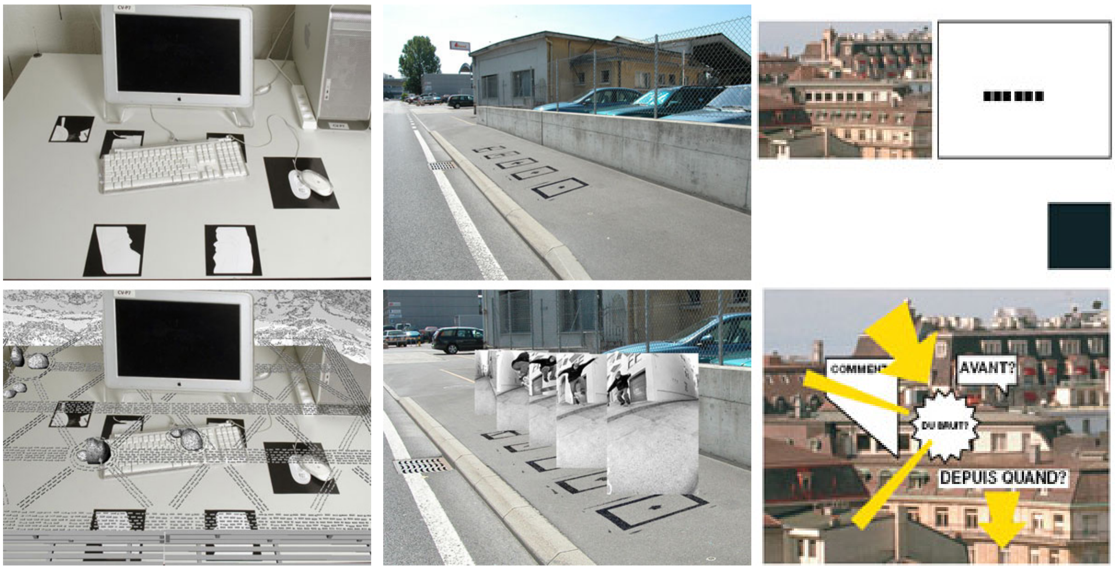
In Spring 2005, the students of Media & Interaction Design developed a semester project with the topic *Augmented reality environments*. The purpose of this project was to explore the possible electronic extension of their mobile objects (content of their backpack, wallet, pocket, etc.). First, the students defined a family of signs associated to those objects. Then, they created the 3D virtual content linked to those signs. In order to exploit the capabilities of our tracking system, signs were defined so that they could be recognised as markers by the system. At the end of the semester, a one-week workshop was carried out for an easier interaction between the designs of the students and the development of our tracking system.

Among the various projects, we highlight three of them. The *Cosmo Lab* project intended to add some privacy to the office desktop of a worker. Instead of having objects that are at sight for other colleagues, the user would employ markers and link them to virtual objects at his own discretion. Since only his system would know the relation between markers and virtual information overlaid, he would have private control over the appearance of his desktop. Another interesting project was called *Skating Wall*. This project focused on the figures accomplished by skaters at specific spots in the city. As a first step, pictures of a skater performing a certain figure were taken at the corresponding spot. After that, signs were sprayed on the floor or on a wall of an urban space. Once this two steps were finished, the signs were recognised by the camera tracking system and the pictures were overlaid in the real environment. Using several signs at a single spot in the city one could see snapshots of the whole figure. Last but not least, the *Ragorama* project attracted our attention for the special definition of signs used. This project aimed at augmenting a panoramic view of the city of Lausanne. Instead of using hand-made signs, the students analysed the recognition potential of the system. Markers were created from urban shapes such as windows or fences, among others. Figure 7.4 shows a snapshot of the camera input and the augmented video for these three student projects.

AiRToolkit

In collaboration with the ECAL and fabric | ch (studio for architecture & research), we have developed a demonstrator called AiRtoolkit (Augmented interactive Reality toolkit, [BMK$^+$07]).

AiRToolkit is an alpha-version software, intended to become freeware. It analyses the video stream of a camera in real time to detect markers. The recognition of a marker can directly affect the surrounding environment, for instance, by switching a bulb on and off. It can also augment the video content with 2D or 3D interactive data linked to reality. In this way, an augmented reality is generated. Interaction modes between the camera, marker(s) and/or inserted 3D objects may vary. They can be related to pattern visibility, distance or relative position between the camera and the marker. Each case can generate multiple results depending on the desired configuration. Figure 7.5 shows some snapshots of a possible interaction. In this interaction, when the camera approximates the marker, a light is turned on and when the camera distances from the marker, the light is turned



(a) Cosmo Lab (b) Skating Wall (c) Ragorama

Figure 7.4: Selected results of the student projects about *Augmented reality environments*.

off again. Another example is opening a web browser in our computer depending on the viewpoint from which we see a marker.

AiRToolkit can be considered as an evolution of ARToolkit [ART07]. It builds on top of our tracking framework. It adds different forms of interaction, increases the range of possible applications (exploiting 2D and 3D media), and opens to new diffusion platforms such as portable devices provided with cameras. This interactivity layer extends our camera tracking framework into a tangible interface.

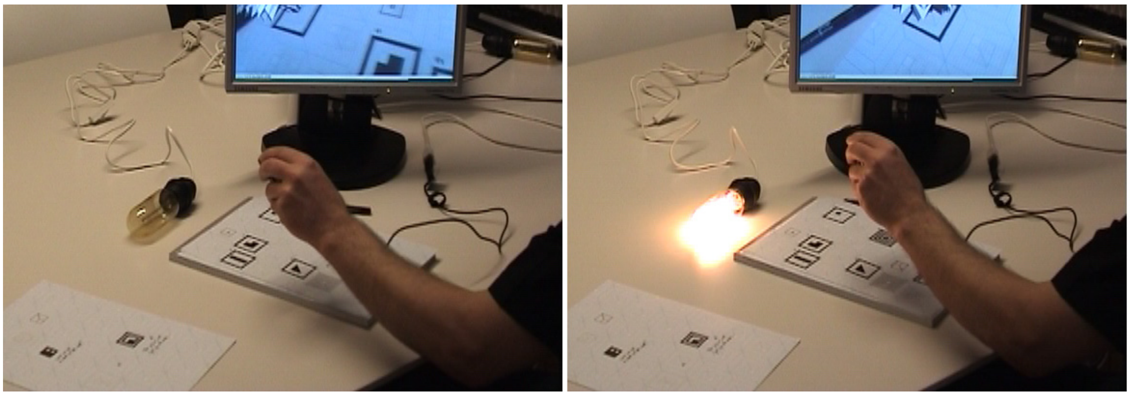
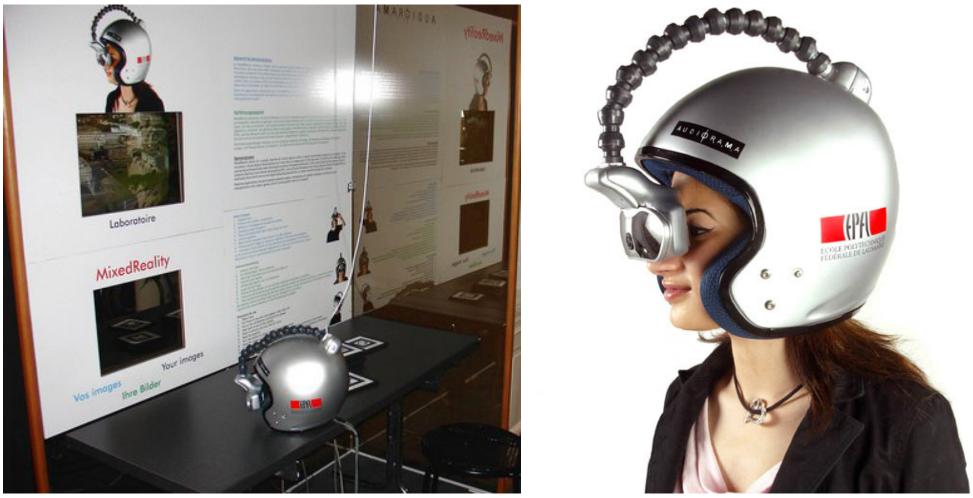


Figure 7.5: Interaction using the AiRToolkit. A bulb is turned on and off depending on the distance between the camera and a marker.

