3D Visual Tracking of Articulated Objects and Hands

Teófilo Emídio de Campos
Saint Anne's College

Robotics Research Group
Department of Engineering Science
University of Oxford

Trinity Term 2006

This thesis is submitted to the Department of Engineering Science, University of Oxford, for the degree of Doctor of Philosophy. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Teófilo Emídio de Campos
Saint Anne's College

Doctor of Philosophy
Trinity Term 2006

3D Visual Tracking of Articulated Objects and Hands

Abstract

The ability to track multiple and articulated objects is an important one, not least in the areas of autonomous and teleoperated robotics, visual surveillance and human motion analysis. This thesis is concerned with marker-free real-time detection and tracking of articulated objects, targeting human hands with the aim to study methods that can be applied to enhance the interaction between humans and 3D (real or virtual) objects.

A survey summarises methods used to approach this and related problems in the literature. It indicates that, despite the large body of research in this field over twenty or so years, the area still proves challenging. Two main approaches have been identified. The first, known as generative tracking, uses an explicit kinematical representation of linkages or constraints between object parts and tracks by minimising error of projected control points. The second, known as discriminative approach, little is specified beforehand, but training data is used in order to create a map between image observations and 3D poses. This thesis describes novel work in both areas.

In the generative area, a method for tracking of articulated objects is described. It is a new extension of a method for tracking rigid objects in which the motion constraints between parts of the object are imposed up-front within the tracking process. The inter-frame pose update is derived as the solution of a linear system. This method has been applied to track articulated objects, including hands and multiple objects with motion constraints.

An alternative method is that based on estimating the motion of each subpart independently, thereby introducing redundant degrees of freedom, and imposing constraints later in a lower dimensional subspace. This method is reviewed and a comparison between this and the aforementioned method is presented in terms of accuracy, efficiency and robustness.

In the discriminative area, an inference-based approach is adopted in which a non-parametric relation between global image measurements and 3D poses is learnt using a multivariate regressor based on Relevance Vector Machine. This relation is a continuous map that allows fast and efficient pose estimation from static images. This method can detect and estimate the 3D pose of hands from static images, so it can be applied to (re-)initialise the generative tracker.

In this thesis, the use of multiple view is adopted as a solution to reduce the ambiguities for both generative and discriminative methods. Experiments with single and multiple views are described and a novel extension of the discriminative method for multiple views is proposed and evaluated.

Contents

Acknowledgements

First and foremost I would like to thank my supervisor, Prof David W. Murray, for his support and faith in my research. His mature and visionary thoughts were key to the development of this work and to my growth as a researcher. Among many other things that I admire in this busy man is his ability to provide wise proof-reading feedback at 3am, having to be awake at 8.30 in the morning.

My DPhil was funded by Brazilian Government CAPES scholarship BEX-1550/00-4. I was also benefited by the resources obtained through the grant GR/N03266 from the UK Engineering and Physical Science Research Council.

Many of the ideas presented here are the outcome of discussions with members of the Active Vision Lab and beyond. In particular I owe much to Ben Tordoff for his helping hand and for his contributions to the VW software library, which were very useful for this project. I must thank Walterio Mayol not only for the technical discussions, but also for helping me to remain sane specially in my first year in Oxford. Thanks Andrew Davison, for making computer vision even more exciting. Among other colleagues who contributed with discussions are: Keith Connor, Phil Tresadern, Nick Appostoloff and Huang Fei. Thanks to Nick Cavalcanti Jr for printing out and submitting the final version of this thesis.

I am grateful to the members of the Visual Geometry Group for their fruitful discussions in reading group meetings and seminars, and for attracting so many VIP visitors, some of whose have contributed to my work with discussions: Bill Triggs, Bjorn Stenger, Ankur Agarwal, Arasanathan Thayananthan and Matthieu Bray.

I would also like to thank my colleagues at Sharp Laboratories of Europe for the patience and support during the writing up of this thesis, in special Graham Jones, Lyndon Hill, Allan Evans, Harry Walton and Grant Bourhill.

I would like to thank Dr Ian Reid and Dr Tom Drummond, who examined this thesis, provided valuable feedback and reassuring discussions.

My Masters at the Universidade de São Paulo was an enabling factor for my DPhil in Oxford. I must acknowledge Roberto Cesar Jr, my friend and former supervisor, who made me interested in Computer Vision. Thanks to my friend Rogerio Feris for the (yet to succeed) plans to conquer the world. Thanks to Carlos H. Morimoto for his teaching and opinions about life out of Brazil.

I would like to thank all my friends who made my life more enjoyable in this period, in special Carlos Machado, Renata Moraes and Kuls Kaur. The one responsible for my peace of mind in the developments of my final experiments and during long period of writing up is my beloved Lucia Specia, who managed to bring me happiness in times of pressure and difficult decisions.

Last but not least, I would like to express my gratitude to my parents Rita and Beneticto, for their infinite support and faith. I dedicate this to them, and also to my grandparents Emilia, Waldomiro, Isabel and Geraldo.

1 Introduction

1.1 The importance of the interface in personal robotics

The vast growth in portable computer power, the loosening of restrictions on wireless communications, and advances in micro-fabrication and electronic miniaturization, have all contributed to the rapid growth of the field of personal or person-oriented robotics — a field that embraces the areas of assistive, recreational and humanoid robotics, and wearable computing [May04].

In the speculative design of future products in this field, it is frequently assumed that the human-computer interface (HCI) will be "natural". However, interfaces between operator and machine have been slow to evolve. The teleprinter, in use since the 1920s, has evolved internally but not radically into today's flat-panel display and ergonomic keyboard. The revolution spawned at Xerox Palo Alto of point and click using mouse and GUI is going on thirty years old. Developing natural interfaces by emulating natural competences in vision, speech, and so on, has proved much more difficult than envisaged — but the motivation to develop them is undiminished largely because impoverished interfaces really *do* limit the acceptance of smart devices. Who over the age of fifteen can be bothered to discover the key code sequences needed to use advanced features of their video recorder or mobile phone?

It is possible to identify three core areas of academic interest in the design of interfaces. First is how to sense and actuate near to, or upon, but *external* to, the human body. This is the area of perceptual user interfaces [TT97, Pen00], involving speech and vision. The second is how to interact with and relate to computing when it intrudes into our personal space, an area called social and affective computing

[DEA99, RN98, BA02]. The last is the study of how to sense and actuate *within* and connect to the human body using, for example, direct brain-computer interfaces [Moo01, LCO$^+$05, WGH$^+$03] .

1.2 Hand pose recovery and tracking

This thesis describes research into the first area: in particular into visual sensing for the recovery of hand pose and its changes over time. The hand, and our skill of manipulation, differentiates humans from all animals but the primates. It also plays an important rôle in the interaction between people, in conveying meaning and intention both non-verbally and as an addition to speech. Particularly powerful are pointing or deictic gestures which provide disambiguation, and gestures which accompany facial expressions [PSH97].

The first developments in hand pose estimation and tracking were achieved by using mechanical devices and gloves. Sturman and Zeltzer [SZ94] commented that ‘the history of tracking devices for mechanically or electrically interpreting hand motions began with post-WWII development of master-slave manipulator arms’ but they noted too the development of the pantograph during the Renaissance. One of the earliest optical hand tracking systems, based on LEDs, was that of Ginsberg and Maxwell [GM83]. However, such was the unreliability of vision-based hand trackers, that glove-based and mechanical systems (like the Phantom) were the only devices in use in the 1980s and 1990s. Indeed, they remained the sole way of recovering detailed and continuous joint information until very recently, when visual marker-based systems became commercially available (e.g. by Vicon). Their disadvantage is that, even when using the lightest fabric, glove-based devices restrict the movement of the hand they are meant to measure, and the user must carry either signal and power cables, or batteries if wireless links are used.

There are a several factors that contribute to the difficulty of visual tracking of the hand and recovery of its pose.

First there are issues of modelling. The shape of the hand is uncertain, even when the joint angles are known, and the lengths of the inter-joint links (bones) are defined. This is because the amount of tissue on the bone differs from person to person, and because it is squeezed about when the hand moves.

Then there are the problems of feature detection and data-association common to any tracking process. These are hugely compounded by the high degree of actual occlusion, and of “apparent occlusion”

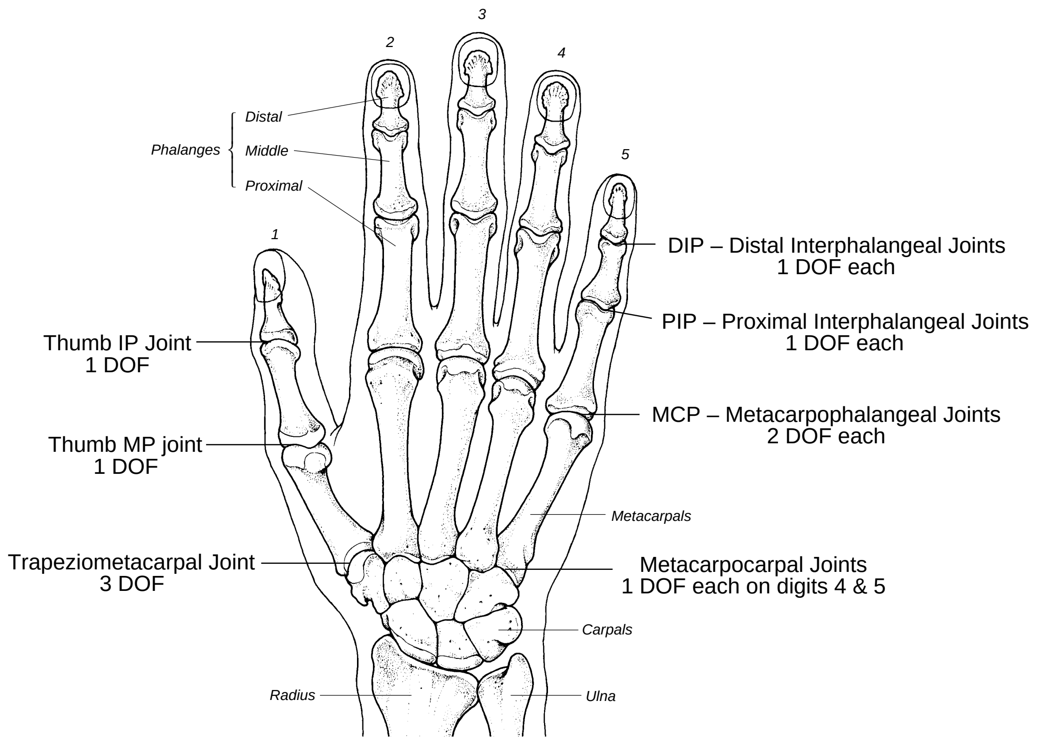


Figure 1.1: The human hand skeletal system and degrees of freedom of each joint as described by [Stu92]. (©[Stu92], reproduced with permission.)

where finger bounding contours are lost. It is, for example, all but impossible to determine the degree of flexion of fingers in some configurations using a single camera facing the back of a hand. These difficulties arise too in whole-body motion capture, but are sometimes eased there by the different colours of clothing. Work in gesture recognition has frequently exploited this, using gloves with differently coloured fingers (e.g. [Lyo02]).

Again on the "data side" of the processing the speed of hand movements can be very high, not only of the hand overall, but also of the joints, causing motion blur, aliasing, and loss of tracking. Again the hand presents more acute problems than that of whole-body motion, where the motion is slower, more constrained, and often cyclic (e.g. walking).

Even when the above are solved or finessed, the core difficulty remains. Recovering the pose is an optimization problem in a high dimensional space. Together, a hand and a forearm have 29 bones and 18 joints with observable movements, each joint with up to 3 degrees of freedom (DOF), as shown in Figure 1.1. Even the more compact model used conventionally in computer graphics models hands with 20 or 21 DOF. Such dimensionality can be reduced using the motion constraints imposed by tendons and

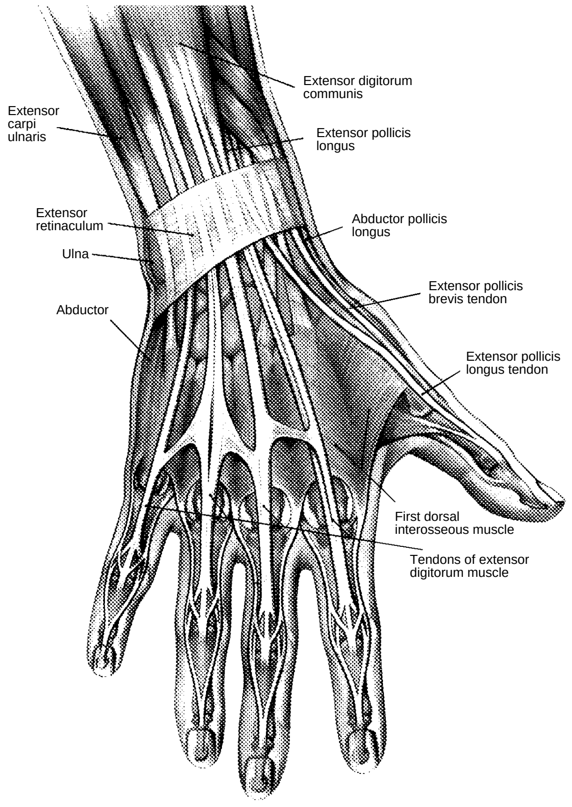

Extensor digitorum communis
Extensor carpi ulnaris
Extensor pollicis longus
Extensor retinaculum
Abductor pollicis longus
Ulna
Abductor
Extensor pollicis brevis tendon
Extensor pollicis longus tendon
First dorsal interosseous muscle
Tendons of extensor digitorum muscle

Figure 1.2: Muscles and tendons of the back of the hand introduce constraints on hand coupling joint positions. (©[Stu92], reproduced with permission.)

limitations of muscles in natural hand motion, as illustrated in Figure 1.2.

Finally, but very much related to the difficulties of taming high dimensionality, is the fundamental uncertainty as to what the overall problem representation should be. Some would argue that although 3D modelling has its imperfections, much is known about human body shape, kinematics and dynamics, and these should form the basis of any explanation of the image data. Others would argue that the visual data do not support recovery of a highly parametrized model, and that explanations based on non-parametric fitting of training data are more valid.

1.3 Key approaches

These two quite different approaches to estimate the pose of articulated objects in 3D are apparent in the literature: generative and discriminative.

1.3.1 Generative algorithms

The more traditional approach for tracking of objects in 3D encompasses the *generative* algorithms, also known as model-based or "estimation by synthesis" methods [SKS01]. In these methods, an initial estimate of the pose is used to update a model that predicts the appearance by projecting a 3D model into the image at a predicted pose. Then new measurements are obtained to estimate the pose update, as illustrated in Figure 1.3. The object model usually is a computer graphics replica of the target object designed by hand. In this approach, the accuracy of the object model and the camera calibration parameters are in many cases critical points for the tracking performance.

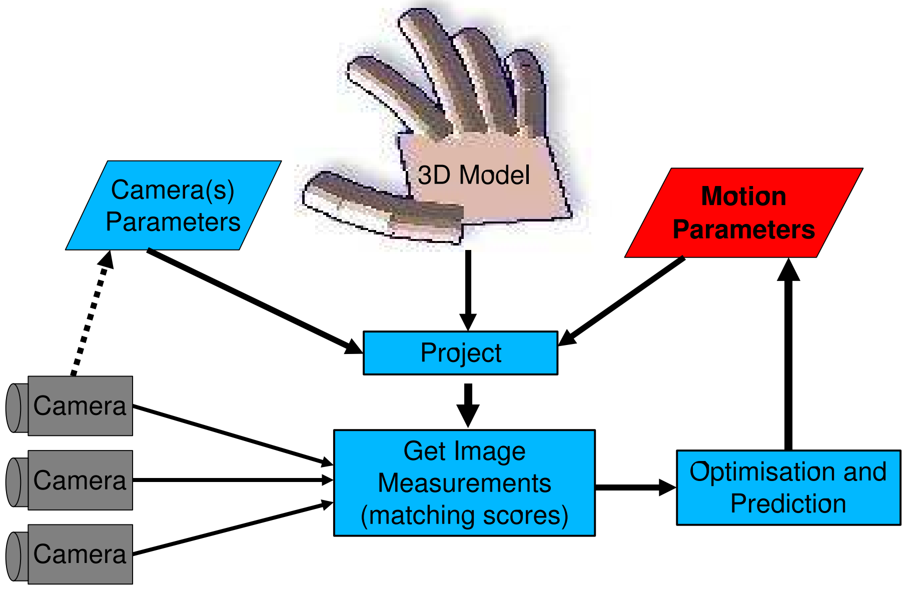


Figure 1.3: Generative approach for tracking: for each frame, an optimisation algorithm uses a motion model and the difference between the projected object model and the observed image to estimate the motion parameters, which then update the model pose.

Hogg's Walker [Hog83] is an example of *generative model-based* method, which relies on a more or less realistic jointed body model "enlivened" with a set of joint angles to predict appearance. The angles are adjusted to best fit the actual appearance in the image. Most hand and body models present in the literature use standard or generalised cylinder models [Hog83, MN78, BM98, SBF00], but both simpler planar and more sophisticated deformable models have been used e.g. [JBY96b, GD96].

Although it seems intuitive to use depth maps from stereo cameras (e.g. [HHD98]), cheaper image features can be explored. The most widely used feature in matching between image and model is the edge, e.g. [GD96, DBR00, DC02, Bra99, dTM06], but increasing use is made of internal features, such as corners [TMdM02a] and image motion [BM98, SBF00, YD98, WN99].

The key distinction in fitting pose to the image data is between works that adopt statistical techniques based on simple unimodal probability density functions (PDFs) and solve deterministically, e.g. [BM98, DC02, dTM06], and those that represent arbitrary multimodal PDFs using mixture models or particle filters [SBF00, DBR00, IB96]. Overcoming ambiguities, particularly troublesome from single views, is explored in [ST01a, RMK03]. By using a set of constraints and linear relations between joint angles, a closed-form solution for the inverse kinematics can be obtained if control points are reliably located in the finger tips and the hand palm [LH98]. However, this is only possible if visual markers are used, for marker-less tracking, more sophisticated estimation methods are required.

1.3.2 Discriminative algorithms

Discriminative approaches for 3D pose estimation provide a bridge between 3D model-based and 2D appearance-based methods. Although such methods output 3D pose configuration of articulated objects, these outputs are obtained after view-based measurements without the projection of an object model at a predicted pose to restrict the search. Instead, they are based on methods such as classification [AS02, AASK04, TSTC03, STTC03], temporal series [Bra99], or regression methods [RASS01, AT04c]. The relation $\mathbf{f}(\cdot)$ between image measurements $\mathbf{x}$ and 3D poses $\mathbf{y}$ is learned from the data, i.e., sets of correspondences $\mathbf{x} \rightarrow \mathbf{y}$. Furthermore, the set of measurements $\mathbf{x}$ may not have topological meaning and can be, for example, a set of global image descriptors. Figure 1.4 illustrates this approach.

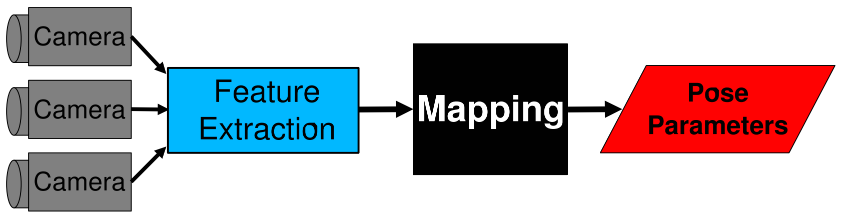


Figure 1.4: Discriminative approach for pose estimation: a mapping learnt from a large database of training pairs can output a pose directly from image measurements, without needing a model to generate image measurements.

Ideally training image measurements $\mathbf{x}$ and angles $\mathbf{y}$ would be recovered simultaneously. For the hand, the best – or at least the most obvious – method of recovering joint angles is using a data glove, but unfortunately wearing the glove ruins the imagery. For whole body, Hwang et al. [HKL06] recently presented a data base of 3D poses and silhouettes, but his has not been done for hands yet. Furthermore, this constrains the training set to the grabbed images, making it more difficult to try new camera configurations or to vary the global orientation of the hand to obtain a more comprehensive training set. Another method is to generate synthetic imagery from a hand model using synthesised joint angle data. In order to constrain the angles to natural motion, a commonly employed method [Bra99, AS03, AT04a] is to render imagery from 3D models using joint angles previously recovered from given data. This provides accurate ground truth data, but the resulting accuracy of the pose estimation for real images is reduced by the lack of realism in the training set.

These methods can be implemented using global image measurements, so prediction of the pose is not necessary and such methods can work without a coherent temporal sequence. On the other hand, their accuracy and comprehensiveness in terms of the parameter space is limited by the training set used, and the image measurements tend to be more expensive than those of generative methods. But the growth of computational power has recently enabled the implementation of methods that can cover a large range of 3D poses that can be recovered quickly.

1.4 This thesis

In this thesis, both generative and discriminative approaches have been explored: a generative model-based 3D tracker that assumes that the inter-frame motion of the hand can be reasonably well predicted; and a discriminative image-based 3D hand detector that can be applied for (re-)initialization of this tracker. Thus, the advantages of both systems can be combined: the robustness of a discriminative detector with the speed and accuracy of a generative tracker. The detector provides a set of initial pose parameters that roughly describes the pose of the objects given the observation, and the tracker quantitatively refines the estimates at each iteration (and for each new image frame).