7.2.3 Transfer of technology

As a result of the advances of our research, we have been involved in a transfer of technology to industrial partners. In particular, we have developed a permanent demonstrator in the Swiss National Museum of Audiovisual and Multimedia (Audiorama) in Montreux. The goal is to showcase the possibilities of augmented reality through a game-oriented demonstrator (see Figure 7.6a). For the robustness and user-friendliness of the equipment, we developed a special helmet integrating a HMD and the video camera (see Figure 7.6b). This integration was done by the EPFL's workshop under our guidance.



(a) Demonstrator booth (b) Integrated helmet

Figure 7.6: Permanent augmented reality demonstrator in Audiorama museum.

This demonstrator has also been shown to the industrial community at two occasions, namely, in the Hannover Messe (April 2004, Hannover, Germany) and in the Communications Days (October 2005, Biel/Bienne, Switzerland). Figure 7.7 shows one of the attendees using our system and experiencing augmented reality.

7.3 Robot navigation and visual servoing

Robot navigation consists in tracking the position of a mobile robot. Among the possible sensors, robots commonly include cameras. As discussed in Section 5.3, TDAs are usually applied. An example application of robot navigation is guided museum visits [DBFT99]. This application has many challenges, mostly related to robot-human interaction. Nonetheless, ego tracking could be performed with our framework. Indeed, initialisation with a marker in a controlled environment such as a museum is completely viable. Tracking could then continue using feature points, which could be found in what is exposed (e.g., paintings). If tracking fails, other markers at different positions properly calibrated could make the system recover easily from its loss of track. As it can

Figure 7.7: Presentation of the demonstrator to industrial audience.

be seen, robot navigation in controlled indoor environments is a potential application which could perfectly benefit from our fusion approach.

Visual servoing is frequently used to control robotic arms. A camera is attached to the extreme of the arm and the intended task is assisted by visual feedback. This corrects possible errors of the mechanical trackers. Again, auto-initialisation followed by an extended tracking area beyond the reference is an interesting property than could be exploited in our case.

7.4 Summary

This chapter has discussed the applicability of the camera tracking framework developed in this thesis. The main areas of application are human-machine interfaces, augmented reality, and robotics.

In the areas of human-machine interfaces and augmented reality environments we have presented results in three ways. First, an augmented machinery inspection example is explained. We show with a toy example that our system is applicable to the wire/plug inspection of a PC. Moreover, the necessary set up is very limited and simple. Second, in the context of the collaboration with the University of art and design Lausanne (ECAL), we show the results of the interaction with designers. One of the outputs of this collaboration is the joint development of the AiRToolkit. This toolkit extends our tracking system into a tangible interface. Finally, we show a transfer of technology in the form of a demonstrator permanently exhibited in the Audiorama museum and in particular industrial fairs.

In addition, a potential application to robot navigation and visual servoing has been discussed.

Part III

Conclusions and future work

Conclusions

8

8.1 Summary of achievements

In this dissertation, we have analysed the benefits of fusing a top-down (TDA) and a bottom-up approach (BUA) to video-based camera tracking. We have investigated whether a synergy could be reached by merging the capabilities of a marker-based tracker (BUA) and a feature point-based Bayesian tracker (TDA). In order to achieve this goal, we have developed a camera tracking system. At the core of this system lies a particle filter that keeps track of the camera pose. The fusion consists in combining two sorts of cues. One cue is given by ARToolkit, called Marker Cue (MC). It provides an estimate of the camera pose with respect to the marker. The other cue is based on feature points' localisation and is called Feature Point Cue (FPC). In order to produce this cue, feature points are searched with a TDA. The 3D position of feature points in real world coordinates is known. The detection of these feature points in the image plane provides a constraint on the pose of the camera. The combination of cues consists in switching between two likelihood models. Due to the robustness of the MC, we choose to rely on it every time the marker is detected. The FPC is used as a fallback process. This happens when the detection of the marker fails due to weaknesses of the MC in front of occlusions or illumination changes, or because the marker is outside of the FoV.

The framework proposed is a low-level fusion. We have proposed a likelihood switching design which permits high flexibility of cues. In our case, very different cues are combined. Indeed, one is an estimate of the camera pose and the other is composed of positions of feature points in the current frame. At the core of this framework there is a common filter which enables continuity and time-coherent estimates. Indeed, our low-level design with a filter merging measurements avoids jumps of the final output. Conversely, a high-level fusion, where the output of the fusion tracker directly switches between the outputs of different trackers, would be prone to generate jumps when the individual estimates of the trackers are very different.

The system is provided with additional capabilities to increase robustness. Since we target hand-held camera tracking, erratic motion and manoeuvres have to be addressed. These changes in motion are unpredictable. Moreover, our experimentation indicates that each dimension of the state space can describe individual changes. Following this observation, we have proposed a dynamic tuning of the motion model that treats separately each dimension. The difference between poses at each frame are used to vary the hyper-parameters that model the errors in motion estimation. This dynamic variation outperforms related methods found in literature either in accuracy or from a system complexity point of view. In order to keep tracking the camera pose when the marker is outside of the FoV, other references of the real space are needed. We have proposed a method to map feature points in the space. This method consists in an iterative triangulation process with a particle filter representing possible 3D positions of the feature point. The method fulfills the goal of extending the trackable area beyond the marker. It also shows higher accuracy when compared to another triangulation technique of the state of the art.

Another problem faced by a camera tracker relying on feature points is accurately localising those points in the image plane. Problems such as illumination changes and viewpoint distortion have to be addressed. In particular, we have dealt with the illumination problem and with 2D rotations. We have introduced a rotation-discriminative region descriptor and an efficient rotation-discriminative method to match feature point descriptors based on the histogram of gradient orientation and on intensity information. The method is presented independently as it is applicable to other areas and not only camera tracking. Experiments show high accuracy especially around rotations quantised by the histogram. For instance, with a histogram of 16 bins, 98% of true positives are achieved for only 10 false positives using textured patches, and 91% using patches selected randomly. When the rotation is in between two quantised rotations, the results drop to 90% and 68%, respectively. Comparing to state of the art techniques, our method has similar or even higher accuracy. More relevant is that our method has much lower computational cost. For instance, when compared to OH+OCM [UK04], the speed up factor is around 20. When compared to NCC-R, which has similar accuracy and is efficiently implemented, our algorithm is still around 5 to 6 times faster. In addition to the discussed accuracy and efficiency, the recognition method provides an additional asset. It estimates roughly the rotation that the patch describing a feature point has undergone. We take advantage of this estimation, propose a potential application to 2D top-view visual tracking and also present results when this method is integrated to our camera tracking system.