While this work is centred on sensing and perception, it was developed in the context of other work in the Active Vision Laboratory in Oxford in wearable computing, and is envisaged as the precursor to

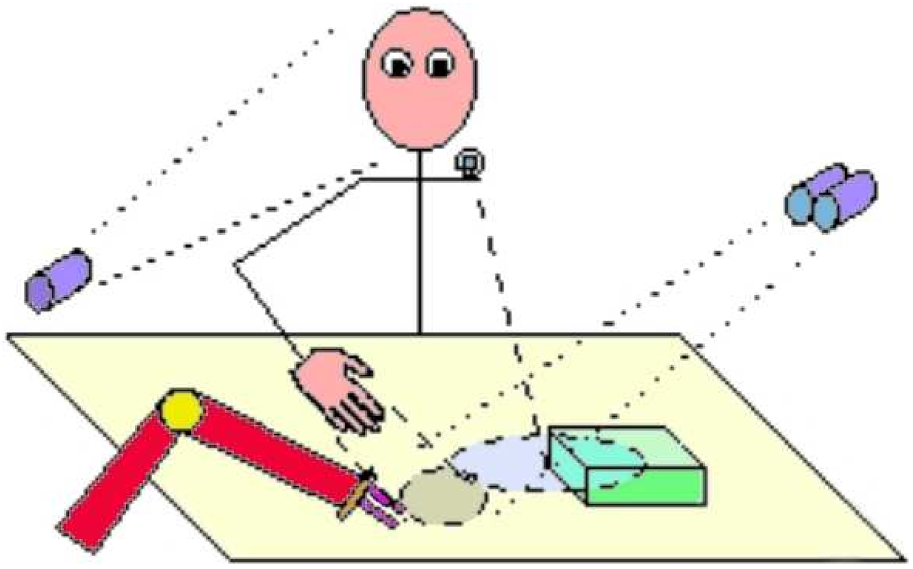


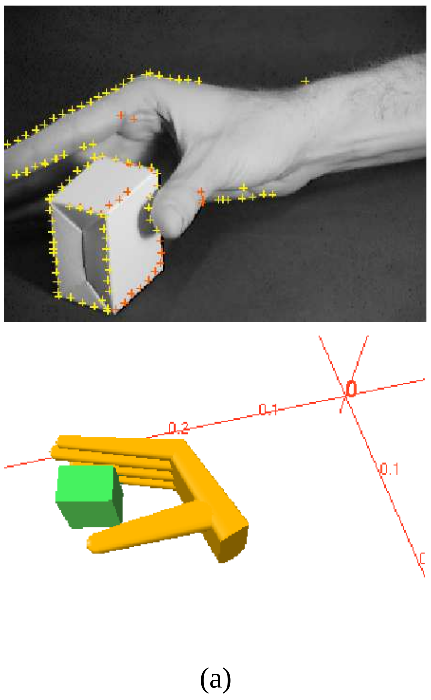
Figure 1.5: Overview of a motivating application. The user is watched by external cameras that track the head gaze to determine the focus of attention and the hands in interaction with objects. A wearable camera provides information from the user's view point and the combined outputs could be used to control an assistive robot arm.

exploring actuation in assistive robotics. Figure 1.5 sketches the sort of assistive workbench proposed. This thesis addresses the element of hand pose recovery using multiple cameras, work that is described chiefly in Chapters 5, 6 and 7 and has been published in [dTM06] and [dM06]. Figure 1.6 shows typical results. Of lesser relevance but still pertinent to this thesis is the work on head tracking using 3D appearance models of [TMdM02a], and the body of work on active wearable cameras described in Mayol's thesis [May04]. Some experiments using the wearable to track hands are reported here in Chapter 4 and in [dMM06].

1.4.1 Thesis outline

Description of the experimental work commences in Chapter 3, where certain background matters are outlined. The physical work space for tracking is described, followed by the client-server architecture for monitoring trackers that can run in parallel and the camera calibration algorithms. Also included is the skin colour detection method, which is a key developtment of this chapter, being used as a component of the works published in [TMdM02b], [dMM06], [MTdC^{+}03] and [dM06].

Chapter 4 describes an implementation of the RAPiD rigid object tracker [Har92a] and experiments



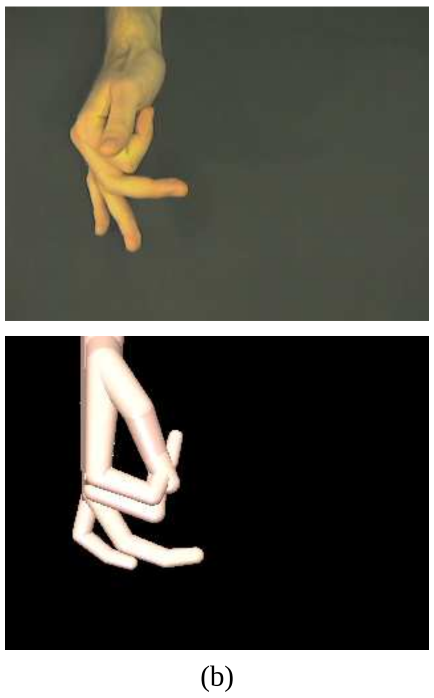


Figure 1.6: Example results from (a) 3D kinematical tracking and (b) 2D appearance based pose recover using a Relevance Vector Machine – work described in Chapters 5 and 7 respectively.

with multiple views using synthetic data. This is followed by the description of an application that combines a simple 2D image-based discriminative detector and RAPiD to estimate the pose of a pointing hand in 3D from the view of a wearable robot. This system is the main contribution of this chapter, published in [dMM06].

In Chapter 5, a novel extension of RAPiD is proposed for tracking of articulated objects. This tracker, dubbed ART (Articulated RAPiD Tracker), is tested on some sequences of images containing synthetic and real articulated objects. Next, Chapter 6 describes another state-of-the-art approach for real-time tracking of articulated objects [DC02] and shows comparisons with ART in terms of accuracy, efficiency and robustness. This novel comparison has been published in [dTM06].

Chapter 7 describes a discriminative approach for hand pose estimation from static images. This method uses global image measurements based on shape contexts [BMP02] and a regression method [AT04a] to map measurements to 3D poses. The effects of the use of rotation invariance shape contexts are analysed. The key development is a combination of multiple views. Experiments comparing the performance of single and multiple views indicated that the proposed multiple view extension improves accuracy and reduce the complexity of the regressor. This work has been published in [dM06].

Conclusions are drawn at the end of each chapter, but Chapter 8 draws the various threads together to provide a formal conclusion to this thesis and to list its main contributions. This chapter also points out new possibilities of research that are open as continuation of the work presented here.

But the thesis continues now in Chapter 2 by presenting a quite detailed survey of the recent research achievements in object, person, limb and hand tracking. The review is rather longer — actually, much longer — than that which is required to support the remaining chapters. (Indeed, at first reading of the thesis, it may be best to continue to Chapter 3, and return later.) The motivation for writing at length is touched on now.

1.5 A postscript on available subject reviews

Although there are a couple of reviews in the literature for 3D hand pose estimation — Wu and Huang [WH99b] review model-based methods for hand analysis and animation in HCI and Erol et al.'s review [EBN$^+$05] also includes discriminative and mapping-based methods for pose estimation from a single frame — there is a lack of archival review material on methods to track articulated objects.

Some useful material is contained in the reviews of the broader fields mentioned below, but it is diffuse. In the area of Human-Computer Interfaces (HCI), Porta's paper [Por02] provides a comprehensive survey on the use of vision in HCI. Turk's slight later review [Tur04] targets non-expert readers. Jaimes *et al*. [JS05] cover multi-modal HCI, including all the senses. Duric *et al*. [DGH$^+$02] review methods regarding to three sets of criteria: the task that they are focused on: detection, tracking or recognition; the models used to represent humans; and the method for pose update used; before focussing on adaptive HCI.

For the field of Human motion capture (HMC), Aggarwal and Cai [AC97] present taxonomies of past research subdividing them on motion analysis of human body parts, tracking of human motion without using body parts, and human activity recognition. A similar subdivision of the research literature was adopted by Gavrila [Gav99]. Moeslund and Granum [MG01] present a more comprehensive survey, proposing a taxonomy based on functionalities, i.e., they describe methods that combined would compose a complete HMC system: subject modelling and initialisation, segmentation and tracking, pose estimation, and action recognition.

When reviewing Gesture Recognition, Cassel [Cas98] sought to provide a common framework for the generation and interpretation of spontaneous gesture (including facial expressions) in the context of speech. Watson [Wat93] presents the earliest survey and cites appearance-based methods based on classification or tracking of deictic gestures. The survey by Pavlovic *et al*. [PSH97] was the first to discuss view-based vs. 3D model-based methods. They support 3D model-based methods, but admit that this approach is more challenging, which explains why view-based methods have been explored in more depth. Parts of this survey have been updated in [WH99c] and [WH01].

2

Articulated object tracking and pose estimation: a literature survey

2.1 Introduction

Research on vision-based sensing of hands was first reported in the early 1980s, but the last decade has seen a burgeoning of the field, driven, of course, by cumulative progress in vision algorithms, but also by advances in computing and camera hardware and the perceived value of potential applications. This chapter presents a detailed survey of hand pose estimation and tracking methods. A number of methods for human motion capture have also been included, because of their parallels with hand tracking.

The review has been divided in three sections. Section 2.2 is concerned with 2D methods, and starts by reviewing image-based methods that run “model-free” in Section 2.2.1, methods based on exemplars are reviewed in Section 2.2.2, and those that run with with 2D models are considered in Section 2.2.3.

Section 2.3 is concerned with those methods where the model is built in 3D. There is a considerable number of methods that use partial 3D models, and these are outlined in Section 2.3.1, 2.3.2 and 2.3.3. Methods that include quite complete kinematic models are less varied, and these are examined in Section 2.3.4–2.3.7.

Section 2.4 of the review considers the intermediate class of methods which has come to prominence in recent years where 3D shapes are deduced directly from image appearance. High-level methods for action and intention recognition are not included, and little attention is given to gesture recognition methods. The survey concludes with a summary, a list of challenges that still remain, and some considerations as to likely future directions in the field.

2.2 2D methods

Two dimensional or image-based approaches are applicable to problems where it is enough to recover a two-dimensional description of the hand pose and a qualitative description of the gesture. The number of recognisable gestures is limited, but they can be very robust.

2.2.1 Model-free methods

Some methods do not model the hand's appearance. Instead they track a cloud of moving features or blobs that are likely to be hands, and attach meaning to them [FAB$^+$98, Que95, WADP97, BPH98, WLH00]. One of the most robust model-free methods is that of Kölsh and Turk [KT04b]. It uses a set of KLT feature trackers (named after Kanade, Lucas [LK81] and Tomasi [ST94]) which is initialised in the hand region (see Figure 2.1). The position of these features is bounded by the skin colour blob and a geometric constraint that prevents features being located too close to or too far from each other. The hand position is determined by the median feature among the set. This system is able to work at video rate, with performance superior both to that of a raw KLT tracker and to that of a mean-shift tracker [Bra98].

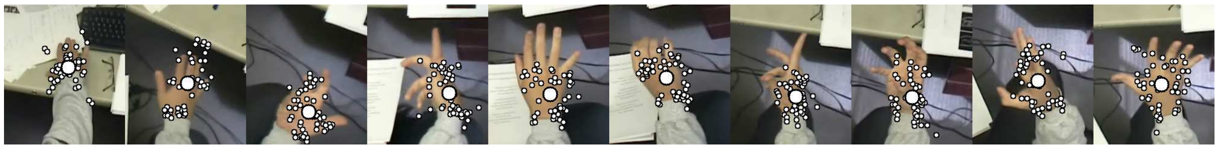


Figure 2.1: 200×230 pixel areas cropped from the 720×480-sized frames of a video sequence showing the tracking result with highly articulated hand motion. The cloud of dots represents the flock of features, the large dot is their mean. (From [KT04b], reproduced with permission.)

2.2.2 Exemplar-based methods

Exemplar-based representations are simple to create and are capable of representing highly nonlinear configuration manifolds. Exemplars can be thought of as minimally preprocessed select representatives of the training data itself, which together 'span' the range of the modelled entity [And01]. Some exemplar-based methods do not present any pre-processing step and use whole image patches to represent instances of the target object. For instance, Darrell and Penland [DP95] simply use correlation for gesture recognition. In some cases, even parts of the background are included. For this reason, such methods are usually memory-intensive. In most cases, accuracy and speed are achieved by using fast search methods, powerful classifiers and temporal information.

Efficient classification methods

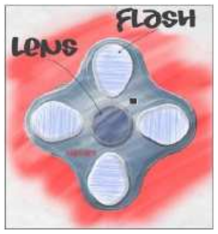
Gavrila and Philomin [GP99] use chamfer matching and a coarse-to-fine search in the image grid to speed up matching. In order to detect multiple body poses, a database of shape templates is partitioned into a number of clusters following their dissimilarity. Clustering is done recursively, leading to the creation of a tree of templates. Matching is then done by traversing the tree structures, from root to leaves, following the path with most similarity with the observed image. This prunes a large number of comparisons that would be done if exhaustive search was used. Pedestrian detection is performed at near video-rate, but the accuracy of this system is between 75-85% of detection rate.

Boosting [RO00] is a fast and powerful classification technique based on a weighted cascade of weak but quick classifiers. This combination provides accurate results with a good description of the decision boundary. Viola and Jones [VJ01] proposed a method for object recognition based on boosting of simple image features founded on Haar wavelets. Their results for face detection make this a gold standard method for this task. Although its application is quick, the training phase is computationally demanding. Variations upon the method have been applied to hand tracking, with nuances to improve the training time [KT04a], [JRM06], [WAP06]. Lockton and Fitzgibbon [LF02] applied this for real-time gesture recognition to replace keyboard and mouse.

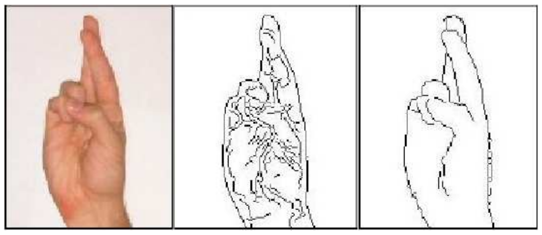
Using temporal information

A variety of techniques has been employed to model temporal sequences, including PNF (past, now, future) networks [PB98], tree-based search [Lyo02], finite state machines [HTH00] and, most commonly, Hidden Markov Models (HMMs) [WBAS03], probably due to its success in speech recognition.

Starner *et al*. [SWP98a] use HMM to represent a lexicon with four states to recognise some gestures of American sign language (ASL). The silhouette of the hands are obtained with skin detection and they



(a)



(b)

Figure 2.2: (a) Camera with multiple offset flash sources. (b) The letter 'R' in ASL recovered Canny edges and the depth edges obtained with the technique of Feris *et al*. (©[FTR+04], reproduced with permission.)

are described using a 16 element feature vector built from basic moment-like blob measurements. In [TB02], Toyama and Blake show how two dynamic processes (global motion and shape changes) can share the same joint observation density provided by the chamfer distance. This leads to an attractive homogeneity of description and implementation. The drawback is that this requires the use of a large particle set, which must simultaneously represent hypotheses of both shape and position. Fei and Reid [FR04] argue that in many applications these processes can (or even should) be decoupled, potentially leading to a more economical use of particles and hence to greater efficiency and reliability. They propose a method for the analysis of complex hand motion that assumes that the hand motion consists of two independent components: cyclic shape variations and hand region motion. The former is modelled by an HMM using silhouette moments, the latter is a particle-based colour region tracker. Each maintains a full PDF of its respective component of the motion, and they interact via an importance sampling function.

Matching image information

This review turns to consider how the observed image is matched with prior knowledge of the hand's appearance. The simplest method in terms of implementation is template matching or normalised cross correlation as used in [FAB+98]; but it is very common to use shape descriptors [dCJ01] for hand pose classification [YI98], image moments [Hu62] being popular descriptors for this task [FAB+98, Lyo02]. Other descriptors employed for matching are (i) those based on the analysis of the curvature of the silhouette contour [HSS02] and (ii) polar histograms [OS03a] computed from the centre of the silhouette of the hand [WAP06] [FTR+04]. (The second reference uses a multiple offset flash gun to help recover depth edges in hand images. The illustration and results in Figure 2.2 are a good example of the difficulty caused by partial occlusion.) In [TSTC03], Thayananthan *et al*. compare two methods for matching of Canny edge images. One method is based on shape context and the other combines an exemplar-based detector based on chamfer matching with an optimisation method for model-fitting. They conclude that chamfer matching outperforms shape contexts, but chamfer matching requires a high number of templates for matching.

2.2.3 Trackers with 2D object models

Prior knowledge can be exploited to obtain a more quantitative description of the hand shape. Model-based methods use a description of possible hand shapes and tracking is performed by matching the model pose with the observed hand images.

Active contours [BI98] as well as deformable templates [BH95, HCT95, BS02, Bow99, KH95, BF95, TLC+98] have been used to model the hand's appearance in images, methods robust to small variations in pose and shape of the hand. Deformable 2D templates comprise points on the outline of the hand that are used as interpolation nodes for an approximation of the outline. The template sets and their corresponding variability parameters are stored, and matching involves minimising the summed squares of differences between points from the image silhouette and the templates. This is a simple and successful method if the original viewpoint is maintained, or if characteristic views are available. Triesch and von der Malsburg [Tv01] employed a variation on this approach, elastic graph matching, in which the hand image is represented as a labelled graph. The distribution of nodes describes postures in 2D.

Another image-based method to track an articulated model was proposed by MacCormick and Isard [MI00]. With an articulated model and active contours to segment the index finger, they tracked four degrees of freedom, namely planar translation, orientation of the thumb and of the index finger, using CONDENSATION [IB98, IB96]). Unlike model-free and classification-based methods, this system recovers continuous parameters rather than recognising gestures from a discrete "vocabulary". In [MI00], the authors show the application of this tracker to implement a virtual workspace with a more natural interface for drawing objects, as shown in Figure 2.3.

For whole body tracking, more complex articulated models have been used recently assuming that each limb part can be modelled in the image by a rectangular shape. Bregler *et al*. [BOC+98] model the body segments with a multi-dimensional mixture of Gaussian blobs, modelling motion, shape and grey

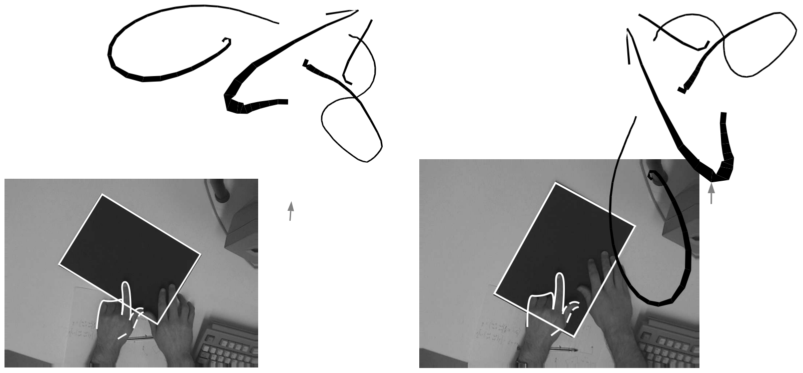


Figure 2.3: The virtual workspace described in [MI00]: the thumb activates the pen and the orientation of the index finger continously controls the thickness of the lines; the black piece of paper (tracked with a simple Kalman filter) controls the position and orientation of the whole drawing image. (©[MI00], reproduced with permission.)

level distributions. The blobs are initialised using motion coherence likelihoods, based on optical flow. For a two parts articulated body, they use Expectation-Maximisation to estimate the pose with simple kinematic priors to constrain the blob estimation. In their "cardboard people" tracker Ju *et al*. [JBY96b] use three different views, viz frontal, oblique and side on, to design two-dimensional templates that represent projections of the object in each view. They track walking motion, but it is assumed that the orientation of the object does not change along the sequence. Lu *et al*. [LPV06] use a planar layered model capable of handling occlusion for gait analysis. For each body part, it actively selects which side is more likely to have reliable edges for walking movements (avoiding, for example, edges between the legs because they are often perturbed by clothing). Local tracking of body parts is based on mean-shift, and strong motion priors (such as arms move in opposition to thighs) are included. Good results are reported for examples which conform with these priors.

In [MR98] Morris and Rehg model the projected motion of an inter-joint link in the scene as affine flow patches with imposed kinematic constraints. This is similar to Ju's model [JBY96b], but with fewer parameters and a more direct connection to the underlying articulated motion. These two approaches are compared in Chapter 6.

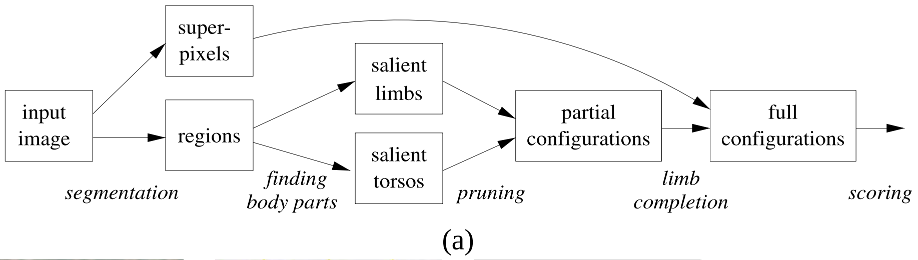
The above methods give continuous pose estimates in 2D, which is not always required in problems related to registration of articulated objects. Felzenszwalb and Huttenlocher [FH00, FH04] proposed the use of pictorial structures for object recognition, based on dynamic programming with discretisation of the parameter space. The algorithm searches the parameter space to minimise a cost function that combines the matching score of the object parts with kinematic constraints. The matching score is based on how well rectangles can fit areas segmented by background subtraction. The joints have a spring-like model and the kinematic constraints try to minimise their degree of distortion from an up-right pose.