Experiments conducted with the camera tracker demonstrate that the synergy is achieved. Indeed, individual failure modes are solved and the overall tracking is benefitted from the characteristic strengths of each approach. The system successfully faces occlusions of the marker and of feature points, illumination and viewpoint changes, and drift. Another benefit of the fusion is that the three main capabilities that a camera tracking framework should have, are covered. These capabilities are: automatic initialisation, re-initialisation after loss of track, and an extended tracking area beyond the visibility of references known to the system prior to run-time.

The system has proven its applicability in the areas of human-machine interfaces and augmented reality environments. The most relevant contributions are in the context of the collaboration with the

University of art and design Lausanne (ECAL). One of the outputs of this collaboration is the joint development of the AiRToolkit. This toolkit extends our tracking system into a tangible interface. We have also been involved in a transfer of technology in the form of a demonstrator permanently exhibited in the Audiorama museum and in particular industrial fairs.

Let us now take a broader look at the achievements of this thesis and analyse the consequences of the results obtained. The main novelty of this thesis is proving that the fusion of a tracking-by-detection technique with feature point-based Bayesian tracking is beneficial for camera tracking. The results obtained evidence this fact. Indeed, a larger set of capabilities when compared to individual trackers is shown. A comparison with the state of the art reveals that no other published tracker covers this set of capabilities at the same time. A consequence of this novelty is opening the path to a different point of view in camera tracking. Most state of the art research on camera tracking has concentrated in incremental improvements in pose estimation accuracy. However, instead of focusing on the degree of accuracy achieved, a large number of applications and users would benefit from research that pays attention to the capabilities covered by a tracking system. This thesis presents a step towards this different paradigm.

8.2 Perspectives

The work presented in this dissertation can be extended in several directions. In Chapter 3, the RDTM method is presented. Research in this area could continue in the following aspects:

- The method concentrates on patches that have undergone rotations with respect to their normal. In many recognition applications, the distortion of a patch is more complex than a rotation. Therefore, our method should be extended to viewpoint invariance. Scale changes could be performed easily by changing the size of the scanning window. Indeed, the implementation using the integral image and integral histogram permits to scan images at a different scale with the same computation time. It would be interesting to study the distortion of gradient information in more generic three dimensional rotations.

- The orientation gradient histogram matching is prone to produce better results if texture information is available. A method is presented to detect such regions in an image. The method concentrates on regions with an heterogeneous-shaped histogram. It would be interesting to define an algorithm to detect regions with histograms having non periodic shapes. The Circular Normalised Euclidean Distance (CNED) used to match orientation histograms would benefit from such non-periodicity.

In Chapter 6, the camera tracking framework has been specified, developed and evaluated. Several variations of the framework proposed could be envisioned:

- The framework proposed is a loose coupling of cues. This type of coupling allows flexibility to explore other cues. One possibility would be to integrate other tracking-by-detection techniques. In this line, a technique that uses an arbitrary pattern and does not need the pattern to have a specific shape would be an interesting option. Moreover, such technique would make applicability to outdoor environments straightforward. A possible technique to integrate as cue is presented by Lepetit *et al.* [LLF05].

- A different exploratory path is to define a framework where the combination has tighter coupling. In our case, both cues work independently. Another possibility would be to assist a bottom-up estimation with top-down information and vice versa. Such approach could improve the accuracy and robustness although it would provide less flexibility.
- The FPC is used to constrain the camera pose. Feature points are searched in regions of the image determined from a top-down approach. If the estimate of the filter fails, the regions are incorrect and probably no reference is found. A way to address this problem would be to detect when the proportion of feature points correctly matched decreases below a certain threshold. Such situation should trigger an exhaustive search beyond the estimated regions. If the reason of the failure was really an incorrect pose estimation, the exhaustive search should find larger support. With more points detected, pose could be corrected. This is a sort of bottom-up approach to solve a top-down problem. Indeed, this particular investigation is related to the previous exploratory path mentioned.
- The RDTM has its own future investigation in the frame of camera tracking due to its use in the FPC. Since this cue relies on a TDA, it is possible to exploit the knowledge of the camera pose also in the matching process. On one hand, the CNED computes all the possible shifts of a histogram. This could be limited to a smaller number of possible shifts according to the estimation of the camera's rotation around the Z axis. On the other hand, scale changes could be approximated using the distance between the feature point and the camera. A possible starting point is the work of Chekhlov *et al.* [CPMC07].

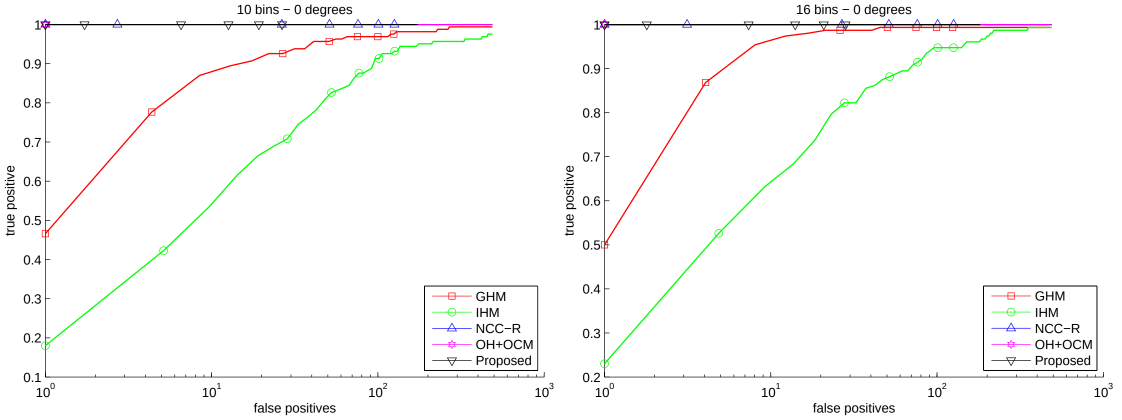
Part IV

Appendix

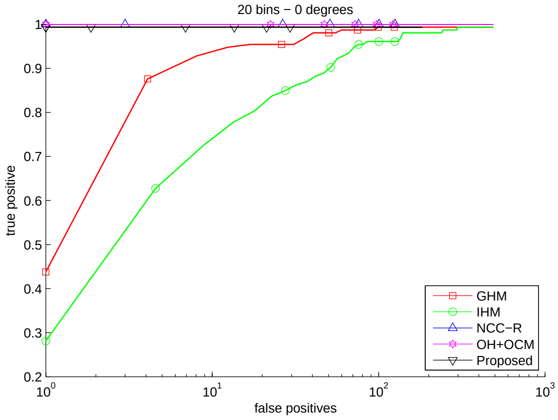
A

Complementary results on rotation-discriminative template matching

Complementary results of the RDTM are given in this appendix. Figure A.1 shows the results obtained for the non-rotated image and the different histogram lengths. Figures A.2-A.4 depicts the results obtained for different histogram lengths and with patches extracted at random positions in the original images. The discussion of this results is given earlier in Section 3.5.4.

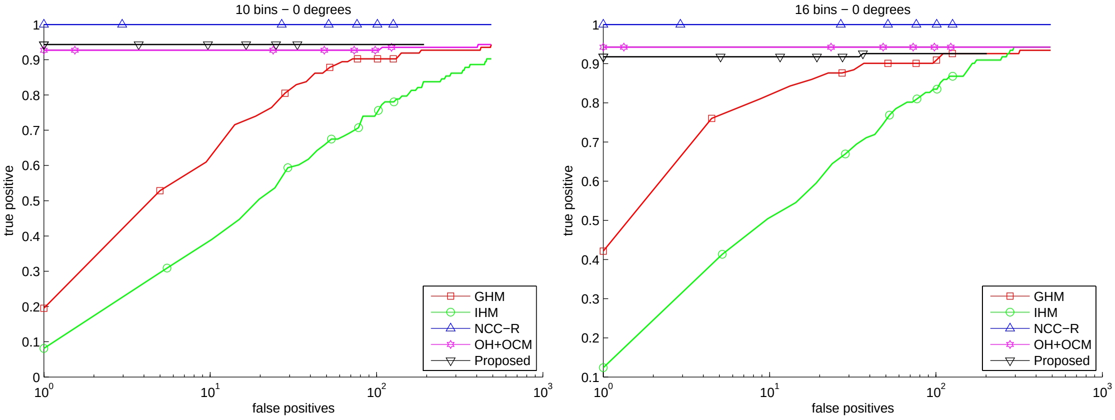


(a) 10 bins (b) 16 bins

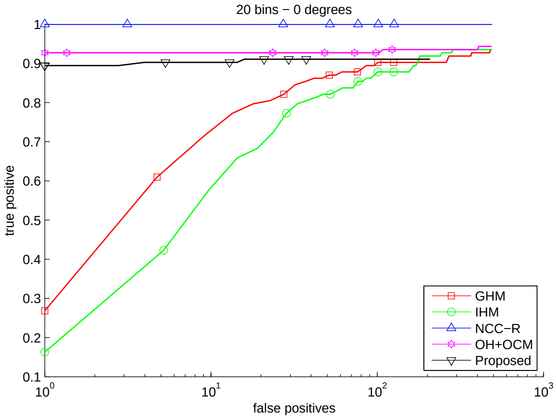


(c) 20 bins

Figure A.1: Average true positive (tp) and false positives (fp) among all the patches extracted at feature points. Rotation angle: 0 degrees.

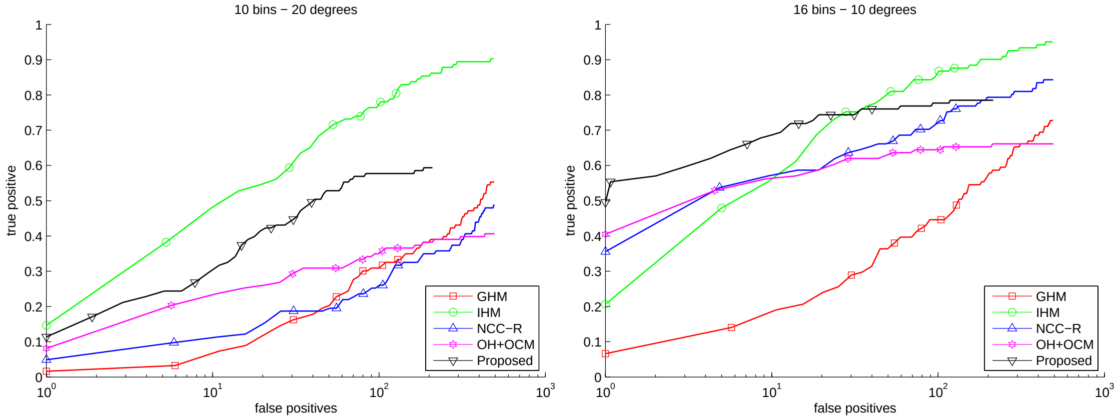


(a) 10 bins (b) 16 bins



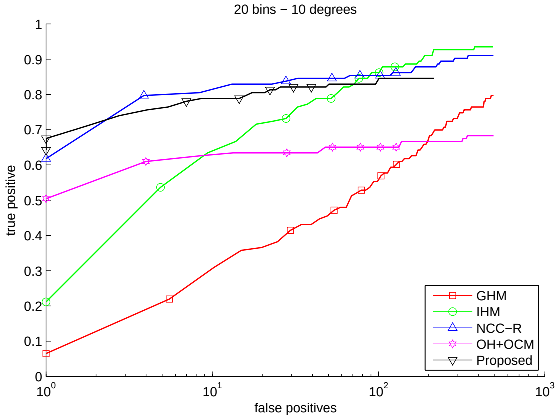
(c) 20 bins

Figure A.2: Average true positive (tp) and false positives (fp) among all the patches extracted randomly. Rotation angle: 0 degrees.



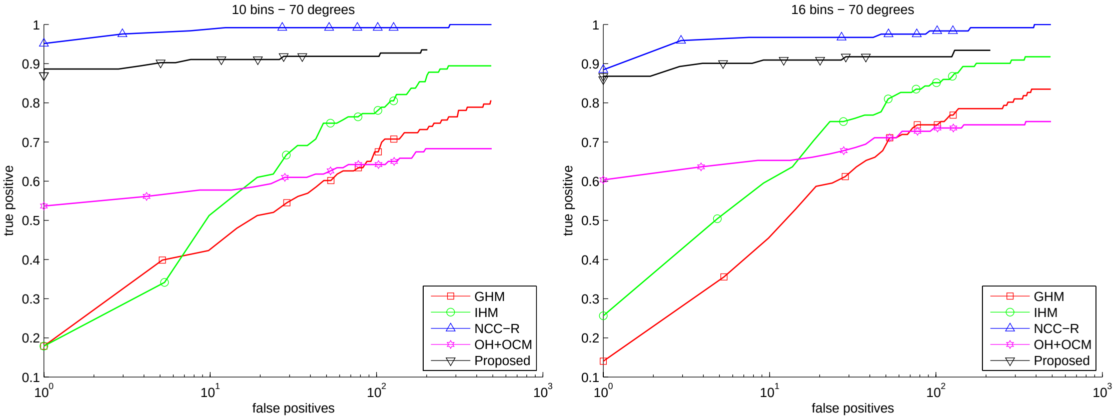
(a) 10 bins, 20 degrees

(b) 16 bins, 10 degrees



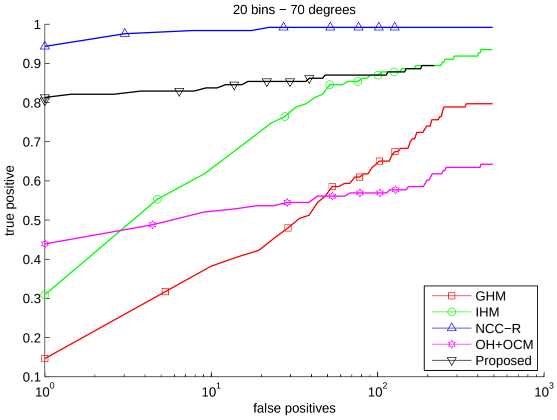
(c) 20 bins, 10 degrees

Figure A.3: Average true positive (tp) and false positives (fp) among all the patches extracted randomly. Rotation angle: $\sim k\Delta + \Delta/2$.



(a) 10 bins

(b) 16 bins



(c) 20 bins

Figure A.4: Average true positive (tp) and false positives (fp) among all the patches extracted randomly. Rotation angle: 70 degrees $\sim k\Delta$.

Bibliography

[ABB$^+$01] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Application*, 21(6):34–47, Nov 2001.

[ABW01] B.D. Allen, G. Bishop, and G. Welch. Tracking: Beyond 15 minutes of thought. In *Course Notes, Ann. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001.

[AC05] A. Agrawal and R. Chellappa. Fusing depth and video using Rao-blackwellized particle filter. In *Proc. Intl. Conf. Pattern Recognition and Machine Intelligence (PReMI)*, volume 3776/2005, pages 521–526, Kolkata, India, December 2005.