Ronfard *et al*. [RST02] built a similar system, but they replace the rather simple part detectors with dedicated detectors learned for each body part using Relevance Vector Machines (RVMs) [Tip01], which are support vector machines-like classifiers that offer a well-founded probabilistic interpretation and improved sparsity for reduced computation.

Ramanan and Forsyth [RF03] use clustering to learn the appearance of objects that move in a video sequence. This approach, called *foreground enhancement*, is different from traditional background subtraction since it is used to learn the appearance and not to find people. Therefore, once the appearance is known, they can track people who initially stand still, so long as they move at some point. They use a probabilistic graphical model to locate and track multiple people in video sequences.

Kumar *et al*. [KTZ04] extend Felzenszwalb and Huttenlocher's approach [FH04] by using a complete graph model, rather than a tree structure. In order to estimate the maximum *a posteriori* estimate of the pose and shape parameters, a loopy belief propagation method is used, which is a message passing Viterbi-like algorithm for graphs with loops. The authors show that this gives more constraints for the pose estimation and better results than a tree structure.

More robust to unconventional human body poses is the method of Mori *et al*. [MREM04]. It uses a method for segmentation that gives the *probability boundaries* based on brightness and texture. This is applied with two different parameters: one that segments regions of the image large enough to be likely to contain half limbs and torso segments, and one that super-segments the image, giving *super-pixels*. Next, a method for detection of salient body parts is applied to the large segments. This method is based on four cues: contour, shape, shading and focus. These cues are combined and the regions with highest score are selected and combined using constraints on relative widths, lengths, adjacency, and similarity of clothing. The output of this system is a ranked shortlist of possible configurations of the human body in the image. Each pose configuration is obtained from the association of different segments of the images, which enables the computation of a body segmentation (see Figure 2.4). The main drawback of this system is the dependence on its training set for robustness. Furthermore, the design of this method was driven by its data: baseball players images. This allowed the use of specific features, such as the symmetry and regularity of the players' uniforms, which, in most of the showed images, are highly distinguishable from the background.



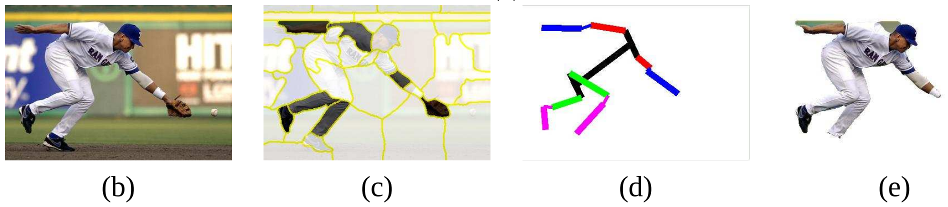


Figure 2.4: **(a):** Data flow the algorithm of Mori *et al*. [MREM04]. **(b-e):** Selected result from the short-list of final configurations: (b) input image, (c) candidate half-limbs, (d) extracted body configuration, (e) associated segmentation. (©[MREM04], reproduced with permission.)

The method above estimates pose from single images independently. This review now turns to methods that exploit spatio-temporal information.

Yacoob and Davis [YD98] use an approach for learning and estimating temporal flow models from image sequences. Such models are created by applying principal component analysis to time sequences of parametric models of body part motion. These observations are obtained using the "cardboard body" of [JBY96a]. This approach bridges the gap between traditional instantaneous optical flow estimation and multi-frame motion estimation. The learned motion models are used in a spatio-temporally constrained image-motion formulation for simultaneous estimation of several rigid and non-rigid motions.

Wu *et al*. [WHY03] also explore temporal priors, but instead of constraining a spatio-temporal manifold, a motion filter is used. Articulated objects are not modelled as single objects with low-DOF joints. Instead, each rigid part k is measured ($\mathbf{z}_k$) in the image independently and the pose of each part is redundantly described by its pose $\mathbf{x}_k$, independent from the other parts of the object. The measurements of each part give a local likelihood $p_k(\mathbf{z}_k|\mathbf{x}_k)$ for the pose. The local prior $p_k(\mathbf{z}_k)$ can be obtained using temporal information, thus it depends on previous measurements. This is combined with neighbourhood

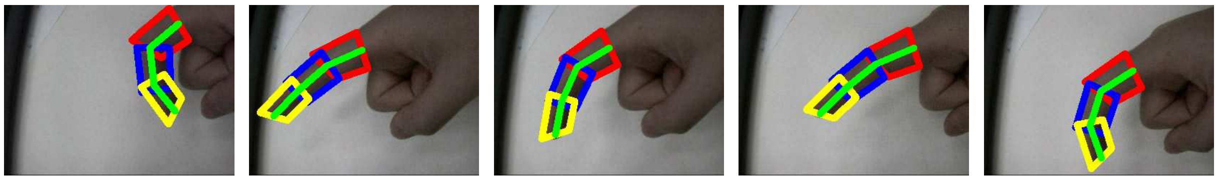


Figure 2.5: Mean field Monte Carlo tracking of 3-part finger presented in [WHY03]. (©[WHY03], reproduced with permission.)

priors, which constrain this part to be connected to its neighbours. The system is modelled with a dynamic Markov network, which serves as a generative model for the articulated motion. In their most challenging experiment, a 10-part articulated body was tracked at 0.56 frames/second, using 200 particles per part, but the lack of different texture on hand parts make this difficult for hands. Figure 2.5 illustrates the results for a 3 parts finger.

2.2.4 Discussion

2D image-based methods simplify tracking and pose estimation by focusing on motions that are parallel to the camera plane and by restricting the appearance changes. A considerable robustness is achieved with methods that model (or are invariant to) scale and shape changes, but they either do not provide enough information about the hand pose (*e.g.* model-free methods) or they are view-dependent. For instance, methods that use image-based articulated models can struggle with fingers pointing at the camera. Such methods have not successfully been applied to full-DOF whole hand tracking because the lack of strong textures challenges their effectiveness.

2.3 3D model-based tracking

The models in the previous section were two dimensional. This review now considers methods based on 3D models, starting with those that employ a simple rigid model to track hands, moving on to methods in which specific image features locations are used, and ending with those that model many or all of the available degrees of freedom.

2.3.1 Tracking hands in 3D without estimating fingers joint angles

Several early 3D hand trackers used rigid hand models, with the intention of using the hand as a 3D pointer or mouse. In [COK93] Cipolla *et al*. tracked four coloured markers on a hand (three on the ends of fingers) to recover 3D orientation which was then used to control the orientation of a graphics model. Without using colour markers, Cipolla and Hollinghurst [CH98] tracked the thumb and index finger using active contours. The intersection between the finger line visible from the two cameras and a ground plane was computed, allowing for simple interaction with a robot. Bretzner and Lindeberg [BL98] used three cameras to track three fingers to establish both the position and orientation of a rigid hand in 3D. An ingenious alternative to the using multiple views was the use in [SK99] and [SLM$^+$02] of a single camera with multiple light sources to cast shadows onto a planar surface

In [OZ00], O'Hagan and Zelinsky propose decoupling gesture recognition and pose estimation. Their system assumes a 3D rigid planar hand model. A curvature analysis allows features on the outline of the hand silhouette to be selected in each image of a stereo pair, allowing the planar pose of the hand palm to be determined. An image-based classification is combined with 3D rigid hand tracking to recognise gestures. More practical results are shown by Segen and Kumar [SK98, SK00] in a video-rate application. As in O'Hagan and Zelinsky, gestures are recognised using peak and valley detection on the hand silhouette. A finite state machine analyses movements to refine gesture recognition. The same image features are also used to estimate 3D position, azimuth and elevation for both the index finger and the thumb. This system is applied to 3D scene composition and navigation.

Sato *et al*. [SSK01] use skin detection to segment hand blobs in a two cameras system, allowing the 3D position to be computed through triangulation of the centre of the hand. The orientation is then determined using the principal axis of the hand and the left and right end points. A small set of gestures is recognised using a neural network applied to segmented, normalised and sub-sampled hand images.

2.3.2 3D tracking of an articulated arm with fixed basis

A number of authors have modelled and tracked hands as a rigid extensions of the forearm, articulated at the elbow and shoulder. Gonçalves *et al*. [GdUP95, BGP96] developed a monocular system capable of tracking human arm in 3D where the limbs are modelled as truncated cones, the shoulder is a spherical joint and the elbow is a planar joint, giving the model 4 DOF. The image measurements are obtained by thresholding and smoothing the image, and the method performs 1D searches for the highest gradient perpendicular to the projection of each limb segment (forearm, arm and hand tip). Only five control points per segment are searched. Tracking is performed by a recursive estimator that performs random walk in the spherical joint velocities and uses the Extended Kalman Filter. Ambiguities are avoided by constraining angles. Tracking was achieved at 11 Hz (in 1995, when typical processor speeds were 100 MHz) and the standard deviation of the estimates of hand tip position is some 1% of the distance between the camera and the user's hand.

Vogler and Metaxas [VM96] use three near-orthogonal views and impose priors on the human body shape. Deformable silhouettes are used and heuristics are applied to locate the position of joints. For tracking, the method actively selects of the best viewpoint for each body part in each frame [KM96] and, once selected, planar rotations and translations are estimated to drive updates to the 3D model. A Kalman filter is used for prediction and gesture recognition is performed using an HMM to recognise a set of 53 gestures from ASL. (The word accuracy achieved was of 88% for the 3D context-dependent experiments with 456 testing gestures. The authors made use of a commercial HMC system based on magnets interchangeably with their vision-based method.)

2.3.3 Using finger tip locations

If a reliable estimation of the position of the finger tips is available, it is possible to obtain solutions for hand pose using inverse kinematics [Cra89]. Both single [CGH02], [LH00] and multiple cameras [Lie04], [Reh95] have been used (as, incidentally, has active illumination [SKK00] and laser tracking [PCI03]), and a wide variety of methods have been proposed to detect and locate fingertips. These include (i) coloured markers [Lie04, CGH02, LH00, RL00]; (ii) circle detection by fitting [vHB01] and Hough transforms [CC03]; (iii) line detection with the Hough transform [Ahm95], [GW00]; (iv) curvature analysis [YI98]; (v) correlation [Reh95]; and (vi) trained neural nets [NR99]. To eliminate ambiguities constraints imposed by limitations of muscles and tendons must be included in the model. Most use hard-coded linear dynamic constraints between joint angles to reduce the dimensionality [LH98, CGH02][1]. Motion priors are used to predict over periods of occlusion [LH00, Lie04, RL00].

Perhaps the main advantage of these methods is that they do not require a model of the hand to be back-projected into the image. However, if no colour markers are used, detection of fingertips is very challenging particularly if the image region around a fingertip is skin coloured, as is common situation

[1]For instance, the relation between the proximal inter-phalangeal θ_1 and the distal inter-phalangeal θ_2 joint angles is modelled as $\theta_1 = 2/3\theta_2$. The abduction dofs are often ignored as in [CGH02].

when the fingers are bent. Even when markers are used, the measurement of the palm is made unreliable by skin movement.

2.3.4 Marker-less articulated tracking using complete 3D models

The methods described on this section onwards involve the use of a complete 3D model.

Symmetric tracking with a kinematic chain

In the early 1990s Rehg and Kanade [RK94, Reh95] developed the first system to track unmarked hands using a realistic (27 DOF) 3D kinematic chain at near video rates. Finger phalanges were modelled as simple cylinders, fingertips as halves of spheres, and the palm as a couple of planes linking two cylinders.

Two feature extractors to measure the sum of squared differences (SSD) were presented: deformable templates registration and point and line features. In template registration, the cost function is based on intensity errors used to measure the geometric misalignment between an input image and the image predicted by the projected kinematic model. Each finger is described by a planar template deformed with an affine transform to approximate the projection. Templates provide a useful level of generality, and make it possible to exploit arbitrary texture cues. But for a specific object like the hand, the constraints provided by the template matching can be approximated by purely geometric error functions involving point and line features [RK94].

Point and line features tracking is performed by projecting the middle axes of the truncated cylinders onto the image and searching for edges in directions perpendicular to the projected segments. Search for edge in the finger tips is also performed. The significantly lower computational cost of computing point and line features makes on-line tracking possible. The residual error between the estimated position of the features and the actual located features are combined and minimised using a weighted Gauss-Newton iterative method to estimate the state update as follows:

$$\mathbf{q}_{k+1} = \mathbf{q}_k - [\mathtt{J}_k^\top \mathtt{J}_k + \mathtt{S}]^{-1} \mathtt{J}_k^\top \mathtt{R}_k, \tag{2.1}$$

where $\mathtt{J}_k$ is the Jacobian matrix for the residual $\mathtt{R}_k$, both of which are evaluated with the state vector $\mathbf{q}_k$ and k is the iteration index. $\mathtt{S}$ is a constant diagonal conditioning matrix used to stabilise the least squares solution in the presence of kinematic singularities.

The method was implemented on multiple processors using separate frame grabbers for the two cameras and a separate computer to render and display the estimated model, resulting in a 10 Hz tracking of 19 degrees of freedom (where the middle fingers were not tracked) and 7 Hz on all 27 degrees of freedom. The on-line version did not include Rehg's method of occlusion reasoning, which was restricted to off-line because of its computational complexity. Although the palm is modelled, for simplicity its projection is not used for tracking.

Lu *et al*. [LMSO03] describe a method for hand tracking using a single view from a motion sequence. A combination of spheres and truncated cones models the appearance of each part of the hand. Three image cues are used for tracking: edges, optical flow and shading information. Since there is not much image features on bare hands, the standard optical flow could not provide good results. As a solution, optical flow and shading information are combined using a generalised version of the gradient-based optical flow constraint that includes shading flow, i.e., the variation of the shading of the object as it rotates with respect to the light source. Similarly to Rehg's work, 2D image feature discrepancies drive changes in 3D pose via the Jacobian. To combine the multiple cues, Lu *et al*. use Lagrange multipliers. An iterative method to impose joint constraints is also described. Basically, once the pose estimation results on some of the joints moving further than its limit, the joint is fixed to its limit and a new solution is estimated with this joint modelled as a rigid object. Tracking at 4 Hz was achieved on a Pentium 4 1 GHz cpu.

An alternative to Rehg and Kanade's notation was proposed by Bregler and Malik in [BM98]. Instead of using standard full projective geometry, scaled orthography is used. Thus the effects of changes in distance from the camera are compensated by changes in scale of the object. This seem to be appropriate for problems with unknown camera calibration and objects far from the camera, as it is common for full body tracking. The image measurements are based on comparisons of internal pixels of warped image of object parts. The method is formulated using Lie Algebra, i.e., the motion between each pair of object parts is represented using a combination of the exponential of the canonical matrix of the coordinate frame of each DOF. These are used to build a linear relationship between instantaneous motion and pose change, allowing to obtain a least squares approximation of the pose update (exponential twist) for articulated objects, given the image measurements. As shown in Chapter 6, this turns out to equivalent to the motion screw obtained by standard Jacobian-based methods for articulated object pose update. The experiments in [BM98] show successful tracking of a 6 DOF human body using a single camera and of a 19 DOF body using three cameras and a well-known “Eadweard Muybridge” sequence. The frame-rate achieved is not reported.

Whole body tracking using a quantised feature space

In their influential work, Gavrila and Davis [GD96] modelled the human body using superquadrics. Four widely spaced cameras were used, and the model projected into each of them under perspective. A fitting cost was defined by chamfer matching in a filtered and background-subtracted edge image, and coarse-to-fine search in parameter space used to determine the best-fitting quadrics,

A local best-first search was used for pose update. However, using 22 dimensions per human makes the the search space large, and brute-force search daunting. Instead they proposed a search space decomposition in which the parameter space was recursively partitioned in a tree-like structure of subsets of parameters. At the leaf level were single parameters that were optimised individually, but the whole parameter set was used to verify the error. Once a parameter was optimised, it was fixed while the remainder were optimised. The parameters that have not been searched yet keep the predicted values from the previous iteration. This is an asymmetric search method, and different results can be obtained if different orders are used. The authors’ preferred order was persons; followed by head/torso position and inclination; torso twist; then arm pose.

The method worked well provided the image conditions were made benign — the subjects wore tight-fitting clothes of contrasting colour and the motion was straightforward. Tracking more complex motions, such as their tango sequence, required manual intervention.

2.3.5 Model refinement

In order to adapt the hand model to different users, some researchers have also optimised the “static” body parameters. Lu *et al*. [LMSO03] for example refine the length and thickness of fingers while tracking. During the first frames of a sequence, and after pose updates, the residual errors in edges and optical flow are accumulated and used to modify the hand shape by anisotropic scaling. Bregler and Malik [BMP04], show that the state space of their earlier motion tracking framework [BM98] can be extended to also optimise over the kinematic model, and over the complete image sequence instead of just image pairs. The twist (state of the pose parameters) is kept fixed to the values obtained by the tracker and the equations are rearranged to optimise the length of each link of the articulated body. Based on Tomasi-Kanade's factorisation method [TK92], Bregler's system of equations is iteratively factored to optimise the articulated model and update the pose and shape parameters along a video sequence. A more advanced model is that of Plänkers and Fua [PF02] who refine meta-balls (generalised algebraic surfaces defined by a summation over n 3D Gaussian density distributions) attached to an articulated skeleton. The meta-balls simulate the gross behaviour of bone, muscles and fat tissue. The model is projected into the images and its silhouette is extracted. Tracking is performed in four steps (i) The silhouette of previous frame serves as initialisation for current frame; (ii) Optimise using active contours on disparity-filtered gradient image; (iii) Refine the body model to stereo data constrained by current silhouette estimate; and lastly (iv) Optimise the silhouette of the fitted model using active contours.

The use of a model that accurately reproduces the objects appearance must have a positive effect on tracking precision. However, there has been no analysis of whether, when resources are finite, such improvements compensate for the extra computational effort necessary for update and projection of a detailed model as it moves.

2.3.6 Motion filters

Motion filters have been used in many hand tracking methods to smooth the pose estimate and to provide predictions that improve the reliability of the tracker. Some have already been mentioned in passing.

Methods based on the Kalman filter

Shimada and Shirai [SS96] use the Extended Kalman filter (EKF) for monocular hand tracking in 3D and also allow model refinement by including the length of finger parts in the state vector. First, the best fitting solution is obtained with EKF and then this solution is modified applying inequality constraints based on human hand physiological restrictions. In cases where multiple solutions satisfy the constraints, multiple hypotheses are generated (based on symmetry w.r.t. the image plane) and their fitness is evaluated. This system was only evaluated using simulations. Wachter and Nagel's persons tracker [WN99] uses an Iterative Extended Kalman Filter (IEKF) which consistently integrates edges and image texture cues for the pose update.

Stenger *et al*. [SMC01] use an Unscented Kalman Filter (UKF) [JU97] to update the pose of their model, which like Rehg's has 27 DOF, but it is built from 39 truncated quadrics (Figure 2.11), giving, of course conic projections. The hand dynamics are modelled using position, velocity and acceleration. The UKF is found to be more tolerant of non-linearities than the EKF, and permits higher frame rates than more sophisticated estimation methods such as particle filtering.

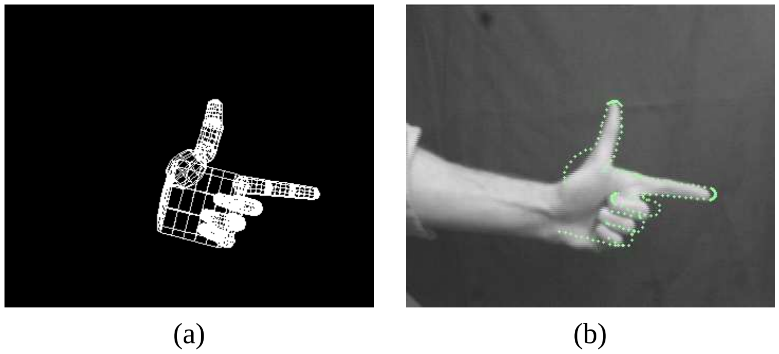


Figure 2.6: **(a)** wire frame of the 3D hand model used by Stenger *et al*. [SMC01]. **(b)** Projection of this model on the input image during tracking. (©[SMC01], reproduced with permission.)

Stochastic and multiple hypotheses search strategies

The challenges of unconstrained tracking and 3D tracking from monocular vision have lead to the research in methods to avoid local minima caused by ambiguities and configurations with singularities.

Deutscher *et al*. [DNBB99] have demonstrated that probability density functions (PDFs) for kinematic variables such as joint angles are actually non-Gaussian. This tends to happen particularly often in joint angle PDFs near their end-stop values and close to singularities where the kinematic chain lies in physically distinct but visually indistinguishable configurations. Their solution for human body tracking was to use CONDENSATION. Sidenbladh *et al*. [SBF00] presented a method similar to that of Deutcher's, but they use limb texture in addition to edges alone.

Another stochastic solution was proposed for hand tracking by Nirei *et al*. [NSMO96]. Given a rough initial estimate obtained by mouse clicks, a Genetic Algorithm was used to minimise the estimation error of optical flow and maximise the overlap between the projected model and silhouette images using the chamfer distance. They then applied Simulated Annealing to refine the pose estimate. The results were not obtained in real-time (unsurprisingly!), but they demonstrate that all the fingers could successfully be tracked in a short video sequence.

Stochastic tracking frameworks such as CONDENSATION are capable of dealing with complex PDFs and avoid local minima, but the curse of dimensionality threatens this approach. The minimum number of particles required for successful tracking is exponentially proportional to the dimensionality of the problem [JDM00]. One of the issues that make CONDENSATION computationally expensive is the definition and evaluation of the likelihood. Deutscher, Blake and Reid [DBR00] address this problem by developing the Annealed Particle Filter (APF) which uses a weighting function to approximate the likelihood. This weighting function is easy to be calculated and, unlike CONDENSATION, the perturbation of the particles always decreases with time. This allows the use of much larger particle distributions with less computational effort. Davison, Deutscher and Reid [DDR01a] demonstrated the application of this algorithm for Human Motion Capture for character animation (see Figure 2.7). The particle distribution is used by the APF to evaluate several parameters of the weighting function in attempts to find a value that minimises it. Clever search strategies are needed to help particles to locate the global minimum of the weighting function to overcome the complexity of the search space. APF tends to be rather wasteful of computational resources in the searching of configuration space. At each time step, the APF must add a noise vector to the particles. The noise has to be large enough to lead to a search that covers a sufficiently large volume of the configuration space. This improves the tracking results, but many particles are wasted in randomly generated configurations.

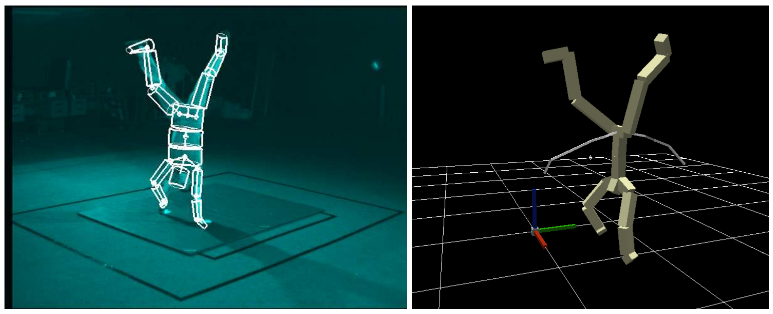

(a) (b)

Figure 2.7: **(a)** Projection of the 3D body model on the image of one of the 3 view from a handstand video sequence used by Davison *et al.* [DDR01a]. **(b)** The virtual character on the pose obtained by the APF algorithm. The curve shows the trajectory of the base of the subject's spine. (©[DDR01a], reproduced with permission.)

To address this problem, Sminchisescu and Triggs [ST01b, ST01a] adopted a covariance weighted sampling in which a covariance matrix representing uncertainty is associated to each body pose hypothesis. This allows iterative generation of hypotheses that are less ambiguous, resulting in a more efficient distribution of the available tracking estimates. In parallel, the searching method of APF was improved by Deutscher *et al*. [DDR01b] by adding noise to each individual parameter of a particle in proportion to the variance observed in that parameter across the particle set. Another improvement was the use of a genetic algorithm-like particle crossover operator. This update on the algorithm lead to a 4-fold increase in processing speed.

An alternative bottom-up approach has been presented by Sigal *et al*. [SISB03]. They represent body parts individually and a stochastic algorithm places the parts randomly in 3D. A graphical model based on message passing and learning combines image measurements and spacial constraints and, in [SBR$^+$04], also temporal constraints. Bottom-up part detectors based on PCA of concatenated images (of multiple views) are used to aid detection of parts. They suggest their results improve on the APF because errors are not accumulated. However, although the optimisation searches for solutions that do not violate the constraints between body parts, these are not hard constraints and the system may provide impossible body configurations. (The stochastic method used is similar to that of [WHY03], mentioned earlier (page 19).)

Bray *et al*. [BKMM$^+$04] proposed the Stochastic Meta-Descent (SMD) method for hand tracking. It is a gradient descent method with local step size adaptation that combines rapid convergence with scalability and, as only a single hypothesis is considered, requires fewer samples than CONDENSATION and less computational power than the APF. Although SMD can avoid some local minima, it does not guarantee that the global minimum is reached. In [BKMV04], Bray *et al*. incorporated SMD within a Particle Filter to form ‘smart particles’. After propagating the particles, SMD is performed and the resulting new particle set is included such that the original Bayesian distribution is not altered. As a particle method, it maintains multiple hypotheses needed to cope with clutter and occlusion, but reduces the number of particles needed. Figure 2.8 shows an example of 3D recovery using (note) structured light.

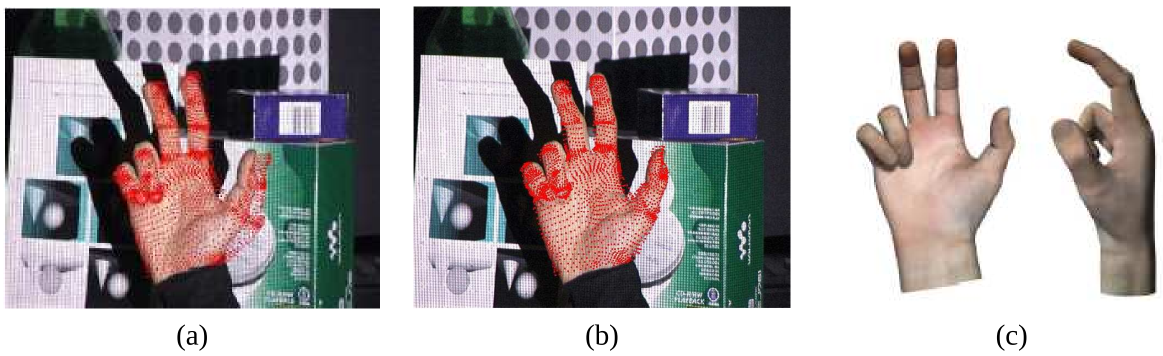


Figure 2.8: A frame from a video sequence used by Bray *et al*. [BKMV04]. (a) and (b) show mesh created by the structured light, the red dots show the projected model with the tracking result obtained using: (a) APF and (b) Smart Particle Filter (SPF). (c) shows two views of the 3D model whose pose was obtained using SPF. (©[BKMV04], reproduced with permission.)

A deterministic alternative

While much effort has been made to explore stochastic approaches for human body tracking, Sminchisescu and Triggs have begun to explore such spaces deterministically, considered a way of avoiding entrapment in local suboptimal minima [ST02] . They address this problem by building ‘road maps’ of nearby minima linked by *transition pathways* – paths leading over low ‘passes’ in the cost surface, found by locating the *transition state* (saddle points with 1 negative eigenvalue) at the top of the pass and then sliding downhill to the next minimum. Their results have shown that their algorithm can stably and efficiently recover large numbers of transition states and minima, and also serve to underline the very large number of minima that exist in the problem of monocular 3D model based tracking.

2.3.7 Using data-driven dimensionality reduction

The use of articulated models simplify occlusion handling and allows the description of a larger number of hand poses. Although there has been an agreement that a 3D hand model should have at least 26 DOF, it is also clear that the configurations of muscles and tendons of the hand constrain the range of motion of each joint. For example, the fourth finger can not be flexed naturally without influencing the pose of the middle and small fingers because of interconnection of tendons (see Figure 1.2).

Heap and Hogg [HH96] represented hands as surface meshes extracted semi-automatically from 3D Magnetic Resonance Images. Since this is not based on an articulated model, the number of DOF of this representation is huge, but they have shown that by applying PCA to the point distribution data, the shape deformation could be represented in a low dimensional space, as illustrated in Figure 2.9. For tracking, the outline of the hand mesh was projected into the image plane and edge measurements are used. The pose update was performed by solving a linear system of equations in least squares. Tracking could be achieved at 10 frames per second using a single camera, but ambiguous motions and self-occlusions were not successfully overcame.

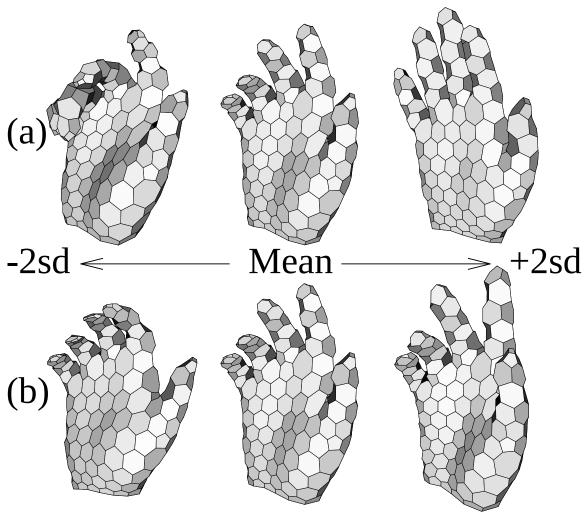


Figure 2.9: The first (a) and second (b) modes of variation of 3D hand point distribution model of Heap and Hogg [HH96]. (©[HH96], reproduced with permission.)

Wu *et al.* [WLH01], describe the space of possible hand configurations using a set of pre-defined states based on binary finger poses: fully flexed or stretched for each finger, giving a set of $2^5 = 32$ possible states of hand pose. Four of them were pruned because they were considered infeasible as most people cannot perform these hand poses naturally. Subjects were asked to move their hands to these 28 states while wearing a data-glove that acquires 15 DOF of the hand. Global position and orientation variations are not considered in this paper. The state space was reduced to 7 dimensions using PCA and it was demonstrated that the transitions between states follow linear paths in this space. Thus the hand pose is represented as a linear combination of the 28 states. An importance sampling approach is used for tracking. This shares some points with CONDENSATION, but the hypotheses are only generated along the nearest linear manifolds between two basis states, with some diffusion in the higher dimensional space. The 3D model is projected to the image as a cardboard model, and this method combines edge measurements with a comparison between the area of the projected model and the hand silhouette image to compute the likelihood of hypotheses. The experiments show that, in comparison to standard CONDENSATION in the $\mathbb{R}^7$ space, this approach provides better results and longer latency requiring an order of magnitude less samples.

Using the same dimensionality reduction method and a similar image likelihood function, in [ZH03] Zhou and Huang propose an eigen-dynamics analysis method to learn the dynamics of natural hand motion as a high order stochastic linear dynamic system. This is used to build a dynamic Bayesian network to analyse the generative process of an image sequence of hand motion. In the inference phase, the hand motion is decomposed into global motion and finger articulation, and an iterative divide-and-conquer approach [WH99a] is used to track the hand. For global motion, the iterative closest point algorithm is applied, and for finger articulation, sequential Monte Carlo is used to sample in the manifold spanned by the learned dynamic model. This system was tested with synthetic and real data and accurate results were obtained even with partial occlusion and cluttered background, but the experiments do not show how the system performs when there are both global and articulated motion at the same time.

Other relevant work in this area is that of Kato *et al*. [KCX06] who reduces the state space dimensionality to 5D using ICA (independent components analysis), showing that it performs better than PCA, and Grochow *et al*. [GMHP04] who represent the probability distribution function of the parameter space using a scaled Gaussian process latent variable model (SGPLVM) proposed in [Law04]. All the parameters of the SGPLVM are learned automatically from the training data. They show that it is possible to optimise the PDF to describe new poses in real-time for applications of inverse kinematics systems. Although this method allows to represent the PDF at a low dimensional space through a non-linear projection, it does not restrict the configuration state. Poses that are very different from those in the training set can still be represented, but they have a very low PDF. The authors have used this method to represent styles of human movements and proposed a method to interpolate between styles (Figure 2.10).

2.3.8 Discussion

The key benefit of model-based trackers is that they permit, in principle, a comprehensive exploration of the space of possible poses – they really do describe the detail of all the degrees of freedom. However, the quality of the measurements to drive the model depends on the similarity between the model and the

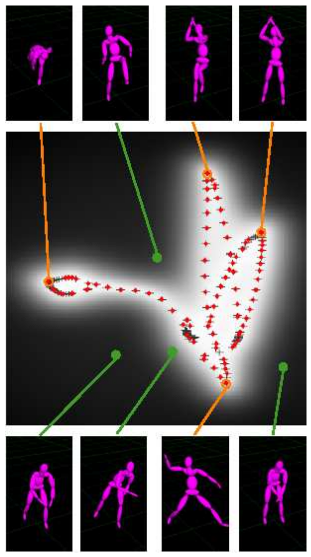


Figure 2.10: An SGPLVM latent space learned from a baseball pitch motion capture sequence by the method described in [GMHP04]. The learning process estimates a 2D position associated with every training pose; plus signs indicate positions of the original training points. Some novel poses are shown, illustrating that new poses extrapolate from the original poses in a sensible way. The grey level indicates the likelihood of each position in this planar projection of the state space. (©[GMHP04], reproduced with permission.)

real object (hand), and accurate models are not broadly available and certainly not cheap to compute.

With imperfect data it becomes hard to justify maintaining large numbers of degrees of freedom, and, even if we suppose perfect measurements, the problem of optimisation in a high dimensional space is significant. While much has been made of stochastic approaches, such spaces have also been explored deterministically [ST02], but results have underlined the very large number of minima that exist in the problem of monocular 3D model based tracking.

Good initialisation is important for these methods and, as they are incremental, a bad image or pose estimate can pollute all later estimates [Bra99]. Applications that target hand tracking that require a highly detailed description of the hand shape in 3D normally use images in which the camera is zoomed in the hands. In these cases, natural hand motions can be too sudden and fast in the images.

2.4 Direct 2D view to 3D pose transformations

This section approaches view-based methods to estimate 3D pose. These are also known as discriminative methods, and provide a bridge between 2D methods and 3D model-based methods. They extract measurements from images which are linked to the kinematic chain representation of the object in 3D.

For instance, template matching or global image descriptors are used. Once the measurements are extracted, a pattern recognition method is applied and a 3D pose estimate is obtained as output. Since the measurements can be extracted without requiring a prediction of the state, discriminative methods can be applied from static images for pose estimation or to initialise 3D trackers.

As usual in pattern recognition, these methods require training. The training set is often an extensive collection of possible hand appearances associated with 3D poses that generate them. The most practical method to obtain a comprehensive training set is based on creating synthetic images using a hand model that is rendered at a range of 3D poses. So to restrict the training set to natural poses, data acquired from glove aided motion capture is used. In the case of whole body, publicly available datasets of human motions can be used.

An advantage of discriminative methods is that they do not require computation of projection and occlusion handling at the inference phase, as this is implicitly done in the generation of training samples. Another advantage is that since the inference uses the training set, these methods naturally incorporate data-driven motion constraints. This also allows to reduce the dimensionality of the parameters space. The obvious disadvantage is that the range of possible poses is limited by the training set and extrapolations are not usually successful. The same is true about camera views which are not included in the training set.

Two approaches encompasses the discriminative methods: classification-based and mapping-based, further described in Sections 2.4.1 and 2.4.2, respectively.

2.4.1 Classification-based methods

In the classification-based approach for 3D pose estimation, a large discrete set of 3D poses constitutes the set of classes. The image measurements are evaluated as an input to the classifier and a 3D pose is obtained as output. The ability to provide a 3D pose output is the key difference between these methods and the 2D appearance-based methods. Furthermore, the fact that the training set contains pairs of measurements and 3D poses is used to aid the search.

Usually only one sample image measurement is available for each class of 3D pose, thus variants of nearest-neighbour classifiers are commonly used. The massive number of classes make this a formidable classification problem, which is eased by avoiding exhaustive search. This can be done by the following methods: (i) performing coarse-to-fine search; (ii) grouping the training set by similarities in appearance and in 3D pose parameters; (iii) using motion priors and time sequence information. Sample research works of these methods are further detailed below.

Coarse-to-fine search

In the coarse-to-fine approach of Athitsos and Sclaroff [AS03], two similarity measures: the approximate directed chamfer distance and the line matching cost. A large pose database was used, containing over 10^5 samples and the query could be made in 15 seconds, but the matching results were poor for real images: only 14% of the queries resulted in the best pose estimate, and even if all the 256 best matches are combined, the mean of correct matches among them is only 84%.

In [AASK04], an improvement was achieved by combining a large set of simple weak classifiers using BoostMap in the coarse search to select a subset of candidate matches. In the fine search, they used exhaustive chamfer matching. This reduced the query time to 2.3 seconds and improved that recognition rate to 95%. The quality of the classification results were judged by a human operator following the visual agreement between query and retrieval image.

Grouping training samples for tree-based search

One problem with exemplar-based matching is that the exemplar sets can grow exponentially with the number of degrees of freedom of the object. For this reason, Stenger *et al*. [STTC03] use a tree search (similar to Gavrila and Philomin's method [GP99]) which leads to a dramatic reduction in the number of comparisons required for matching.

Another improvement is that [STTC03] also uses the probabilistic tracking framework proposed by Toyama and Blake [TB02], so the search tree works as a dynamic Bayesian network for motion estimation, as illustrated in Figure 2.11. But unlike Toyama and Blake, Stenger *et al*. perform the probabilistic tracking in the space of the kinematic parameters of the articulated object (joint angles, rotations and translations). An advantage of using a parametric model to generate templates is that less storage space is required, because a finer pose estimation can be obtained by generating new templates on line, as the leaf is reached. Furthermore, two poses that are distant in the parametric space can be close to each other in appearance [TSTC03]. For example, the appearance of the outline of a flat hand with the palm facing the camera can be similar to that of the back of the hand facing the camera, but parametric-based clustering puts these two poses far apart.

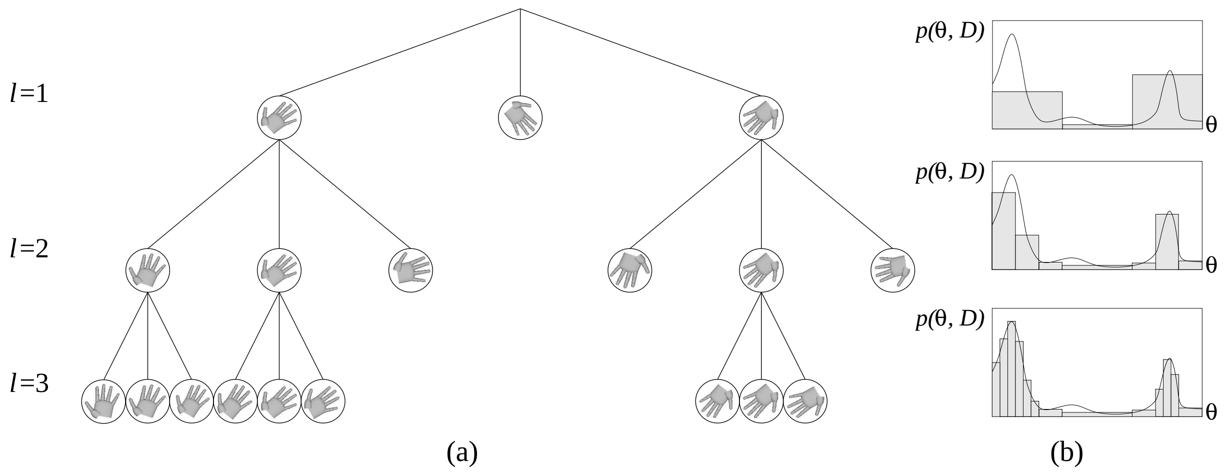
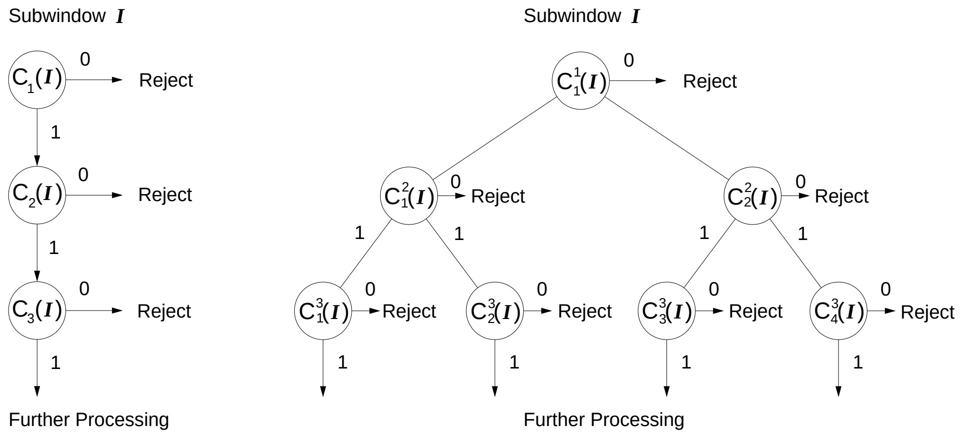


Figure 2.11: Schematic example of tree-based estimation of the posterior density, obtained from [STTC03]. (a) Each node is associated with a non-overlapping set in the state space, defining a partition of the state space (here one DOF of rotation of the hand). The posterior for each node is evaluated using the centre of each set, and sub-trees with low posterior are not further evaluated. (b) Corresponding constant posterior density of the state θ given the measurement D, and piecewise constant approximation obtained. (©[STTC03], reproduced with permission.)

To build the search tree, two methods have been evaluated Thayananthan *et al.* in [TSTC03]. The first is based on the hierarchical k-means algorithm, which partitions the space as a multi-dimensional Voronoi diagram and the cluster centres are used as nodes in each level of the tree. In the second method, the dimensionality of the parameters space is reduced using PCA and the resulting space is partitioned by a regular hierarchical grid where, again, the centres of the obtained hyper-cubes are used as nodes in each tree level. Their experiments show that the tree obtained by both partitioning methods give qualitatively similar results and search time of around 2s per query. However the training process with the PCA-based method is much faster.