[AME05] Y. Abdeljaoued, D. Marimon, and T. Ebrahimi. Tracking and User Interface for Mixed Reality. In O. Schreer, P. Kauff, and T. Sikora, editors, *3D Videocommunication : Algorithms, concepts and real-time systems in human centred communication*, pages 315–332. John Wiley and Sons Ltd, 2005.

[AMGC02] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, Feb. 2002.

[AMR04] F. Ababsa, M. Mallem, and D. Roussel. Comparison between particle filter approach and kalman filter-based technique for head tracking in augmented reality systems. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, volume 1, pages 1021–1026, April-May 2004.

[ARS06] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 798–805, June 2006.

[ART07] ARToolkit. `www.hitl.washington.edu/artoolkit/`, 2007.

[Azu97] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):335–385, Aug 1997.

[Bak00] T. Bak. Lecture notes - estimation and sensor information fusion. Technical report, Department of Control Engineering, Aalborg University., 11 2000.

[BB93] G. Baratoff and S. Blanksteen. Tracking devices. In B. Shneiderman, editor, *Encyclopedia of Virtual Environments*. Human Interface Technology Lab (HITLab), 1993.

[BC03] G. Burdea and P. Coiffet. *Virtual Reality Technology*. Wiley, 2 edition, 2003.

[BdB94] J. Bigün and J. M. H. du Buf. N-folded symmetries by complex moments in Gabor space and their application to unsupervised texture segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(1):80–87, 1994.

[BFHS96] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proc. National Conf. on Artificial Intelligence (AAAI)*, volume 2, pages 896–901, 1996.

[BFO92] M. Bajura, H. Fuchs, and R. Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. In E. Catmull, editor, *Ann. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, volume 26, pages 203–210, July 1992.

[Bir98] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 232–237, June 1998.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BK02] M. Billinghurst and H. Kato. Collaborative augmented reality. *Communications of the ACM (CACM)*, 45(7):64–70, 2002.

[BMK$^+$07] C. Babski, D. Marimon, P. Keller, B. Dauw, and T. Rihs. AiRToolkit - Variable environment / mobility, interaction city and crossovers Research Project. `http://sketchblog.ecal.ch/variable_environment/archives/2007/07/projects_the_so.html`, July 2007.

[Bou07] Boujou from 2d3 Ltd. . `www.2d3.com`, 2007.

[BR05] S. Birchfield and S. Rangarajan. Spatiograms versus histograms for region-based tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1158–1163, San Diego, California, June 2005.

[BSB00] Y. Bar-Shalom and W. D. Blair. *Multitarget-Multisensor Tracking Applications and Advances - Volume III*, chapter 3. Artech House, 2000.

[BVBC04] V. Buchmann, S. Violich, M. Billinghurst, and A. Cockburn. Fingartips: gesture based direct manipulation in augmented reality. In *Proc. of the 2nd Int. Conf. on Computer graphics and interactive techniques in Austalasia and SouthEast Asia (Graphite)*, pages 212–221, Jun 2004.

[CCP02] K. W. Chia, A.D. Cheok, and S.J.D. Prince. Online 6 dof augmented reality registration from natural features. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 305–313, Sep–Oct 2002.

[CF04] D. Claus and A. Fitzgibbon. Reliable fiducial detection in natural scenes. In *Proc. European Conference on Computer Vision (ECCV)*, volume 3024, pages 469–480, Prague, Czech Republic, May 2004. Springer-Verlag.

[CH96] I.J. Cox and S.L. Hingorani. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(2):138–150, Feb 1996.

[Che03] Z. Chen. Bayesian filtering: From Kalman filters to particle filters, and beyond. Technical report, McMaster University, 2003.

[CJ02] G. Carneiro and A. D. Jepson. Phase-based local features. In *Proc. European Conference on Computer Vision (ECCV)*, pages 282–296, London, UK, 2002. Springer-Verlag.

[CM92] T. Caudell and D. Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proc. Hawaii International Conf. on Systems Science (HICSS)*, volume 2, pages 659–669, 1992.

[CNHV99] L. Chai, K. Nguyen, B. Hoff, and T. Vincent. An adaptive estimator for registration in augmented reality. In *Proc. IEEE and ACM Intl. Workshop on Augmented Reality (IWAR)*, pages 23–32, 1999.

[CPMC07] D. Chekhlov, M. Pupilli, W. Mayol, and A. Calway. Robust real-time visual slam using scale prediction and exemplar based feature description. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[CPMCC06] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway. Real-time and robust monocular slam using predictive multi-resolution descriptors. In *2nd International Symposium on Visual Computing*, November 2006.

[CRM03] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

[Dav03] A. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2003.

[DBFT99] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 1999.

[DC02] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):932–946, July 2002.

[DdG01] A. Doucet, N. deFreitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001. ISBN 0-387-95146-6.

[DGBD05] B. Deutsch, Ch. Graessl, F. Bajramovic, and J. Denzler. A comparative evaluation of template and histogram based 2d tracking algorithms. *Lecture Notes in Computer Science*, 3663:269–276, 2005.

[DM02] A. Davison and D. Murray. Simultaneous localization and map-building using active vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):865–880, 2002.

[ED06] E. Eade and T. Drummond. Scalable monocular slam. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 469–476, Washington, DC, USA, 2006. IEEE Computer Society.

[EE07] ECAL and EPFL. Variable environment / mobility, interaction city and crossovers Research Project. `http://sketchblog.ecal.ch/variable_environment/`, July 2007.

[FA91] W. Freeman and E. Adelson. The design and use of steerable filters. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[Fak04] FakeSpaceLabs. `www.fakespacelabs.com/`, 2004.

[FAS07] FASTRAK from Polhemus. `www.polhemus.com/?page=Motion_Fastrak`, 2007.

[Faw06] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[FB81] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM (CACM)*, 24(6):381–395, 1981.

[FDS90] P. Fischer, R. W. Daniel, and K. V. Siva. Specification and design of input devices for teleoperation. In *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume vol.1, 1990.

[FHP98] E. Foxlin, M. Harrington, and G. Pfeifer. Constellation™: A wide-range wireless motion-tracking system for augmented reality and virtual set applications. In *Proc. Computer Graphics (SIGGRAPH)*, pages 371–378. ACM Press, Addison-Wesley, Orlando(FL), USA, 1998.

[Fia05] M. Fiala. ARTag, a fiducial marker system using digital techniques. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 590–596, Washington, DC, USA, 2005. IEEE Computer Society.

[FKCK05] A. J. Fitch, A. Kadyrov, W. J. Christmas, and J. Kittler. Fast robust correlation. *IEEE Trans. on Image Processing*, 14(8):1063–1073, 2005.

[FN03] E. Foxlin and L. Naimark. Vis-tracker: a wearable vision-inertial self-tracker. In *Proc. IEEE Virtual Reality (VR)*, pages 199–206, Mar 2003.

[Fox96] E. Foxlin. Inertial head-tracker sensor fusion by a complementary separate-bias Kalman filter. In *Proc. IEEE Virtual Reality Annual Intl. Symp. (VRAIS)*, pages 185–194, Santa Clara, CA, USA, 30 March – 3 April 1996.

[FT97] P. Fieguth and D. Terzopoulos. Color-based tracking of heads and other mobile objects at video frame rates. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 21–27, Washington, DC, USA, 1997.

[FU01] K. Fredriksson and E. Ukkonen. Faster template matching without FFT. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, volume 1, pages 678–681, 2001.

[FZ00] A. Fitzgibbon and A. Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *Proc. European Conference on Computer Vision (ECCV)*, pages 891–906, London, UK, 2000. Springer-Verlag.