Based on the above search tree idea, in [STTC04] Stenger *et al.* proposed an alternative classification method which uses a multi-class cascade of classifiers for shape template matching. Unlike the normal use of boosting for single object detection, the cascade of classifiers is arranged in a tree order to recognise multiple object classes (hand configurations) hierarchically, as shown in Figure 2.12. Each weak classifier is trained to detect a single hand pose, if that pose is detected, the search continues for child classifiers that do a finer classification. As usual with boosting approaches, the main advantage is its speed, but the number of classifiers needed grows exponentially with the dimensionality of the pose parameters, demanding much memory.



(a) (b)

Figure 2.12: (a) Standard single class cascade of classifiers to detect an object – each classifier has a high detection rate and a moderate false positive rate. (b) Cascade of classifiers C_j^i from [STTC04] for multiple classes j in a tree structure (with levels indexed by i) – similar objects are grouped together and the classifiers on the leaves recognise single objects. A binary tree is shown here, but the branching factor can be larger than two. (©[STTC04], reproduced with permission.)

Using spatio-temporal priors

The tree-based system of Stenger *et al.* [STTC03] can also incorporate temporal priors. Given the partition of the state space, the state transition distributions $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$ are modelled as first order Markov processes, and the transition probabilities are computed by histogramming transitions in the training set. This allows the computation of the temporal priors $p(\boldsymbol{\theta}_t)p(\boldsymbol{\theta}_t|\mathbf{D}_{0:t-1})$ (where D are measurements and $\boldsymbol{\theta}$ is the state vector) in a video sequence, which facilitates pruning the search tree, speeding up pose estimation as the motion follows a prediction. Although high accuracy can be obtained, the computational cost of this system is still too high for real-time applications. Using a relatively small range of hand poses, in [TSTC03] each frame takes 2 seconds to be processed in a 1 GHz Pentium 4.

An alternative is to use simpler image measurements and stronger temporal priors. In [FAK03], Fillbrandt *et al.* use a simple graph of transitions between states of the hand pose that restricts the search space, as only neighbouring states are checked. This graph was coded following transitions that happen in German sign language. A similar idea was implemented in [HSS02], where a simple moment-based descriptor is used. In the learning stage, if the distance between the image descriptor of the current image and the previous state is greater than a threshold, a new state is built.

Brand [Bra99] uses ten scale-invariant central moments on low resolution silhouette images. A dynamical manifold is used for inference of trajectories. This is defined as a locus of all possible poses and velocity configurations, embedded in a higher-dimensional measurement space. The inference is a search for a sequence of events (path on the manifold) that best explains a sequence of observations. To model manifolds a method identifies neighbourhoods where the relationship of position to velocity is roughly linear. Each neighbourhood is described with a multivariate Gaussian PDF. The manifold is approximated by an HMM with each neighbourhood Gaussian being the output of a hidden state, and a transition topology specially matched to the dynamical structure of the manifold. The HMM is learnt using entropy minimisation which, unlike previous methods, leads to a model that does not get “lost” at crossings and gives a more compact and accurate representation. To handle rotations around the gravity axis, the HMM is replicated once for each view, re-estimating the output distribution of each view-specific HMM. The 3D pose results are, in most cases, qualitatively close to the actual pose of the input image sequence. But with evidence as weak as image moments, the learned prior dominates the reconstruction, so input images of poses that are not in the training set result in the nearest 3D pose in the training set, which, in many cases, is not accurate.

Hee-Deok *et al*. [YPL06] proposed a framework of HMM models for whole body gesture recognition which recognises continuously, without the need of gesture segmentation. This framework initially has an array of HMMs for meaningless actions followed by an array of HMMs for gestures that are recognised and the whole scheme is closed as a loop. This paper concentrates on gesture/action recognition, rather than low level vision, so it is based on accurate human motion capture data obtained from the system described in [HKL06]. The dimensionality of the pose parameters is reduced using Fisher discriminant analysis.

In order to get a more continuous (in terms of inter-class difference in the pose output) estimation of 3D poses from a discrete set of training appearances, Shimada *et al*. [SKS01] combine an appearance-based discriminative method with a three-dimensional generative tracker. In the first stage, the silhouette of the hand (segmented by threshold) is described using the normalised eccentricity, which is a position and scale invariant descriptor. For rotation invariance, the maximal points are aligned with the training vectors for matching. Classification is sped up using an adjacency map and beam search, which is implemented in a distributed system. Once the appearance has been matched, its 3D pose combined is with the predicted pose in order to generate the next prediction using a motion model. The new prediction helps to speed up the appearance matching method by restricting the search area. This paper shows good qualitative matching results, which were obtained at video rate (30Hz) on a 6 node cluster, but it does not show results using the 3D motion prediction module.

2.4.2 Mapping-based methods

Mapping-based methods use pairs of image measurements and 3D poses to learn a continuous map between them. They can provide smooth pose estimation results rather than an estimate that is out of a discrete set. The results can be compared to an interpolation of the training data, but in some cases small extrapolations are also possible. Mappers can usually be implemented with parametric functions, which mean that their memory complexity is much lower than that of classification-based methods. In those cases, their evaluation does not require large numbers of comparisons, so their speed is also greater than that of classification-based methods.

Lin *et al.* [LWH01] modified the method of data-driven dimensionality reduction described in [WLH01] to create a mapping-based method. A feature vector built from measurements obtained from shape descriptors was acquired from each basis state in the training phase. In the application phase, this feature vector is acquired and its distance to each basic state is measured. This distance is taken as the weight of each state, determining a point in the state space, which is then lifted to the original 15 DOF configuration space to reconstruct the hand pose.

Shakhnarovich *et al*. [SVD03] introduced an algorithm that learns a set of hashing functions that efficiently indexes examples. The method uses local regression, which works as interpolated k-nearest neighbours and accounts for proximity not only in the 2D measurements, but also in the 3D pose parameters.

Prior to that, Rosales *et al*. [RASS01] proposed a system that uses a non-linear supervised learning framework, the specialised mappings architecture (SMA). As in Brand's paper [Bra99], image moments are used as measurements: seven real-valued scale, translation and rotation invariant Hu moments

[Hu62]. These are computed from hand silhouettes which are detected and tracked using a skin colour blob tracker that locates and refines the solution adaptively. A face detector is used to improve the initialisation of the skin colour detector. The pose estimation system consists of a set of 30 specialised forward mapping functions, each one built as a one hidden layer feed-forward network with 5 hidden neurons. These functions are learned using expectation-maximisation (EM). Each of them provides a mapping from the whole measurement space to the state space of 3D poses. To select the best solution, a feedback function takes the estimated pose, renders the 3D hand model and generate image measurements that are then compared with the input data. This method was evaluated quantitatively with a database of synthesised images generated using ASL gestures rendered at several orientations varying pan and elevation (the hand pose is described using 22 joint angles and two orientation parameters). This added up to 300,000 synthesised images, of which 8,000 were used for training and the rest for testing. The reported mean error was very small ($\approx 1^\circ$ to 3°), but the standard deviation was large enough to provide results that do not match the input ASL gesture entered. Qualitative results were also shown using real hand images.

In [MM01, MM02], Mori and Malik used shape context matching [BMP02] to locate the centre of limbs joints. The 3D pose is then estimated by using Howe et al.'s method [HLF99]. This is a Bayesian learning framework to recover 3D pose from known joint centres based on a training set of pose-centre pairs obtained from re-synthesised motion capture data.

A global image descriptor that is a simplification of shape contexts is used by Guan *et al.* in [HGT06], where the multi flash approach of [RTF+04] and [FTR+04] provides a clean depth discontinuity map, so the shape contexts describe a virtually noise free hand edges image. The mapping method used is based in self-organising maps.

In [AT04a], Agarwal and Triggs use a 100D global image descriptor based on a histogram of shape contexts of the silhouette contours. A human body model with 55 DOF is used to render training images and a regression-based method was used to learn the relation between image measurements $\mathbf{x}$ and 3D poses $\mathbf{y}$. Four regression methods were evaluated: (1) regularised least squares and (2) Relevance Vector Machine (RVM) [Tip01] regressors applied in both case to (a) linear and (b) Gaussian kernel bases. For synthetic images, resulting mean error in 3D pose estimation were: $(2a) > (1a) > (2b) > (1b)$, but the difference between the best and the worst of them is only less than 3°. However, the implicit feature selection obtained by RVM regression gives much more sparsity, reducing the complexity of the pose estimation process: only 6% of the training examples were retained. They have shown good quantitative results on synthetic data: mean estimation error of 6° over all joints for the Gaussian RVM (though many of the 55 DOF are inactive and it is not clear whether this is considered for this result). Only poor qualitative results were obtained for real images, and the demonstrative video shows reconstruction with many jitters along the sequence. Figure 2.13 illustrates a result of this tracker.

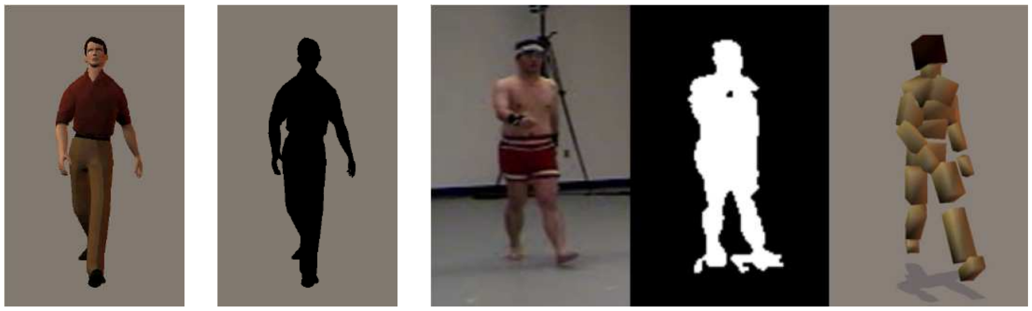


Figure 2.13: A sample result of [AT04b] **(a)** 3D human model used for training and its noise-free projected silhouette **(b)**, which is used for training the regressor. **(c) left**: a test image obtained from *http://mocap.cs.cmu.edu*; **centre**: background subtracted image segmentation result used for extraction of the shape contexts; **right**: reconstructed 3D pose. Note the difference between the subject and the training model and the amount of noise in the segmented image. (©[AT04b], reproduced with permission.)

This method has been modified to include a dynamical model with motion priors [AT04c] and has been embedded in a tracking framework combining dynamics from the previous state estimate with a special regressor to disambiguate the pose. Tracking is then formulated either as a single fully regressive model or by using the regression estimates in a multiple hypothesis tracker based in CONDENSATION [AT06b]. In contrast to Rosales *et al*. [RASS01], this method demonstrate an ability to deal with ambiguities in a probabilistic manner. A similar method was contemporaneously proposed by Sminchisescu *et al*. [SKLM05]. For hand tracking, Thayananthan *et al*. [TNS$^+$06] extended the Tipping's original Relevance Vector Machine method [Tip01] for multidimensional target spaces and multiple hypotheses. Unlike Agarwal and Triggs [AT05] regressor, this includes the hyper-parameters in the optimisation process.

In [AT06a], Agarwal and Triggs used SIFT features [Low04] computed on a regular grid on the whole image. No segmentation is required, but the contribution of the background noise to the image descriptor is minimised by eliminating or downweighting background features using non-negative matrix factorisation [LS99], which is trained with features from clean foreground images. Pose estimation is then performed using the same unimodal regression method as in [AT04a], because the experiments only show estimation of the upper body pose, which is less ambiguous than the whole body. The experiments show that, for images with cluttered background, this method provides similar pose estimation performance to the method based on segmented silhouettes. The downside of this method is that it is not invariant to scale, rotation or translation, but it can be robust to some variation in clothing. In [AT06a], extensive experiments with synthetic images were performed, but only a few real image samples are shown. Both in the real and synthetic images shown, people wear tops with fairly uniform textures. The authors claim that better results can be achieved if larger training sets are employed.

2.5 A note on criteria for comparative evaluation of results

Despite a large body of work has been found in the literature, no standard methodology has been found to evaluate tracking and pose estimation results. For body tracking there are some human motion capture (HMC) data available publicly, (*e.g.* [Car]), but such data was used to generate synthetic images to which the tracker is applied, as the original HMC natural images are not available. An exception is the database described in [HKL06], which has human motion capture data with silhouettes and original images, but it is not publicly available. For hand tracking, there is no standard database or systematic evaluation method.

Ramanan and Forsyth [RF03] report tracking success whenever there is *any* overlap between a limb and the ground truth. This is probably very generous, but it is a good criteria for real-time applications on images with severe occlusions, fast movements and large acceleration. Most of the researchers have claimed that a qualitative visual agreement between the back projected models and the image is the most basic requirement of tracking performance. This is usually demonstrated with videos made available in the Internet.

The cost function that is used to minimise the state estimate of the trackers can provide a quantitative description of the tracking result. However, it does not provide a meaningful evaluation of the error of the pose estimate.

Track life is the length of time that the tracker remains on target. Track loss occurs if the measured cost grows arbitrarily large because the model is no longer projected on the correct parts of the image. Track life can be used to validate the result, though it is not a strong criteria, because problems caused by singularities and deficiencies of the model may not be made explicit [Reh95].

For whole body tracking, Sigal *et al*. [SBR$^+$04] were able to perform a quantitative evaluation using a professional marker-based motion capture system that was calibrated with the cameras used for tracking. This is the most accurate solution, but such professional systems are rarely available for research purposes mainly due to their cost.

Manually measured ground truth data has been employed by some authors (*e.g*. [BGP96]). Such measurements are usually obtained through mouse clicks in the position of the joints in the images and the use of a minimisation method to estimate the model posture from the measured positions. The obvious disadvantage of this method is that human operators are not reliable (or not available), specially for long sequences. The measurements can also be inaccurate because some joints may not be visible in all the images and determining the position of the joints is not always obvious.

Some researchers have used off-line processing using more computationally demanding parameters and multiple cameras to estimate the ground truth data. Such data is used to evaluate on-line real-time or monocular implementations (*e.g*. [FGTK02, TRMM01]). However, the reliability of such method is dubious if the same method is applied for on-line and off-line tracking.

A plausible alternative is to perform experiments in which the user is asked to touch known points in the world, as Bernardo *et al*. did in [BGP96]. But this does not accurately evaluate the estimation of all the joints of the hand.

Therefore, there is a demand for comparative evaluations of different methods for 3D hand tracking. While a standard framework or benchmark database is not defined, the evaluation method will be chosen according to the resources available and comparisons can be made between methods implemented within an institute, rather than globally.

This thesis shows comparisons of methods in Chapters 6 and 7. In the former, two generative methods are compared in terms of accuracy, efficiency and robustness. The comparison is based on their formalism, quantitative tracking results on synthetic data, and time measurements. In the latter, a single view is compared with a multiple views discriminative method. That comparison is based on quantitative tracking results on synthetic data, qualitative results on real images and time measurements.

2.6 Summary and concluding remarks

This chapter reviewed the main approaches to hand tracking including a range of references from methods based on extracting meaning directly from low level image features to higher level methods. The main interest was on methods that estimate the hand pose in 3D in real-time. A lower level method to locate areas of interest is studied in Chapter 3, where it is identified that there has been a consensus that the most reliable cue for hand tracking is obtained by skin colour detection using classifiers applied to a colour space with brightness normalisation.

Two main approaches for 3D hand pose estimation have been identified: generative model-based and discriminative estimation methods. The use of a training set of natural hand poses is essential for discriminative methods. For model-based methods, it has been shown that such data is of great importance to reduce the dimensionality of the state space, reduce ambiguities and increase accuracy and reliability.

Temporal information and motion priors have originally been used only for model-based methods, but recent discriminative methods have shown that the use of such information reduces the estimation cost and the ambiguities.

Although discriminative (or "tracking as detection") approaches are robust and have no latency limit, they do not make model-based methods obsolete [LF05]. As shown later in Chapters 4–6 of this thesis, model-based methods are view independent and less dependent on the training data. They can also provide a complete and continuous coverage to the parameters state. Model-based methods are easily scalable for multiple views and this has shown obvious improvements to the estimation results. But recently such approaches have been left aside for model-based methods. In discriminative methods they have only been explored modestly so far (*e.g.* [HSS02]). This thesis exploits multiple views for both generative and discriminative approaches. A novel multiple view discriminative method is described in Chapter 7 and comparisons with a single view implementation are shown.

Considering the literature, overall, some very good results have started to be shown with fast methods, but their robustness and accuracy have still not reached a point where a wide range of follow up applications could be successful. This shows that much improvement can still be achieved. Some contributions have been achieved in this thesis, which are described in the next chapters.

3

Software tools and apparatus

3.1 An architecture for object tracking

The methods for hand tracking to be described in Chapters 4, 5 and 6 of this thesis were developed within the context of the assistive workbench illustrated earlier in Figure 1.5 of the introductory chapter.

From that sketch, it can be surmised that the principal task modules are (i) rigid object tracking; (ii) hand tracking; (iii) head tracking; and (iv) hand tracking from the wearable robot. In addition there are the tasks of (v) data fusion and (vi) 3D visualisation. In this thesis we are concerned with tasks (i), (ii), (iv) and (vi). They have been implemented as separate processes in a client-server CORBA-like architecture, with communication via TCP sockets, as shown in Figure 3.1.

An object is tracked on the client side and visualised (and reasoned about) on the server side of this architecture. To avoid communicating graphics primitives, two instances of the object are maintained, one on either side. One way to do this would be to construct the instances simultaneously from a common configuration file which would specify the object type, communication port and initial configuration and pose. However, specifying the port in this way would require all objects to exist throughout. Instead, to enable the dynamic addition and removal of objects, the object is created first on the client side, and a predefined port is defined on the server side via which new objects can register their existence. When one does, the object is duplicated on the server, and a mechanism invoked which dynamically allocates a dedicated port for communication between the two instances.

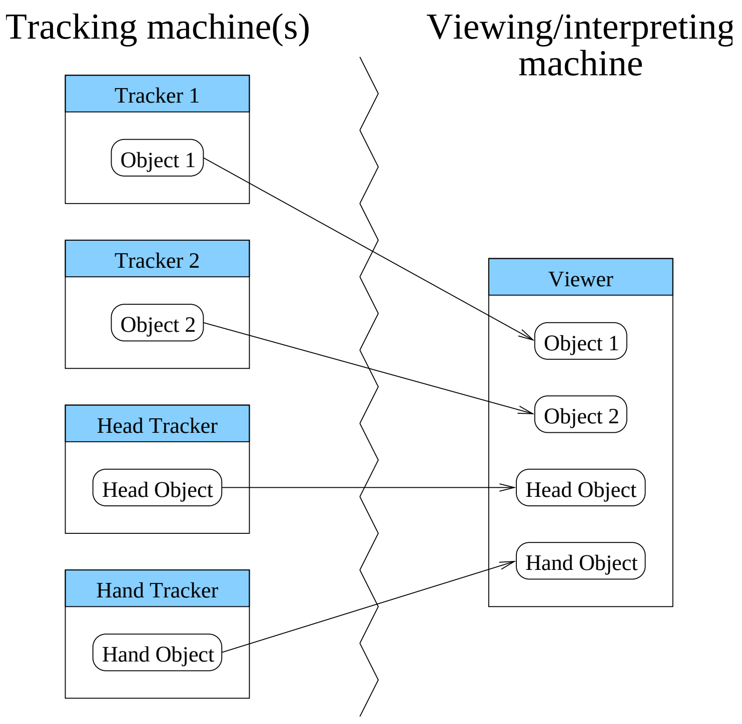
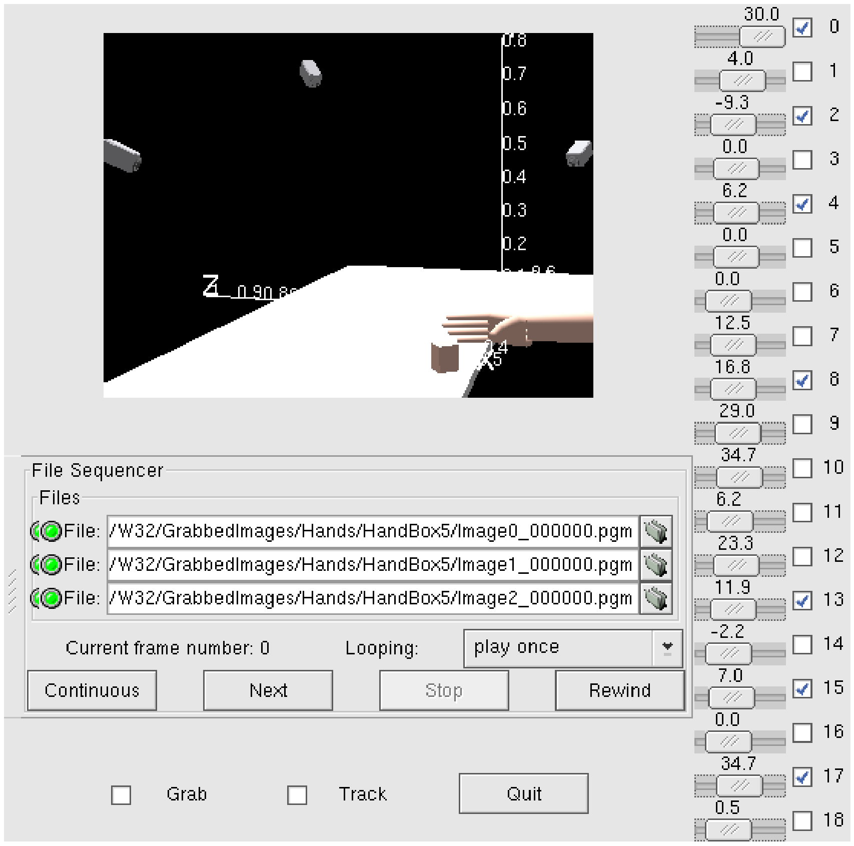


Figure 3.1: The communication of object position, orientation and other configuration requires the existence of a corresponding object in the viewer application. The tracking blocks are independent processes.