[GF02] J.-S. Gutmann and D. Fox. An experimental comparison of localization methods continued. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, October 2002.

[GM04] B. Georgescu and P. Meer. Point matching under large image deformations and illumination changes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(6):674–688, 2004.

[Ham63] W. Hamilton. *Elements of Quaternions*. Chelsea Publishing Co., New York, third edition, 1963.

[Har92] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*, pages 59–73. MIT Press, 1992.

[HB96] G.D. Hager and P.N. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 403–410, 1996.

[HGN01] E. Hadjidemetriou, M.D. Grossberg, and S.K. Nayar. Spatial information in multiresolution histograms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 702–709, 2001.

[HiB07] HiBall from 3rdTech. `www.3rdtech.com/HiBall.htm`, 2007.

[HKM+97] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 762–768, Washington, DC, USA, 1997.

[HL01] D. L. Hall and J. Llinas, editors. *Handbook of multisensor data fusion*. Number ISBN: 0849323797. CRC Press, 2001.

[HLON94] R. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, December 1994.

[HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conf.*, pages 147–151, 1988.

[HSD73] R. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Trans. on Systems, Man., and Cybernetics*, 3(6):610–621, 1973.

[HZ00] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[IB98] M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *Intl. Journal of Computer Vision*, 29(1):5–28, 1998.

[Ich02] N. Ichimura. Stochastic filtering for motion trajectory in image sequences using a monte carlo filter with estimation of hyper-parameters. In *Proc. Intl. Conf. on Pattern Recognition (ICPR)*, volume 4, pages 68–73, 2002.

[Imm04] Immersion. `www.immersion.com/3d/`, 2004.

[Int05] InterSense. `www.intersense.com/`, 2005.

[JD02a] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):996–1000, 2002.

[JD02b] F. Jurie and M. Dhome. Real time robust template matching. In *Proc. British Machine Vision Conference (BMVC)*, pages 123–132, September 2002.

[JU97] S. Julier and J. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Proc. Intl. Symp. on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II (AeroSense)*. SPIE, 1997.

[Kai67] T. Kailath. The Divergence and Bhattacharyya distance measures in signal selection. *IEEE Trans. on Communications*, 15(1):52–60, 1967.

[KB99] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proc. Intl. Workshop on Augmented Reality (IWAR)*, pages 85–94, Oct 1999.

[KB01] T. Kadir and M. Brady. Saliency, scale and image description. *Intl. Journal of Computer Vision*, 45(2):83–105, 2001.

[KD03] G. Klein and T. Drummond. Robust visual tracking for non-instrumented augmented reality. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 113–122, Oct 2003.

[KD04] G. Klein and T. Drummond. Sensor fusion and occlusion refinement for tablet-based ar. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 38–47, Nov 2004.

[KFTY00] M. Kanbara, H. Fujii, H. Takemura, and N. Yokoya. A stereo vision-based augmented reality system with an inertial sensor. In *Proc. IEEE and ACM Intl. Symp. on Augmented Reality (ISAR)*, pages 97–100, Oct 2000.

[KKR$^+$97] D. Koller, G. Klinker, E. Rose, D. Breen, R. Wihtaker, and M. Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *ACM Symposium on Virtual Reality Software and Technology (VRST)*, 1997.

[KKSES05] R. Koch, K. Koeser, B. Streckel, and J.-F. Evers-Senne. Markerless image-based 3d tracking for real-time augmented reality applications. In *Intl. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, April 2005.

[KOTY00] M. Kanbara, T. Okuma, H. Takemura, and N. Yokoya. A stereoscopic video see-through augmented reality system based on real-time vision-based registration. In *Proc. IEEE Virtual Reality (VR)*, pages 255–262, Mar 2000.

[KRC97] D. Kim, S. W. Richards, and T. P. Caudell. An optical tracker for augmented reality and wearable computers. In *Proc. IEEE Virtual Reality Annual Intl. Symp. (VRAIS)*, pages 146–150, 1997.

[Kuh55] H. W. Kuhn. The Hungarian method for the assignment and transportation problems. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[LC04] T.-L. Liu and H.-T. Chen. Real-time tracking using trust-region methods. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(3):397–402, 2004.

[Lew95] J.R. Lewis. Fast template matching. *Vision Interface*, pages 120–123, 1995.

[LF05] V. Lepetit and P. Fua. Monocular model-based 3D tracking of rigid objects. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, September 2005. ISBN: 1-933019-03-4.

[LLF05] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 775–781, Washington, DC, USA, 2005. IEEE Computer Society.

[LLS05] T. Lemaire, S. Lacroix, and J. Sola. A practical 3d bearing-only slam algorithm. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2449–2454, Aug 2005.

[Low92] D. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Intl. Journal of Computer Vision*, 8(2):113–122, 1992.

[Low04] D. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. Journal of Computer Vision*, 60(2):91–110, 2004.

[LPF04] V. Lepetit, J. Pilet, and P. Fua. Point matching as a classification problem for fast and robust object pose estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 244–250, June 2004.

[LSM$^+$03] Y. Liu, M. Storring, T.B. Moeslund, C.B. Madsen, and E. Granum. Computer vision based head tracking from re-configurable 2d markers for ar. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 264–267, Oct 2003.

[LSP05] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005.

[LVTF03] V. Lepetit, L. Vacchetti, D. Thalmann, and P. Fua. Fully automated and stable registration for augmented reality applications. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 93–102, Oct 2003.

[May79] P. S. Maybeck. *Stochastic models, estimation, and control*, volume 141-1, chapter Introduction, pages 1–16. Academic Press, 1979.

[May82] P. S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 141-2, chapter Parameter uncertainties and adaptive estimation, pages 68–158. Academic Press, 1982.

[MDR04] N. Molton, A. Davison, and I. Reid. Locally planar patch features for real-time structure from motion. In *Proc. British Machine Vision Conference (BMVC)*, 2004.

[ME07a] D. Marimon and T. Ebrahimi. Combination of video-based camera trackers using a dynamically adapted particle filter. In *2nd Intl. Conf. on Computer Vision Theory and Applications (VISAPP07)*, 2007.

[ME07b] D. Marimon and T. Ebrahimi. Orientation histogram-based matching for region tracking. In *Intl. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2007.

[MMAE07] D. Marimon, Y. Maret, Y. Abdeljaoued, and T. Ebrahimi. Particle filter-based camera tracker fusing marker- and feature point-based cues. In *Proc. of the IS&T/SPIE Conf. on Visual Communications and Image Processing (VCIP)*, 2007.

[MRG95] P. Milgram, A. Rastogi, and J.J. Grodski. Telerobotic control using augmented reality. In *Proc. IEEE Intl. Workshop on Robot and Human Communication (RO-MAN)*, pages 21–29, Jul 1995.

[MS05] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[MTKW02] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[MTS$^+$05] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *Intl. Journal of Computer Vision*, 65(1/2):43–72, 2005.

[Mue92] P. Mueller. Posterior integration in dynamic models. *Computer Science and Statistics*, 24:318–324, 1992.

[Mur99] K. Murphy. Bayesian map learning in dynamic environments. In *Neural Information Processing Systems (NIPS)*, 1999.

[Nav04] Nassir Navab. Developing killer apps for industrial augmented reality. *IEEE Computer Graphics and Applications*, 24(3):16–20, 2004.

[NF02] L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 27–36, Sep–Oct 2002.

[NNB04] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 01, pages 652–659, Los Alamitos, CA, USA, June 2004. IEEE Computer Society.

[NNK04] H. Najafi, N. Navab, and G. Klinker. Automated initialization for marker-less tracking: a sensor fusion approach. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 79–88, Nov 2004.