This system and all video-rate code has been implemented in C++, using the Active Vision Laboratory's Vision Workbench (VW) library. This library incorporates basic computer vision methods and some modules of numerical processing based on VXL [The03]. The graphical user interface is based on GTK-- and 3D visualisation methods uses OpenGL, which takes full advantage of any available accelerated hardware for 3D graphics. This frees the cpu from the heavy processing needed to render 3D objects. VW also defines standard interfaces to acquire images from cameras on-line, or from disk off-line.

All data files for information on 3D objects, camera configurations, colour classification data, and so on, are written in XML, where each information block lies between two readable and meaningful tags. This allows complex articulated objects to be modified by editing text files, without re-compilation. Figure 3.2 shows a sample graphical user interface created using these libraries showing the visualisation of cameras, hand, object and underlying desk.

The server and clients run under Linux. This is not a real-time operating system and cannot guarantee completion times. Even with careful algorithm design the occasional frame is dropped, particularly when



File: /W32/GrabbedImages/Hands/HandBox5/Image0_000000.pgm
File: /W32/GrabbedImages/Hands/HandBox5/Image1_000000.pgm
File: /W32/GrabbedImages/Hands/HandBox5/Image2_000000.pgm

Current frame number: 0 Looping: play once

30.0	✓	0
4.0	☐	1
-9.3	✓	2
0.0	☐	3
6.2	✓	4
0.0	☐	5
0.0	☐	6
12.5	☐	7
16.8	✓	8
29.0	☐	9
34.7	☐	10
6.2	☐	11
23.3	☐	12
11.9	✓	13
-2.2	☐	14
7.0	✓	15
0.0	☐	16
34.7	✓	17
0.5	☐	18

Figure 3.2: The GUI created to control and visualise articulated objects.

images are captured to disk while being processed. Care has been taken to account for the occasional variation in inter-frame duration. As suggested by Figure 3.2 up to three cameras are used in this work. In most of the experiments in this thesis, Sony VL500 digital cameras cameras have used with up to three sitting on the same Firewire (IEEE 1394) interface. Table 3.1 shows the frame rate obtained for various sizes of image and capture modes. If c cameras are connected to a single interface, all the cameras deliver one frame sequentially, so the maximum interval between the acquisition of an image from each camera is $1/(rc)$, where r is the frame-rate. Although the Sony VL500s can be externally synchronized, other cheaper cameras cannot. In practice the speed of movement is sufficiently small for the time skew to be tolerable.

Mis-calibration of the cameras, however, is much less tolerable.

Image Size	YCbC Mode	Max frame rate (Hz)	Max. no. cameras
320×240	4:2:2	30	3
640×480	4:2:2	15	3
640×480	4:1:1	30	2

Table 3.1: The maximum frame rate achievable for the given number of Firewire cameras, image size and capture mode.

3.2 Camera calibration

The methods of object pose recovery using explicit 3D models described in Chapters 5 and 6 require cameras whose internal and external parameters are known. To estimate these parameters, a method based on Section 2.5 of [Tor02] is employed: the radial distortion and calibration parameters are estimated iteratively. To ease initialisation, the user clicks on the hinge of the calibration grid (shown in Figure 3.3b). Corner features are located using Harris' corner detector [CH88] and the algorithm tries to fit lines to the un-distorted location of the corner features. The distortion is modelled using Harris' formulation [Har92b], in which the relation between the displacement of an ideal image point and its radial distance r_u from the centre of distortion is modelled as

$$r_d = r_u \left(\frac{1}{\sqrt{1 - 2\kappa_1 r_u^2}} \right) \tag{3.1}$$

This is the forward distortion equation, where $\kappa_1 < 0$ models barrelling distortion and $\kappa_1 > 0$ models pin-cushion distortion. The backward equation (below) corrects measured distorted image points back to their ideal position:

$$r_u = r_d \left(\frac{1}{\sqrt{1 - 2\kappa_1 r_d^2}} \right) \tag{3.2}$$

The matching error between the undistorted image and the ideal image is minimised with Levemberg-Marquardt to estimate κ, which is initialised with zero. An estimate of the calibration is then performed and the process is iterated until convergence.

The camera calibration step is based on the method described by Faugeras [Fau93] (at least for the intrinsic parameters as will become clear below). Given a 3D point $\mathbf{X}$ represented in homogenous coordinates in the world coordinate frame, its projection $\mathbf{x}$ onto the image plane of a camera (with radial distortion corrected for) is taken as

$$\lambda \mathbf{x} = \mathtt{P}\mathbf{X}, \tag{3.3}$$

where $\mathtt{P}$ is the 3×4 projection matrix and λ is a scale factor. If $\mathbf{X}$ is in a Euclidean frame, $\mathtt{P}$ can be decomposed as two meaningful geometric entities: the internal and external calibration parameters

$$\mathtt{P} = \mathtt{K}(\mathtt{R}\,\mathbf{t}) \tag{3.4}$$

where the external parameters are the rotation and translation that transform points defined in the world coordinate frame into those defined in the camera frame, and where the internal calibration describes the transformation between an ideal image and the pixel image

$$\mathtt{K} = \begin{pmatrix} f & \bar{s} & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix} . \tag{3.5}$$

Here f is the focal length, α is the aspect ratio, (p_x, p_y) defines the principal point, and $\bar{s} = -f_x s$ describes the often neglible image skew. As there are 6 rotational and translational DOF and 5 internal calibration parameters, a minimum of 6 correspondences $\{\mathbf{X}_i \leftrightarrow \mathbf{x}_i\}$ between known scene points and measured image point correspondences are required to recover $\mathtt{P}$. However, a useful rule of thumb [HZ01] is that for a good estimation the number of constraints should exceed the number of unknowns by a factor of five, suggesting that around 30 correspondences is practical minimum. The set of world points is defined by the corners of the squares on the ubiquitous 3D calibration grid (Figure 3.3), and the image positions determined to around ±0.1 pixel by fitting extended straight lines to the edgels computed along the edges of the squares, and then intersecting the lines.

Initial values for the elements of $\mathtt{P}$ are found by a direct linear transformation. In practice it is safe enough here to set the scale by fixing $p_{34} = 1$ and recovering the other 11 elements, but more generally one should guard against $p_{34} \approx 0$ by recovering all 12 using a null space method. These initial values are then refined by non-linear minimization

$$\mathtt{P} = \arg\min_{\mathtt{P}'} \sum_i d(\mathbf{x}_i, \mathtt{P}'\mathbf{X}_i)^2 \tag{3.6}$$

where $d(\mathbf{x}_i, \mathtt{P}'\mathbf{X}_i)$ is the Euclidean distance observation and estimation. Here the Nelder-Mead simplex method has been used [NM65] but others have used Levenberg-Marquardt [Lev44, Mar63] with equal

(a)

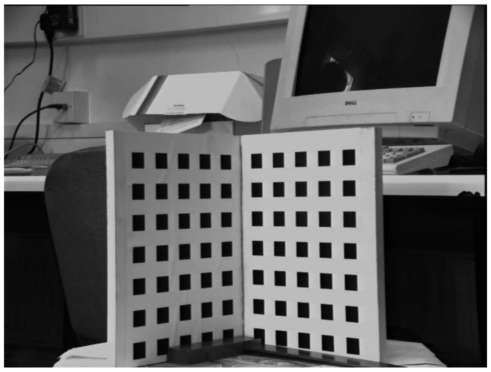

(b)

Figure 3.3: (a) The Sony VL500 camera, three of which are used in this work. (b) The calibration grid in which the corner of the squares are used to compute $\{\mathbf{x}_i \leftrightarrow \mathbf{X}_i\}$ correspondences. For calibration, best results are obtained if the grid occupies the whole image.

success. With $\mathtt{P}$ determined, the rotation and internal matrix is found by applying QR-decomposition to the inverse of the leftmost 3×3 block of $\mathtt{P}$

$$\lambda' \mathtt{R}^{-1} \mathtt{K}^{-1} = \boldsymbol{\mathcal{Q}} \boldsymbol{\mathcal{R}} \leftarrow \mathtt{P}_L{}^{-1}$$

so that $\mathtt{R} = \boldsymbol{\mathcal{Q}}^{-1}$ and $\mathtt{K} = \lambda' \boldsymbol{\mathcal{R}}^{-1}$, where the scale is fixed so that $k_{33} = 1$. (There are also other sign ambiguities that between rows and columns of $\mathtt{K}$ and $\mathtt{R}$ that are resolved by requiring the focal length, aspect ratio and principal point coordinates to be positive.) The translation is determined by: $\mathbf{t} = \boldsymbol{\mathcal{R}}(p_{14}, p_{24}, 1)^\top$.

To generalise the calibration data for any image resolution used, and to benefit from statistical centring, a normalised image is employed. For an image with width w and height h, the conversion of parameters is done as follows:

$$\hat{f} = \frac{f}{w} \qquad \hat{p_x} = \frac{p_x}{w} - 1 \qquad \hat{p_y} = \frac{1}{w}(p_y - h) \; . \tag{3.7}$$

3.2.1 Interpolation over zoom

The Sony cameras have controllable zoom lenses and it is convenient to be able to adjust these without performing a full re-calibration.

The process described above was repeated for values of zoom motor setting 40, 100, 200, 300, . . ., 1300, 1400 from the accessible range of 40, 41, . . ., 1432. For each calibration position, ten images of the grid were acquired (under small variations in lighting) for each of these zoom positions and the internal

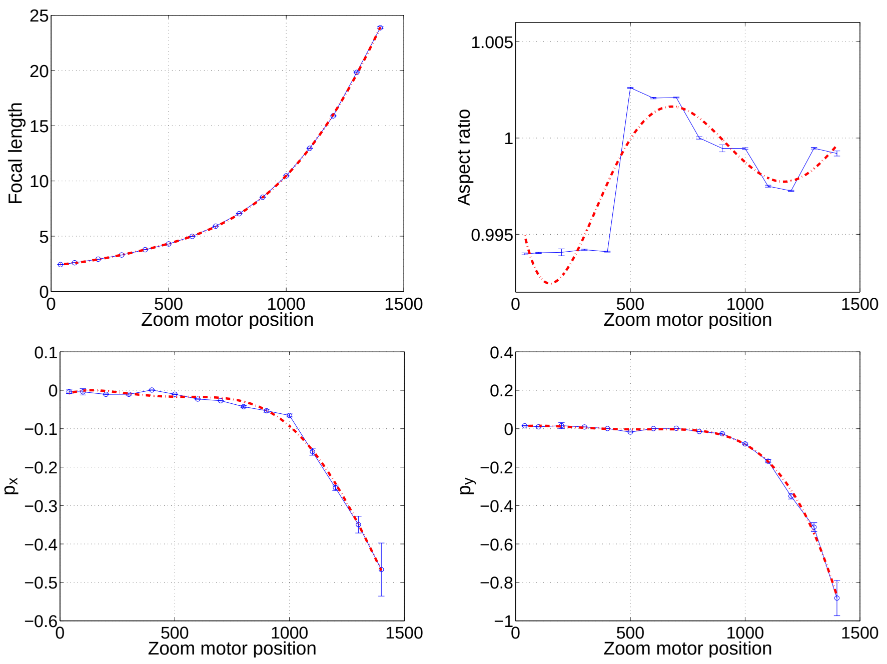


Figure 3.4: Interpolation of calibration parameters obtained from 10 images at each zoom motor position. The mean parameter value is shown by a circle and the standard deviation by errorbars. The dotted lines show interpolation using 5$^{\text{th}}$ order Chebyshev polynomials.

parameters recalculated. Since calibration estimates were performed only for one out of fifty odometry positions, it was found that a more reliable interpolation would be obtained if a polynomial fit was used across the range, rather than local linear interpolations. Chebyshev polynomials were chosen due to their stability, the fitting method described in section C.2 of [Tor02] was employed. Through experimental evaluation, it was found that the use of fifth order polynomials gave good interpolations.

The plots in Figure 3.4 show the estimated focal length, aspect ratio and principal points for one of the cameras throughout the zoom range. The parameters are shown for normalised images. Note that the principal points are close to the image centre up to odometry position 900. The aspect ratio, which should be constant across the range of odometry positions, presented some small variation explained by the “mopping up” of errors elsewhere in the system.

The error bars are relatively small for most estimages because the image data was acquired with small intervals and small pose changes for each zoom position. But the obtained polynomials provided good genaralisations across the zoom range, considering that calibration estimates were available from a very limited set of zoom positions.

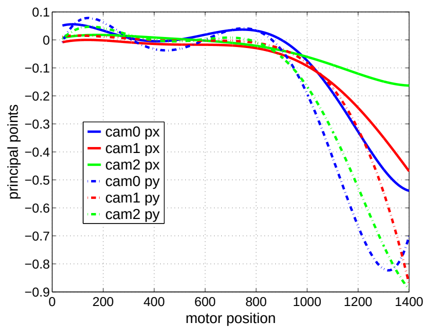


Figure 3.5: Variation of the principal point coordinate (for normalised images) with zoom for 3 cameras of the same model. Continuous curves show p_x and dashed curves show p_y.

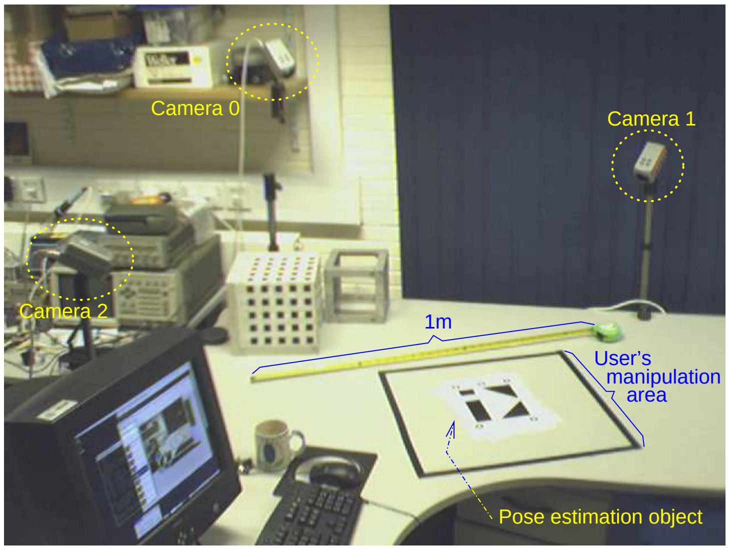


Figure 3.6: Desktop environment with the tree cameras and the planar pose estimation object.

For other two cameras of the same model the focal length and aspect ratio showed all but identical behaviour, but there was greater variation in the principal point as shown in Figure 3.5. Again this is expected depends on the alignment of all the lens elements and image plane and would be harder to control in manufacture than the distance of the lens to the image plane.

3.2.2 Re-working the external calibration

In normal use, the internal calibrations of the Sony cameras, including the variation over zoom, have been found to remain valid over long periods of time. However, the external calibration is much more susceptible to change, by accidental bumping of furniture and so on. To avoid having to perform a complete recalibration (and, hence, completely wiping out the benefits of interpolation) a more convenient planar calibration of the external parameters has been used. It has the advantage too of forcing the world's Z-axis to be perpendicular to the desk.

Figure 3.6 shows the planar calibration object in use. The planar object defines the world $Z = 0$ plane, and its oriented pattern defines the X- and Y-axes. The projection equation (Eq. 3.3) is simplified to that of a plane to plane homography

$$\lambda \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \mathtt{P}_{3\times 3} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix} \tag{3.8}$$

where $\mathtt{P}$ is the homography whose elements $\mathbf{p}$ can again be estimated linearly, up to scale, as the null space of $\mathtt{A}$ in $\mathtt{A}\mathbf{p} = \mathbf{0}$ where

$$\mathtt{A} = \begin{pmatrix} & & & & \vdots & & & & \\ X_i & Y_i & 1 & 0 & 0 & 0 & -X_i x_i & -Y_i x_i & x_i \\ 0 & 0 & 0 & X_i & Y_i & 1 & -X_i y_i & -Y_i y_i & y_i \\ & & & & \vdots & & & & \end{pmatrix} \tag{3.9}$$

for $n \geq 4$ points. Again one can refine the initial estimate by non-linear optimization. As the internal calibration is known, one can recover

$$\mathtt{H} = (\mathbf{h}_1 \;\; \mathbf{h}_2 \;\; \mathbf{h}_3) = \mathtt{K}^{-1}\mathtt{P} \, , \tag{3.10}$$

where $\mathbf{h}_3$ is the translation and where $\mathbf{h}_1$ and $\mathbf{h}_2$ are the first two columns on the rotation matrix, all modulo a scale factor. Rotation matrices are orthogonal and have unit norm, and the actual translation and rotation matrix rows are first estimated as

$$(\mathbf{r}_1 \; \mathbf{r}_2 \; \mathbf{t}) = \frac{2}{||\mathbf{h}_1|| + ||\mathbf{h}_2||} (\mathbf{h}_1 \; \mathbf{h}_2 \; \mathbf{h}_3) \; . \tag{3.11}$$

The third column of the rotation is first determined as $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$. However, due to image noise and discretization problems, these columns will not be mutually orthogonal and of unit norm. This is corrected by means of the singular value decomposition (SVD) [Str88]

$$\begin{aligned} \mathtt{U}\,\mathtt{W}\,\mathtt{V}^\top &\leftarrow \tilde{\mathtt{R}} = (\mathbf{r}_1 \; \mathbf{r}_2 \; \mathbf{r}_3) \\ \mathtt{R} &= \mathtt{U}\,\mathtt{V}^\top \, . \end{aligned} \tag{3.12}$$

(Here $\mathtt{W}$ is the diagonal matrix of singular values of $\tilde{\mathtt{R}}$, and $\mathtt{U}$ and $\mathtt{V}$ are both orthonormal matrices. The columns of $\mathtt{U}$ form a basis in $\mathbb{R}^3$ for the range of $\tilde{\mathtt{R}}$ and the columns of $\mathtt{V}$ form the basis of the nullspace of $\tilde{\mathtt{R}}$. If all the singular values are set to 1, the reconstructed matrix $\mathtt{R}$ is the orthonormal rotation matrix closest to $\tilde{\mathtt{R}}$ in the Frobenius distance sense [PTVF88].)

The pose estimation methods described both here and earlier estimate the rotation and translation of the observed object in the camera coordinate frame. Later it will be more convenient to describe the camera position in the world frame. This is simply the inverse Euclidean transformation

$$\mathtt{R}_C^W = \mathtt{R}^{-1} \qquad \mathbf{t}_{CW} = \mathtt{R}_C^W(-\mathbf{t}) \ . \tag{3.13}$$

3.2.3 Detection and localisation of the calibration object

Some care has been taken to automate the detection and localization of the calibration object (Figure 3.7(a)). The initial detection is based on ring templates (Figure 3.7(b)), which are rotationally invariant but not easily confused with random darkspots in the image. To improve the robustness, the template is generated with a number of registered sample images obtained with slightly different scales and perspective.

The template is correlated with the image (Figure 3.7(b)) the maxima detected, and straightforward geometric reasoning determines the appropriate correspondence in the scene. The least squares estimation method of Section 3.2.2 (referred to as the linear method hereinbelow) gives the first estimate of the transformation which can be refined non-linearly using, e.g., Nelder-Mead simplex method over the sum of squared distances between the projected and measured disc centres (Equation 3.6). Examples results are shown in Figure 3.8.

It is of course necessary to improve the calibration by using more image evidence. This could be done just as earlier by using a grid to generate points and line and capturing one image, but here a different approach is taken. The estimate of the transformation is handed over to a tracker which refines the rotation and translation to best fit the projection of the known triangular and rectangular objects to the observed edges in the image, as shown in Figure 3.9. The RAPiD tracker is first described in its proper context in Chapter 4. However, in addition to its using more image data, the tracker makes repeated estimates allowing an assessment of the error that arises because of unmodelled variations in lighting,

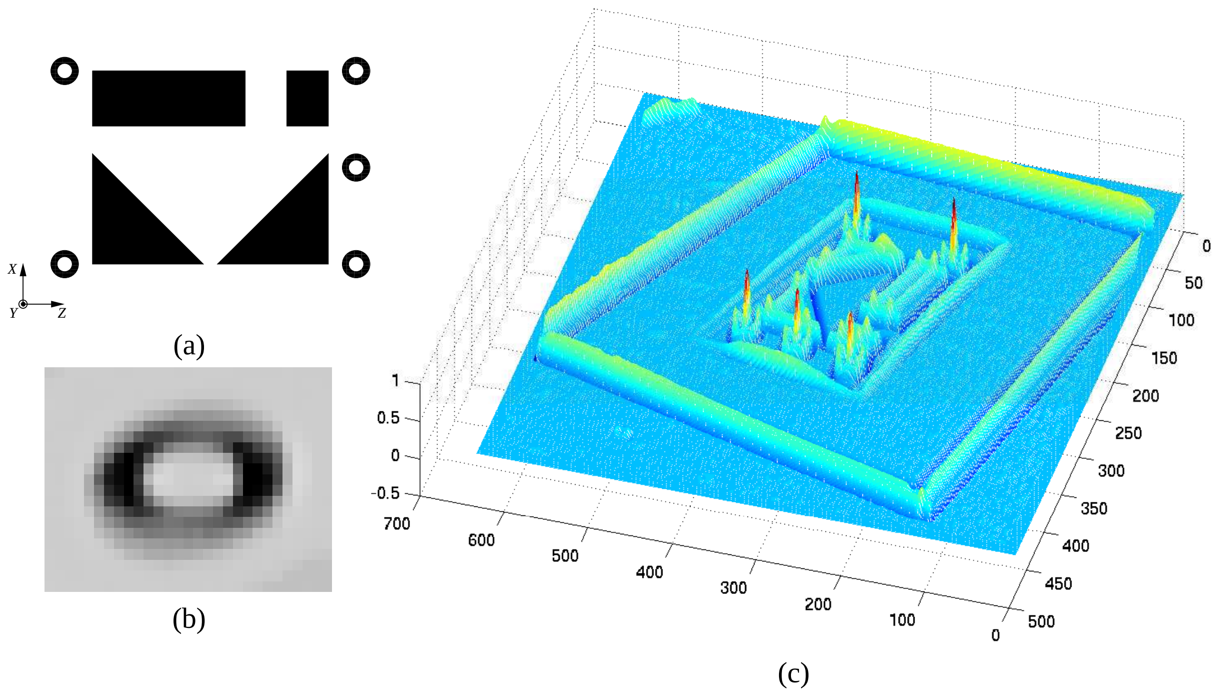


Figure 3.7: (a) Calibration pattern used for camera pose estimation, showing the origin and axes of the world coordinate. (b) Template for ring detection obtained from 15 images of rings; (c) Result of template match for a view of the scene shown in Figure 3.8.

image noise, and vibration, and because of the stochastic nature of robust pose and robust collinearity methods (described in Section 4.2.4).

3.2.4 Calibration results

The accuracy of the camera external parameters was assessed by determining the error in the angles from the world origin to the camera and the error in the distance.

The arrangement of cameras was as in Figure 3.6: they were widely separated, and zoomed out sufficiently far that the entirety of the 0.5m sided manipulation cube was visible, volume consistent with that based on biomechanical analysis and used by Mayol *et al.* [MTM02]. Figure 3.10 shows a view of the scene rendered using the information return from the calibrations of the three cameras.

To assess the likely error in the individual camera external calibrations, and hence the likely error in the position of an object recovered in three cameras, 21 images per camera were acquired at different times (without moving the cameras or target object) and the pose estimation algorithm was applied 1000 times per image. On average, the RAPiD tracker used 700 control points. Robust pose and robust

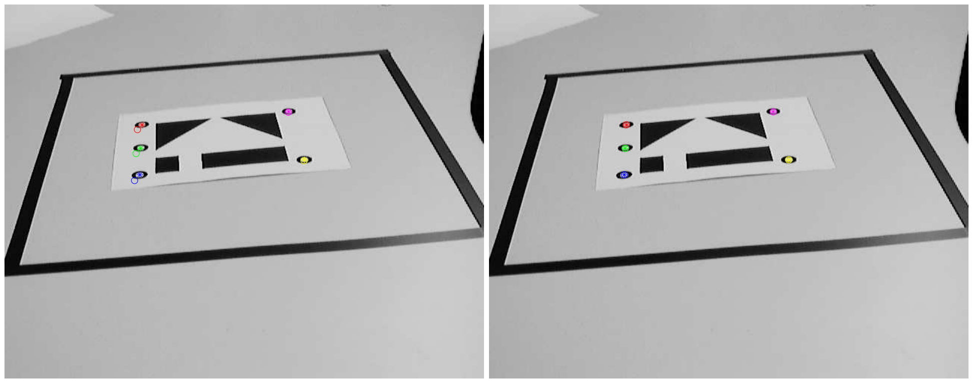


Figure 3.8: Example View from camera 2 showing the detected feature points as asterisks and the projection of the corresponding scene points after linear and non-linear estimation.

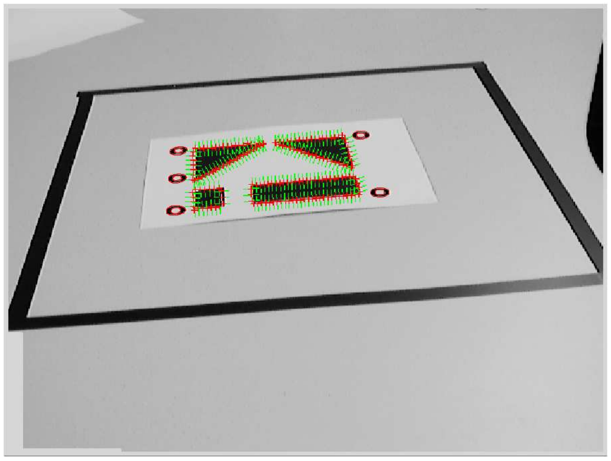


Figure 3.9: Refinement of the pose estimations obtained in Figure 3.8 using the RAPiD 3D rigid object tracker with edge features.

collinearity (see Section 4.2.4) were applied using 20 and 10 iterations, respectively. The whole process was iterated 10 times per frame to ensure that starting transients had decayed.

Following a procedure suggested by Thompson *et al.* [TRMM01], the mean translation $\bar{\mathbf{t}}_{WC}$ from the world origin to the camera was derived as was the mean unit direction $\bar{\mathbf{n}}$ of the camera axis in world coordinate frame. The object pose in the camera coordinate frame, is expect to be estimated with less accuracy in translation along the camera axis and rotations in depth. These translate to camera pose estimates in the world coordinate frame (centred on the calibration object) as translation errors that are highly correlated in the three dimenions. Thence, the inaccuracy in translation was computed using a measure-

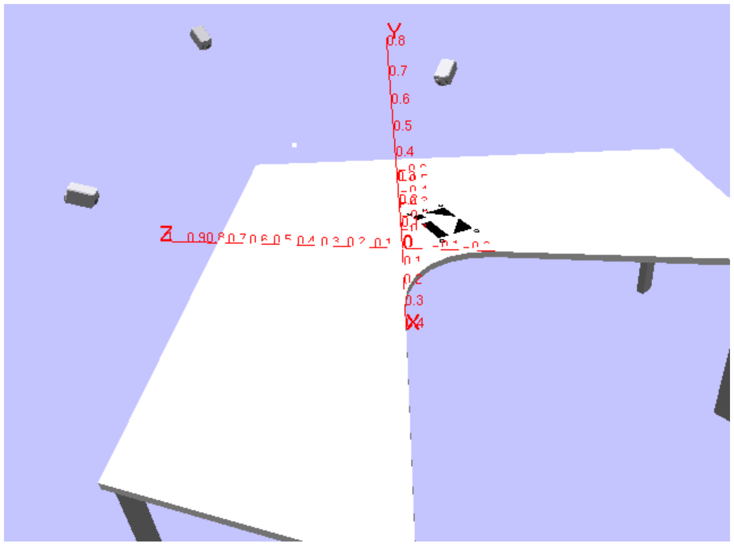


Figure 3.10: A view of a 3D model of the cameras, desk and calibration plane, rendered using the rotations and translations returned from the calibration.

ment that is suitable for multivariate cases. Assuming that the distribution of the pose estimations can be modelled by a Gaussian, one can determine the principal axes by obtaining the eigenvalues and eigenvectors of the covariance matrix of the pose estimations $\Sigma[\mathbf{t}] = \sum_j [\mathbf{t}_{WC\,j} - \bar{\mathbf{t}}_{WC}][\mathbf{t}_{WC\,j} - \bar{\mathbf{t}}_{WC}]^\top$. Each eigenvalue λ corresponds to the variance in the direction of its eigenvector. An inaccuracy measurement can be defined by the volume of the ellipsoid defined by these eigenvectors and eigenvalues through this formula (see Figure 3.11a):

$$V = \frac{4}{3}\pi\lambda_1\lambda_2\lambda_3 \tag{3.14}$$

A more meaningful measurement is obtained if the expected standard deviation σ can be used (recall that $\sigma = \sqrt{\lambda}$). This can be thought of as an expected value of the distance between the pose estimation and the mean of the estimated pose. This measurement can be defined by

$$\Delta\mathbf{t} = \sqrt[3]{\sigma_1\sigma_2\sigma_3} \tag{3.15}$$

Note that the equality $\sigma_1\sigma_2\sigma_3 = \sqrt{det(\Sigma[\mathbf{t}])}$ simplifies the implementation of the last equation.

The error in rotation was expressed by two angles, $\Delta\alpha$ which is the standard deviation in the angle between the individual direction vectors and the mean $\bar{\mathbf{n}}$, and $\Delta\theta$ the standard deviation in the cyclotorsion about $\bar{\mathbf{n}}$, as shown in Figure 3.11. Table 3.2 shows the inaccuracy of the pose estimation, computed with the 21000 trials for each camera.

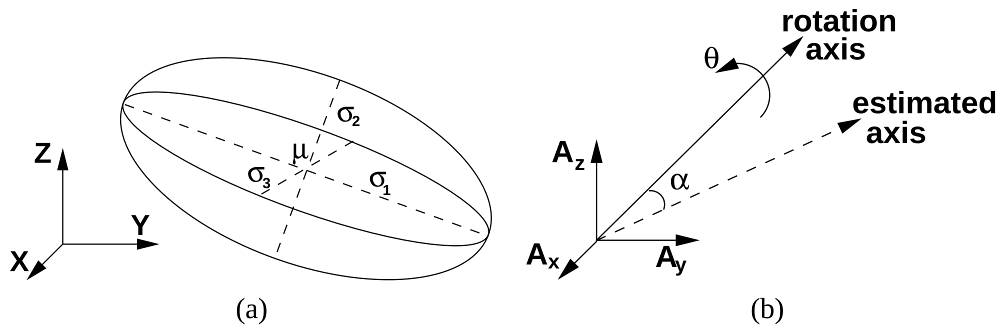


Figure 3.11: (a) Ellipsoid defined by the principal standard deviations of the position estimation in the translation space. (b) Angles used to evaluate the deviation in rotation axis α and in rotation angle θ.

Camera	$\Delta\mathbf{t}$mm	$\Delta\theta^\circ$	$\Delta\alpha^\circ$
0	8.1	0.48	0.12
1	4.6	0.46	0.15
2	9.3	0.85	0.29

Table 3.2: Pose estimation inaccuracy for the position of the cameras $\Delta\mathbf{t}$, the orientation of their optical axis $\Delta\theta$ the rotation about this axis $\Delta\alpha$. Position and orientation are expressed in the world coordinate frame.

Now it should be noted that these errors are from *individual* camera's calibration. If it is assumed that the camera position is correct, the error can be tranferred back to a point in the scene. But because the cameras are close to orthogonal, the error ellipsoids for a point viewed close to the centre of each camera will intersect orthogonally, and the resulting error covariance is not ellipsoidal, but can be approximated by a sphere of radius

$$r \sim \Delta\alpha \left(\frac{\pi}{180}\right) D$$

where D is the typical distance from camera to scene. Here $D \sim 1000$ mm, giving a translational error in an observed object of r between 2 mm and 4 mm. One expects this error to scale proportionally with depth, but inversely with focal length because a fixed error in the image corresponds to a smaller angular error. For this reason, where zoom lens is available, the pose estimate is done by zooming into the calibration object. Using the interpolation method described earlier, once the pose estimate is done, the cameras can zoom out to increase the field of view for the tracking experiments.

3.3 On the detection of hands images: a skin colour classifier

The third competence developed to support the research in the remaining chapters is that of hand detection. Any markerless visual method designed to detect and track an object without intervention must confront the question of how in the first instance to associate features observed in the image (be they pixels, edges, corners, etc.) with the object itself. Locating hands is likely to be difficult compared with, for example, face detection because hands are articulated objects that present both high variation in their shape and in their degree of self-occlusion. However, there is a useful uniformity in human skin colour allowing the development of a localization method based on pixel colour classification. As the review has indicated, and Chapters 4 and 7 will show, the resulting silhouette is sometimes all that is needed for 3D pose estimation. If internal edges are to be used, as they are in Chapters 5 and 6, the silhouette is still valuable in restricting search for an initial pose.

3.3.1 Eliminating brightness from the colour space

Classification based on colour requires pixels imaged from skin to form a tight cluster in some colour space. Although we loosely describe skins as being of different colour, the spectral variability is dependent mainly on the amount, density and distribution of melanin pigment in the skin, not on its colour [Mar02]. Thus, to a large extent it is the brightness of the skin that varies, not its colour [YLW98a]. Brightness normalisation involves reducing the dimensionality of a colour space (typically from three dimensions to two) by projecting points into a plane of constant brightness in the space. It is inevitable that un-modelled variations result in some overlap in the 2D space between skin and non-skin clusters, but the drop in dimensionality substantially cuts the volume of data and time required for training. Moreover, if the colour space decouples brightness and colour information from the outset, the task of brightness normalisation can be achieved by neglect rather than computation.

3.3.2 The choice of a colour space

Researchers into colour science, an important area long before the digital era, have proposed a large number of colour spaces each tailored to a different task. Among colour spaces that are decoupled, the most common — and commonly used for skin detection — are the CIE Chromatic space (used in, for example, [YLW98b]), the HSV (hue, saturation and value) space (e.g. [RMG98], [AP96], [ZYW00]), and the YUV/YCbCr space (e.g. [Coh, Fri99, YLW98a, FdC00]). Several comparisons of spaces for skin detection have been carried out, but Martinkauppi's thesis [Mar02] suggests that there is no definitive conclusion as to which is the best, in part because different databases with different illumination conditions have been used, but principally because the differences in output quality are marginal. Explanatory and exploratory notes about these spaces are given in Appendix A.

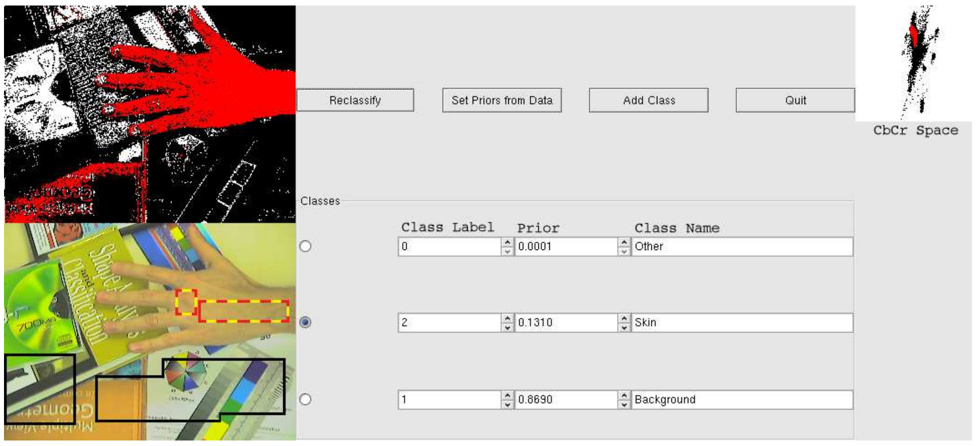
Since the goal here is to implement a video-rate method, processing time becomes the key criterion by which to assess methods. Now the decision becomes straightforward. The conversion to HSV or HSL requires a non-linear transformation algorithm, making this the least efficient in terms of computational cost. Conversion to CIE is linear and fast enough. However, it turns out that many digital colour cameras, like the Sony VL500 used here, deliver images already encoded by hardware in the YCbCr space. As explained in Appendix A, the Y channel holds the luminance information, which is to be neglected, and the Cb and Cr channels hold the chrominance information which is to be used for classification.

3.3.3 Classifying pixels for skin detection

Several methods have been applied to the problem of classification of skin pixels. [DHS00]. The simplest "manually" carve out a portion of colour space to be classified as skin [CnN98], defining it by thresholds or a lookup table. More common is to allow different colours to have a probability of arising from skin, and to learn the underlying PDF. Within this approach there are variations in how the distribution of skin samples is modelled.

Yang and Waibel [YW96] (and see [FdC00]) argue that a Gaussian PDF is good enough for their small dataset of skin colour samples. However, is not able to account for subtle variations in large databases. Nor do Yang and Waibel include training data to model the background colour distribution. This is assumed to be uniform, and a simple threshold in the PDF of the skin class defines the decision boundary. Multi-modal Gaussian mixtures were proposed by Jebara [JP97]: indeed any two-class classifier could be applied to this problem, but doing so would miss the point that the feature space has only three or fewer dimensions and the classes do not need to be modelled analytically.

Jones and Rehg [JR98, JR02] have shown that non-parametric histogram models provide higher accuracy and lower computational cost than using multi-modal Gaussian mixtures. However, their clas-



Class Label	Prior	Class Name
0	0.0001	Other
2	0.1310	Skin
1	0.8690	Background

Figure 3.12: Screen shot of the application to train the classifier for skin detection: the classification result is shown in the top left (skin is indicated by red, background by black and unknown by white). The panel on the bottom left shows the training areas already selected by the user for skin (dashed) and background (solid).

sification was done in RGB colour space, which is less robust to illumination changes. This problem is largely eliminated in the truncated (Y)CbCr colour space, and it is this approach which is developed here.

Histogram-based classification in the CbCr colour space

Skin colour detection is modelled as a maximum *a posteriori* classification problem, using histograms to model discrete PDFs [FP03]. During training histograms are built in the colour space for each class involved — here there are just two, skin $\mathcal{S}$ and background $\mathcal{B}$. If a pixel with colour $(C_{\rm b}, C_{\rm r})$ is known to be in the class $\mathcal{S}$ the bin count $c_{\mathcal{S}}[uv]$ is incremented, where

$$u = \text{floor}\,(C_{\rm b}/b) \qquad v = \text{floor}\,(C_{\rm r}/b)\ \ , \tag{3.16}$$

and b is the bin size. The resulting bin counts are normalized so that

$$P(uv|\mathcal{S}) = c_{\mathcal{S}}[uv]/T_{\mathcal{S}} \tag{3.17}$$

where $T_{\mathcal{S}} = \sum_{uv} c_{\mathcal{S}}[uv]$ is the total number of pixels labelled as skin during training.

During classification the posterior is determined using Bayes' rule

$$P(\mathcal{S}|uv) = \frac{P(uv|\mathcal{S})P(\mathcal{S})}{P(uv|\mathcal{S})P(\mathcal{S}) + P(uv|\mathcal{B})P(\mathcal{B})} \tag{3.18}$$

where the prior probability is $P(\mathcal{S}) = T_{\mathcal{S}}/T$ and T is the total number of pixels used in training. The likelihoods and priors involving the background are defined similarly (and as this is a two-class problem can be derived without storing a second histogram). Then a particular pixel with its colour (u, v) is labelled as skin if

$$P(\mathcal{S}|uv) > P(\mathcal{B}|uv), \tag{3.19}$$

which can for this two-class problem be simplified to $c_{\mathcal{S}}[uv] > c_{\mathcal{B}}[uv]$. (When there is neither skin nor background training sample for a given uv bin, an uncertainty arises, as $P(\mathcal{S}|uv) = P(\mathcal{B}|uv) = 0$. For the skin detection application, what matters in this case is that it is known that this uv bin does not represent a skin value, so it is classified as background.)

Iterative Training Method

The training process consists of selecting skin and background regions of images. This task is performed manually and can be very tedious for a large training set. But often only a few images are enough to create a good model for classification. To inform the user, a train-and-classify system was implemented. Before the user starts segmenting a new image, the system shows the classification result for this image using the current training set. The user then judges whether it is necessary to use this image for training or not depending on the size of miss-classified areas in the image. For each image, the software shows a track of the areas already selected by the user and it does not add samples from areas selected before. This idea is similar to the iterative training method proposed by Saxe and Fouls [SF96]. Figure 3.12 shows a screen shot of this software.

3.3.4 Qualitative evaluation

Tuning the generalisation power via the bin size

The training method described in Section 3.3.3 was used to populate the CbCr space with more than 500 thousand skin samples and more than 1.2 million background samples obtained from the image database described in Section A.5. Both Cb and Cr ranges were 0-255, and the bin size was $b = 1$. Figure 3.13(a) shows the histogram of the skin samples in the CbCr colour space, and Figure 3.13(b) shows the area of this colour space populated by skin samples some 1273 CbCr locations. The same is done for the

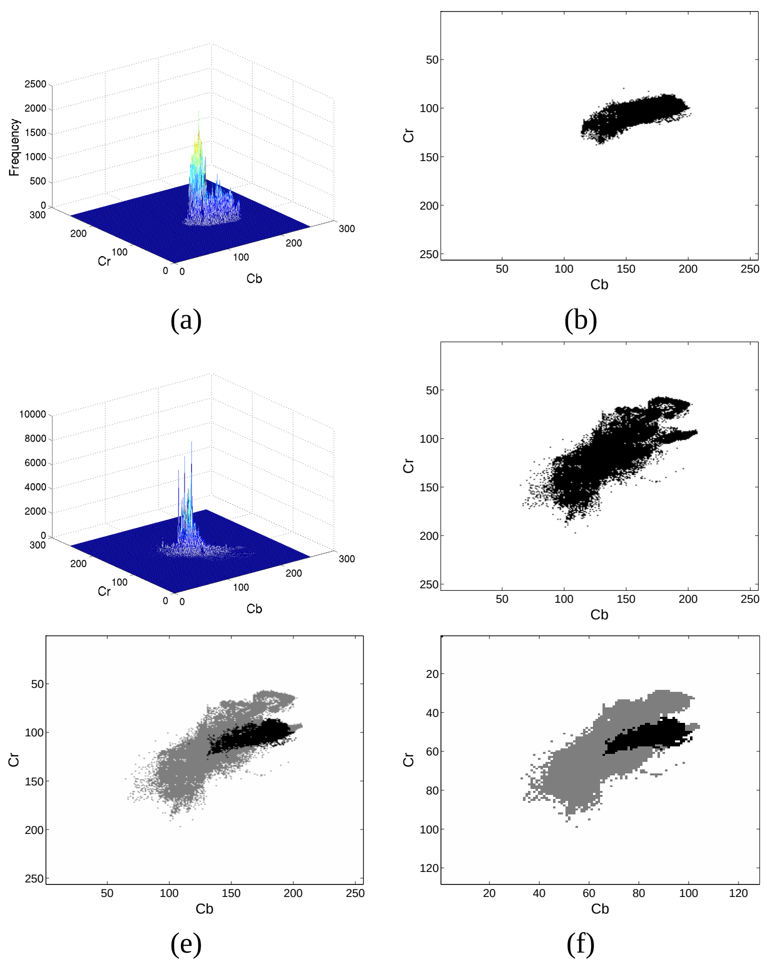


Figure 3.13: (a) Skin histogram for unitary bin size and (b) the “overhead” view of the occupied region of CbCr space. (c) and (d) are same for the background. (e,f) Derived lookup tables using a bin size of 1 and 2 respectively

background in figure parts (c) and (d), where 5144 CbCr locations are populated. The skin “area” is quite large because samples were acquired under different illumination conditions.