[NP98] U. Neumann and J. Park. Extendible object-centric tracking for augmented reality. In *Proc. IEEE Virtual Reality Annual Intl. Symp. (VRAIS)*, page 148, Washington, DC, USA, 1998. IEEE Computer Society.

[NY99] U. Neumann and S. You. Natural feature tracking for augmented reality. *IEEE Trans. on Multimedia*, 1(1):53–64, Mar 1999.

[OKS03] T. Okuma, T. Kurata, and K. Sakaue. Fiducial-less 3-d object tracking in AR systems based on the integration of top-down and bottom-up approaches and automatic database addition. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, page 260, 2003.

[OKS04] T. Okuma, T. Kurata, and K. Sakaue. A natural feature-based 3D object tracking method for wearable augmented reality. In *Proc. IEEE Intl. Workshop on Advanced Motion Control (AMC)*, pages 451–456, 2004.

[PC05] M. Pupilli and A. Calway. Real-time camera tracking using a particle filter. In *Proc. British Machine Vision Conference (BMVC)*, pages 519–528, September 2005.

[PC06a] M. Pupilli and A. Calway. Real-time camera tracking using known 3D models and a particle filter. In *Proc. Intl. Conf. on Pattern Recognition (ICPR)*, August 2006.

[PC06b] M. Pupilli and A. Calway. Real-time visual slam with resilience to erratic motion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2006.

[PHVG02] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. European Conference on Computer Vision (ECCV)*, pages 661–675, London, UK, 2002. Springer-Verlag.

[PK97] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(3):206–218, 1997.

[PKV99] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown intrinsic camera parameters. *Intl. Journal of Computer Vision*, 32(1):7–25, 1999.

[Por05] F. Porikli. Integral histogram: a fast way to extract histograms in cartesian space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 829–836, 2005.

[PXC02] S.J.D. Prince, Ke Xu, and A.D. Cheok. Augmented reality camera tracking with homographies. *IEEE Computer Graphics and Applications*, 22(6):39–45, Nov–Dec 2002.

[QC04] G. Qian and R. Chellappa. Structure from motion using sequential monte carlo methods. *Intl. Journal of Computer Vision*, 59(1):5–31, 2004.

[RBSJ79] F. H. Raab, E. B. Blood, T. O. Steiner, and H. R. Jones. Magnetic position and orientation tracking system. In *IEEE Trans. Aerospace and Electronic Systems*, volume AES-15, pages 709–718, 1979.

[RDLW95] S. Ravela, B. Draper, J. Lim, and R. Weiss. Adaptive tracking and model registration across distinct aspects. In *Proc. Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots'*, 1995.

[Rei79] D. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automatic Control*, 24(6):423–432, Dec 1979.

[Rob07] Robotics Research Group, Department of Engineering Science, University of Oxford. Visual geometry group's image database. `http://www.robots.ox.ac.uk/~vgg/data/`, 2007.

[RPTB01] Y. Rubner, J. Puzicha, C. Tomasi, and J.M. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, 84(1):25–43, 2001.

[RTG00] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *Intl. Journal of Computer Vision*, 40(2):99–121, 2000.

[SB02] G. Simon and M.-O. Berger. Pose estimation for planar structures. *IEEE Computer Graphics and Applications*, 22(6):46–53, 2002.

[SEGL05] R. Sim, P. Elinas, M. Griffin, and J. J. Little. Vision-based SLAM using the Rao-Blackwellised particle filter. In *Proc. IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR)*, pages 9–16, Edinburgh, Scotland, 2005.

[SFZ00] G. Simon, A.W. Fitzgibbon, and A. Zisserman. Markerless tracking using planar structures in the scene. In *Proc. IEEE and ACM Intl. Symp. on Augmented Reality (ISAR)*, pages 120–128, Oct 2000.

[SG99] J. Sherrah and S. Gong. Fusion of 2D face alignment and 3D head pose estimation for robust and real-time performance. In *Proc. Intl. Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS)*, page 24, Washington, DC, USA, 1999. IEEE Computer Society.

[SK01] D. Stricker and T. Kettenbach. Real-time and markerless vision-based tracking for outdoor augmented reality applications. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, 2001.

[SL04] I. Skrypnyk and D.G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 110–119, 2004.

[SLB99] G. Simon, V. Lepetit, and M.-O. Berger. Computer vision methods for registration: mixing 3D knowledge and 2D correspondences for accurate image composition. In *Proc. Intl. Workshop on Augmented Reality (IWAR)*, pages 111–127, Natick, MA, USA, 1999. A. K. Peters, Ltd.

[SLG$^+$96] A. State, Mark A. Livingston, William F. Garrett, Gentaro Hirota, Mary C. Whitton, Etta D. Pisano, and Henry Fuchs. Technologies for augmented-reality systems: Realizing ultrasound-guided needle biopsies. In *Proc. Computer Graphics (SIGGRAPH)*, pages 439–446, 1996.

[SM97] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(5):530–535, May 1997.

[SSC87] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *Intl. Symp. of Robotics Research*, pages 467–474, Cambridge, MA, USA, 1987. MIT Press.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.

[SUYT03] K. Satoh, S. Uchiyama, H. Yamamoto, and H. Tamura. Robot vision-based registration utilizing bird's-eye view with user's view. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 46–55, Oct 2003.

[SWB92] L.S. Shapiro, H. Wang, and J.M. Brady. A matching and tracking strategy for independently moving objects. In *Proc. British Machine Vision Conference (BMVC)*, pages 306–315, 1992.

[Thr03] S. Thrun. Robotic mapping: a survey. In *Exploring artificial intelligence in the new millennium*, pages 1–35. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[TK92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *Intl. Journal of Computer Vision*, 9(2):137–154, 1992.

[UK95] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. In *Proc. Intl. Conf. on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed)*, pages 13–22, London, UK, 1995. Springer-Verlag.

[UK04] F. Ullah and S. Kaneko. Using orientation codes for rotation-invariant template matching. *Pattern Recognition*, 37(2):201–209, February 2004.

[Val02] N. M. Vallidis. *WHISPER: A Spread Spectrum Approach to Occlusion in Acoustic Tracking*. PhD thesis, University of North Carolina at Chapel Hill, Department of Computer Science, 2002.

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 511–518, 2001.

[VLF04a] L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, Arlington, VA, Nov. 2004.

[VLF04b] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3D tracking using online and offline information. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, 2004.

[VMU96] L. J. Van Gool, T. Moons, and D. Ungureanu. Affine/ photometric invariants for planar intensity patterns. In *Proc. European Conference on Computer Vision (ECCV)*, pages 642–651, London, UK, 1996. Springer-Verlag.

[VRB01] C. J. Veenman, M. J. T. Reinders, and E. Backer. Resolving motion correspondence for densely moving points. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(1):54–72, January 2001.

[WAB$^+$90] J. F. Wang, R. Azuma, G. Bishop, V. Chi, J. Eyles, and H. Fuchs. Tracking a head-mounted display in a room-sized environment with head-mounted cameras. In *Proc. SPIE Helmet-Mounted Displays II*, volume 1290, 1990.

[WF02] G. Welch and E. Foxlin. Motion tracking: no silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications*, 22(6):24–38, Nov–Dec 2002.

[WFM$^+$96] A. Webster, S. Feiner, B. MacIntyre, W. Massie, and T. Krueger. Augmented reality in architectural construction, inspection and renovation. In *Proc. ASCE Third Congress on Computing in Civil Engineering*, pages 913–919, June 1996. Anaheim, CA.

[WS03] D. Wagner and D. Schmalstieg. First steps towards handheld augmented reality. In *Proc. IEEE Intl. Symp. on Wearable Computers*, pages 127–135, Oct 2003.

[WS04] D. Wolf and G. S. Sukhatme. Online simultaneous localization and mapping in dynamic environments. In *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, April 2004.

[WVdM01] E. Wan and R. Van der Merwe. The unscented kalman filter. In S. Haykin, editor, *Kalman Filtering and Neural Networks*. Wiley, 2001.

[XL06] X. Xu and B. Li. Rao-blackwellised particle filter with adaptive system noise and its evaluation for tracking in surveillance. In John G. Apostolopoulos and Amir Said, editors, *Proc. of the IS&T/SPIE Conf. on Visual Communications and Image Processing (VCIP)*, volume 6077, page 60770W. SPIE, 2006.

[YJS06] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4):13, 2006.

[YNA99] S. You, U. Neumann, and R. Azuma. Hybrid inertial and vision tracking for augmented reality registration. In *Proc. IEEE Virtual Reality (VR)*, pages 260–267, Mar 1999.

[YWC04] Y.K. Yu, K.H. Wong, and M.M.Y. Chang. A fast and robust simultaneous pose tracking and structure recovery algorithm for augmented reality applications. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, volume 2, pages 1029–1032, 2004.

[ZC93] Q. Zheng and R. Chellappa. Automatic feature point extraction and tracking in image sequences for unknown camera motion. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, pages 335–339, 1993.

[ZGN01] X. Zhang, Y. Genc, and N. Navab. Taking AR into large scale industrial environments: Navigation and information access with mobile computers. In *Proc. Intl. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 179–180, Oct 2001.

Curriculum Vitae

DAVID MARIMÓN SANJUÁN

Av. de la Dent d’Oche 3
CH - 1007 Lausanne
david.marimon@gmail.com

Spanish
Age: 28
Status: single

EXPERIENCE

2003 - present

Research Assistant, Swiss Federal Institute of Technology (EPFL, Switzerland)

- R&D on camera tracking, computer vision and image processing. Producer of a novel approach in video-based cue combination for camera tracking.
- R&D partner with University of Art and Design Lausanne. Established shared platform for user interaction through Augmented Reality.
- Delegate in EU project K-SPACE. Responsible for financial and technical reporting.
- Delegate in EU project PERSEO. Developer of an MPEG-7 authoring tool and a client/server prototype for multimedia search and retrieval.
- Teaching assistant of Image Processing and of Media Security lectures. Co-advisor of 3 diploma projects.

2000 - 2001

Responsible for Logistics Intern, IBM Learning Services, Barcelona (Spain)

- Managed schedule, room and material. The procedure was not organised when I started. I structured the whole process and wrote a handbook for future interns. I involved myself in tasks beyond my responsibilities providing timely solutions.
- Responsible for official certification process. I dealt directly with the customers.
- Given my results, I was invited to join IBM as a teacher.

1998 - 2000

Maths Teacher for external support of university students, ASES 94 S.L. (Spain)

- Algebra (1st year) and Telecommunications Maths (2nd year).
- Introduced a new course topic and a new product line that doubled and tripled the number of students compared to previous semesters.

EDUCATION

2003 - present	PhD student in Computer, communication and computer science, EPFL
1997 - 2003	Master in Telecommunications Engineering. Universitat Politècnica de Catalunya (UPC), Barcelona (Spain) Exchange Student, EPFL (Switzerland). Feb-Sep 2002. During this period, I did my diploma project. This work awarded me a research assistant position in the same lab.
Beyond academic studies:	
2000	*3G: GSM takes Internet everywhere*. Summer course organised by the Board of European Students of Technology. Sponsored by ERICSSON. Bucharest (Romania)

LANGUAGES

Spanish	Mother tongue (bilingual with Catalan)
English	Fluent spoken and written. Certificates: Cambridge Advanced Exam
French	Fluent spoken. Intermediate written. Certificates: DELF 1st degree A1, A2, A3.
German	Simple conversation

COMPUTER SKILLS

Programming	C/C++, Java, HTML, XML, PHP, SQL, LaTeX. Libs: OpenCV, Intel IPP, OpenGL.
Applications	MSVisual Studio .NET and 2005, Matlab, AdobePhotoshop, MS-Office.
Systems	MS-Windows, Linux
Servers	Apache HTTP, Apache Xindice (XML DB), Real Helix Server (Multimedia Streaming).

SPORTS - HOBBIES - OTHERS

Ski and capoeira. Fan of films based on true stories. Riding my SUZUKI Bandit 600.
Member of Agrupament Escolta Ramon Llull (Catalan Scout Association) from 1994 to 1998. Food steward in two summer camps (15 children each) and participant in group's council.
I like challenges and I commit to them.

PUBLICATIONS

Book chapter

- ▷ Y. Abdeljaoued, D. Marimon and T. Ebrahimi, Tracking and User Interface for Mixed Reality, in *3D Videocommunication : Algorithms, concepts and real-time systems in human centred communication*, John Wiley and Sons Ltd, 2005.

Journal paper

- ▷ D. Marimon and T. Ebrahimi, Rotation Correlation Maps, *EURASIP Journal on Image and Video Processing*, ref. 75862, 2007.

Conference papers

- ▷ D. Marimon and T. Ebrahimi, Efficient rotation-discriminative template matching, *12th Iberoamerican Congress on Pattern Recognition (CIARP)*, L. Rueda, D. Mery, and J. Kittler (Eds.), Lecture Notes in Computer Science (LNCS) 4756, pp. 221–230, 2007.
- ▷ D. Marimon and T. Ebrahimi, Orientation histogram-based matching for Region Tracking, *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2007)*, 2007.
- ▷ D. Marimon and T. Ebrahimi, Combination of video-based camera trackers using a dynamically adapted particle filter, *Proc. 2nd International Conference on Computer Vision Theory and Applications (VISAPP07)*, pp. 363–370, 2007.
- ▷ D. Marimon, Y. Maret, Y. Abdeljaoued and T. Ebrahimi, Particle filter-based camera tracker fusing marker- and feature point-based cues,*Proc. of the IS&T/SPIE Electronic Imaging Conf. on Visual Communications and Image Processing (VCIP)*, 2007.
- ▷ D. Marimon, Y. Abdeljaoued, B. Palacios and T. Ebrahimi, Feature point tracking combining the Interacting Multiple Model filter and an efficient assignment algorithm, *Proc. of the IS&T/SPIE Electronic Imaging Conf. on Visual Communications and Image Processing (VCIP)*, 2007.
- ▷ Y. Maret, D. Marimon, F. Dufaux and T. Ebrahimi, Hierarchical Indexing using R-trees for Replica Detection, SPIE, 2006.
- ▷ D. Marimon, Y. Abdeljaoued and T. Ebrahimi, Online Registration Tool and Markerless Tracking for Augmented Reality, *Sixth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2005)*, 2005.
- ▷ O. Steiger, D. Marimon and T. Ebrahimi, MPEG-Based Personalized Content Delivery, *Proc. of IEEE International Conference on Image Processing (ICIP'03)*, Vol. 3, pp. 45-48, 2003.

Technical Report

- ▷ O. Steiger, M. Schneider Fontan, D. Marimon, Y. Abdeljaoued, T. Ebrahimi, S. Dominguez, J. San Pedro Wandelmer, N. Denis, F. Granelli and F. De Natale, Personalized Content Preparation and Delivery for Universal Multimedia Access, TR2005.010, 2005.