Lookup tables were built from these histograms. The first, in Figure 3.13(e), retained the bin size of 1, but it was found to leave gaps in the skin region. Figure 3.13(f) shows the result of increasing the bin size to $b = 2$. Many of the unknown CbCr classification values are extinguished. The effect of increasing the bin size on an arbitrary image is shown in Figure 3.14.

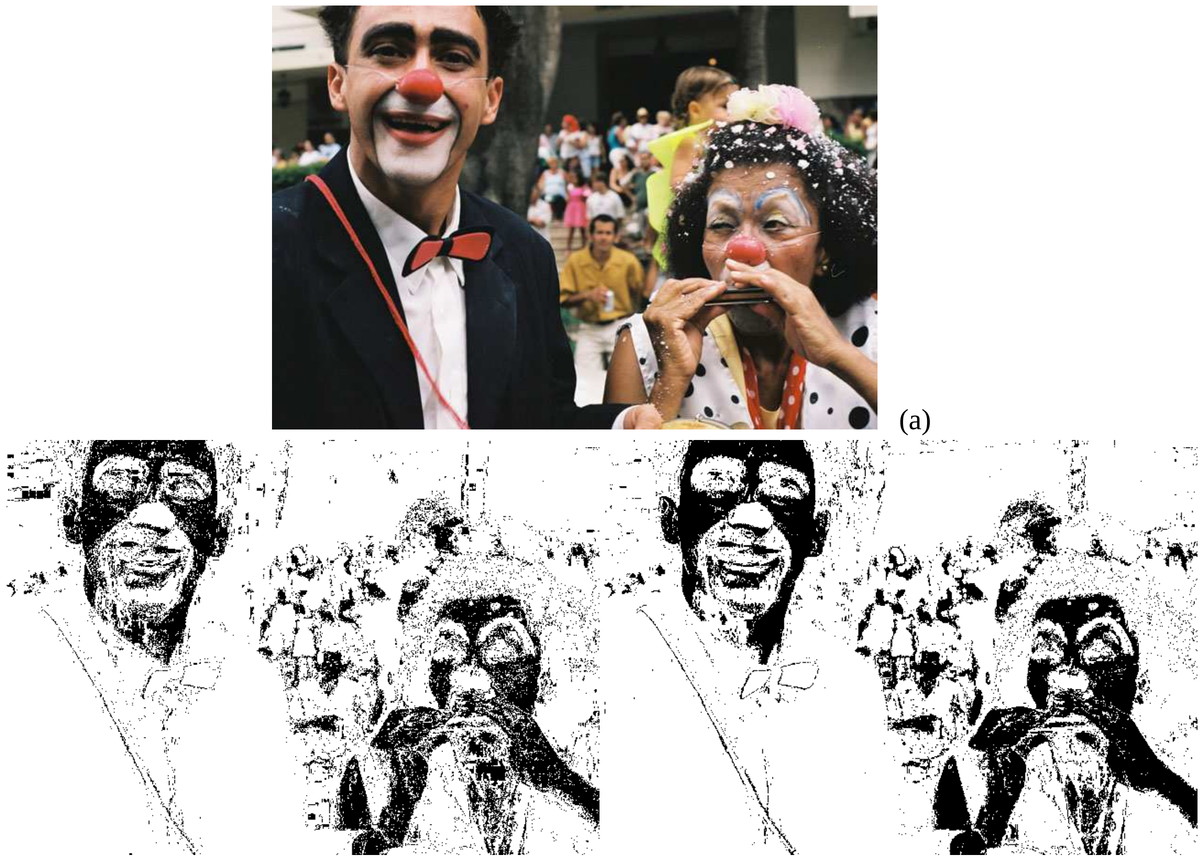


Figure 3.14: (a) An arbitrary image, and the effect on increasing the bin size from $b = 1$ in (b) to $b = 2$ in (c) for skin detection. (The original image (a) is ©2006 http://www.palhacomatraca.com.br, reproduced with permission.)

Results in uncontrolled conditions

Further results are shown in Figure 3.15. Some of these images were obtained with the camera set to adjust the brightness and contrast automatically. Such automatic modes also normalise image colours to “improve” the appearance to the human eye under variations in illumination. Others have cluttered background and include wooden objects whose colour is often close to that of skin.

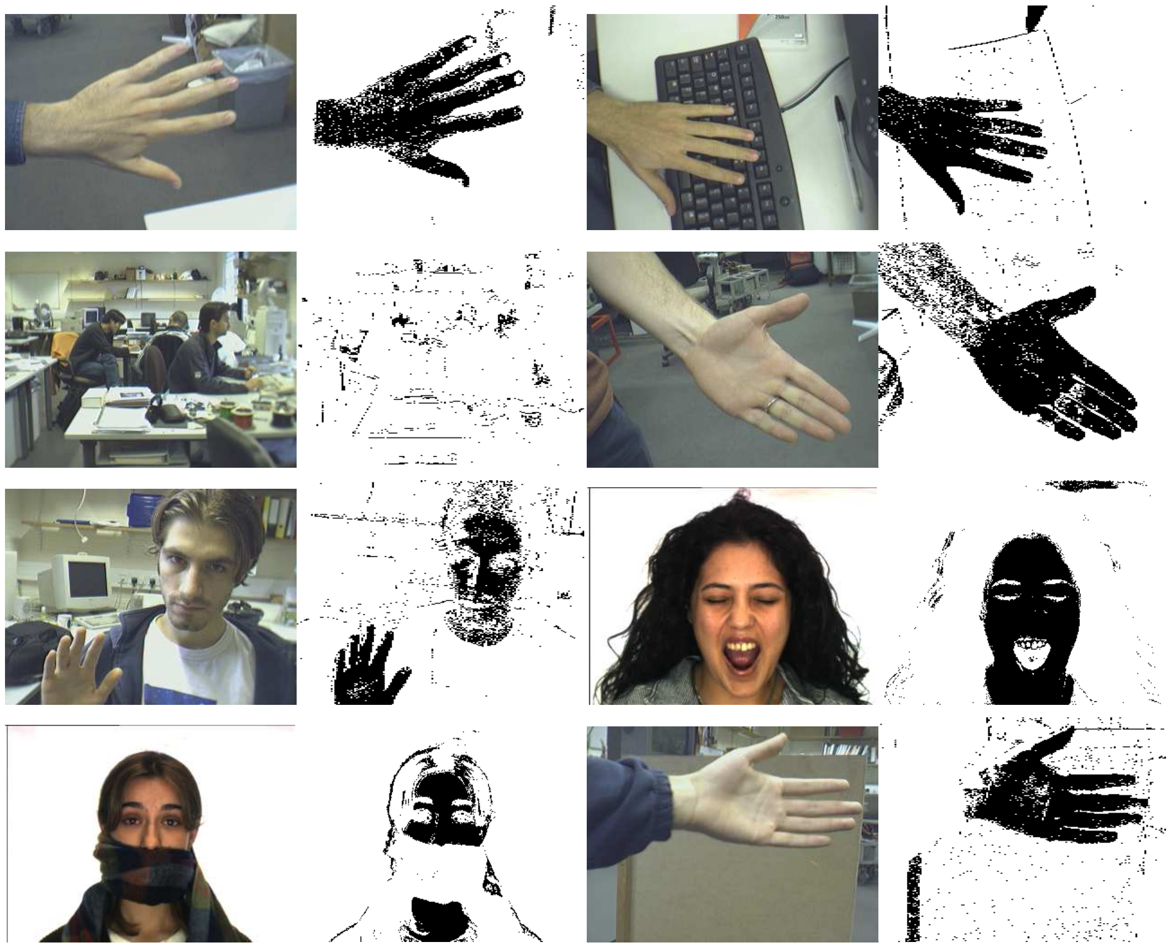


Figure 3.15: Input images and their classification results. Some of them present challenging background with wooden objects whose colour is similar to skin colour.

3.3.5 Sources of noise and dealing with them

Even when the illumination is controlled and the camera parameters are static, there are several sources of noise that lead to miss-classification. Table 3.3 lists some of the sources of noise and some possible methods to reduce their effect.

Source of noise	**Alleviated by**
Saturated white pixels and black shadows	Adaptive iris
Compression artifacts	Interfaces with no compression
Light oscillations	Increase generalisation of the classifier
Colour subsampling	Smoothing or morphological operations

Table 3.3: Sources of noise and methods that can be used to reduce their effect.

One of the less expected of these is a systematic mis-classification of pixels at the edges of objects, where the colour appears to belong neither to the object nor the background. In digital colour cameras, the image is acquired on a planar CCD array composed by grey level photosensors laid behind colour filters. These colour filters are usually arranged in the Bayer pattern [Bay76], as shown in Figure 3.16. Each pixel is composed of four subpixels, one red, one blue, and *two* green. These proportions acknowledge the human eye's greater resolving power in green light.

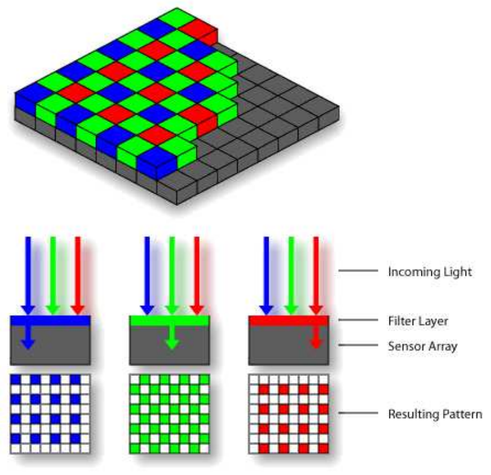
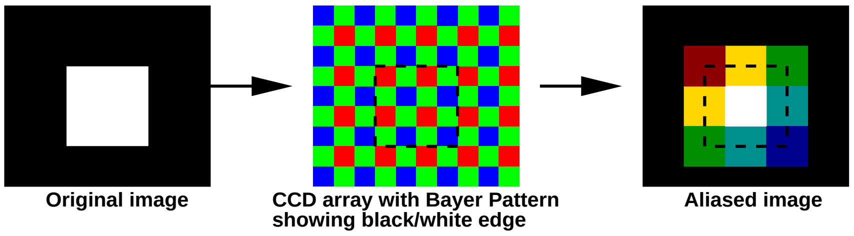
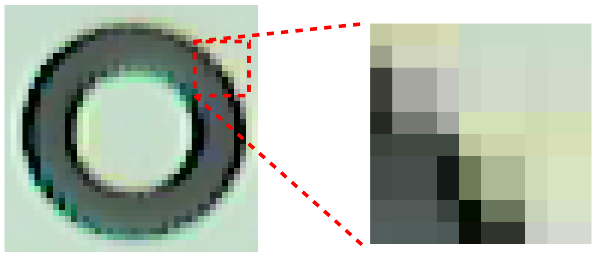


Figure 3.16: Bayer arrangement of colour filters on the pixel array of an image sensor.

Spatial aliasing occurs at sharp edges since each colour is acquired from a different position. This is illustrated in Figure 3.17(a), where pixels on the edge of a white square on a black background are assigned to intermediate colour tones. An example from a real image in 4:1:1 YCbCr format is shown in Figure 3.17(b), where various tints are visible around the ring.



(a)



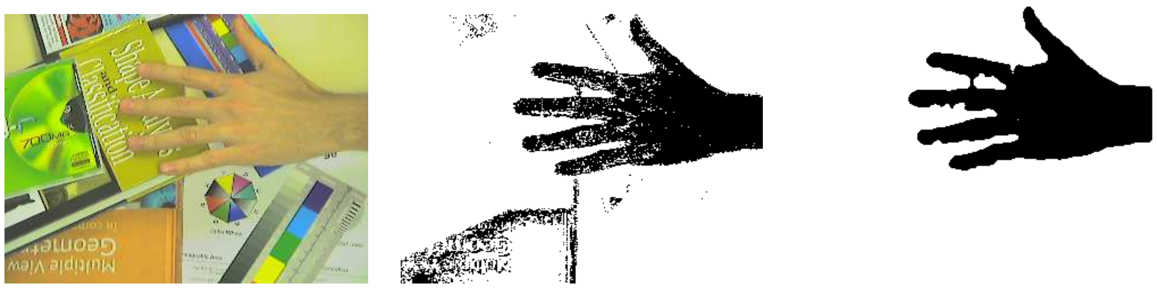
(b)

Figure 3.17: (a) An illustration of colour aliasing due to subsampling for CCD arrays with Bayer filters. (b) Detail of a YCbCr 4:1:1 image of a black disc on a white background where colour aliasing is evident near the edges. Subject to limitations of colour reproduction on paper, the enlarge region is tinted yellow.

3.3.6 Noise reduction

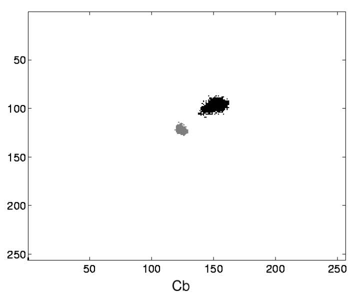
A variety of more or less principled methods can be applied to reduce noise, all implementing spatial low-pass filtering of some sort, and all assuming that the hands are of substantial size in the image. Routinely used are computing connected regions and thresholding on their area, followed by median filtering (e.g.[GW00]). Figure 3.18 shows a typical result of applying both these techniques. Among other possible alternatives is the successive application of opening and closing morphological operations.

Another approach (perhaps one that is acceptable only because the emphasis of research is elsewhere)

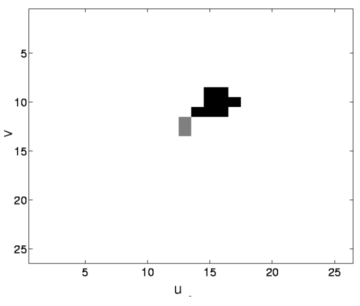


(a) (b) (c)

Figure 3.18: Noisy classification results, such as that shown in (b), can be improved using large blob segmentation followed by the median filter, leading to the result shown in (c).



(a)



(b)

Figure 3.19: Classification under more constrained situation: (a) skin and background clusters; (b) classification look up table obtained using bin of size 10. Black represents skin colour and grey represents background.

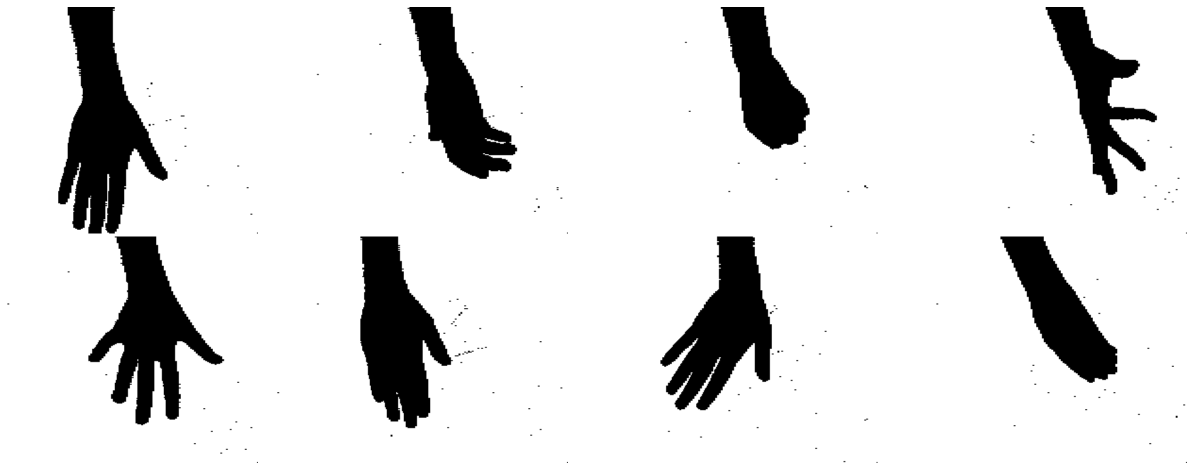


Figure 3.20: Samples of segmented hand images of a single user from a simple background (wooden table). No post-processing was applied to the images.

is to apply the low pass filter to the environment, reducing the variability in the lighting, the degree of clutter, and cameras settings. In this kind of situation, the clusters of skin and background can be very compact and have little intersection. As an example, Figure 3.19(a) shows the clusters obtained from a background that consists of a dark wooden table, and from the hands of four subjects under stable illumination. Note that the clusters are so well separated that a simple threshold in the Cb channel could classify them. To allow more generalisation and noise in the background, the histogram-based classifier was used with bins of size 10, making the lookup table very compact and classification very fast: the average processing time for 640×480 images was 10ms using a 1 GHz Pentium 4. The resulting hand segmentations are shown in Figure 3.20, where no post-processing has been applied.

3.3.7 A note on adaptation

Although brightness normalisation provides robustness to light intensity variations, it will not account for changes in the colour of the ambient lighting or reflected light. There are two approaches to retaining the compact skin clusters already computed: first, adapt the camera parameters to the environment, so that the colour appearances are the same as during training; or, second, adapt the lookup table to the illumination conditions by using samples of skin coloured objects in the image.

The first option is achieved by the process of *white balance*, which is usually implemented in the hardware of video cameras or digital still cameras. White balancing a camera is done by acquiring the image of a white region of the scene. The camera then shows true white as white and adjusts all the other colors accordingly [WS00]. The second consists of locating an area of the image which is known to be skin coloured. Once such area is located, the maximum and minimum values in each channel (CbCr) in this area can be used to translate and scale clusters of the training set and the lookup table can be updated. In [May04], for example, a skin patch is selected manually in the beginning of acquisition, and in [SWP98b] part of the user's face is guaranteed always to be visible in the bottom of the image. Alternatively, a face detection method that works independently of colour such as Viola and Jones' method [VJ01] could be used to locate a skin patch. For situations in which the only skin coloured object are hands, Han *et al*. [HASW06] collect samples of skin from the first frame using a rough skin colour model in RGB space. A region grow-based algorithm is applied to collect more training samples of skin and these are used to train a SVM classifier.

3.4 Summary

In this chapter, the software tools and apparatus used throughout this thesis have been described.

First, the architecture and software system underlying the video-rate tracking of multiple objects with multiple cameras was outlined. This is a modular system, built in the context of a larger project, that can distribute tasks over processors and communicates results via sockets.

The use of calibrated cameras facilitates model-based tracking of objects in 3D using multiple cameras. The second section of the chapter described the two methods of camera calibration. The first method, based on a 3D calibration grid, was used to recover the internal calibrations of the three Sony digital cameras over their complete range of zoom settings. In order to avoid having to repeat this complete calibration if the cameras were moved, a second method based on a planar tile was used to recover the external parameters. A method for detecting and determining the alignment of the planar object was described, and the accuracy of recovery when three cameras are used was assessed.

The last section of the chapter outlined the classification method used to detect initially where the hand is located in the image. Because skin varies predominantly in brightness, not colour, brightness normalisation reduces the dimensionality of the pixel and makes skin colour clusters more compact, while retaining distinguishability from background pixels. Results in Appendix A confirmed the view in the literature that there is little difference in terms of colour separatation between those colour spaces that use brightness as one of its axes, so the choice of colour space was determined by computational cost.

Classification of colour pixels was achieved by learning the likelihood of a pixel of some colour arising from a particular class, and using Bayes' rule to determine the posterior. This is a simple and effective method that is able to model classes that have multimodal and discontinuous PDFs, essential for large datasets of skin colour pixels acquired with different cameras under different illumination, and essential for modelling the background class. The bin size of the histogram can be set to be inversely proportional to the number of training samples. The larger the bin, the more general is the model and the faster is the classifier.

Real-time tracking of rigid objects

A method to track known rigid objects in 3D is described. This is based on the RAPiD tracker, proposed by Harris [Har92a]. A sparse set of edge features is used to measure the observed image movement. An efficient search method is used and the pose update is computed by solving a linear system. For these reasons, this system is very fast and a multiple view implementation can run in real-time.

Due to its speed, this tracker has been chosen as the basis to build an articulated object tracker, described in later chapters. To give background for the next developments, this chapter describes the RAPiD tracker and experiments that validate the implementation for multiple view tracking. This is followed by the description of a single view application that associates RAPiD with a detection method to locate and track a pointing hand in images. This system was applied to command the gaze direction of a wearable active camera.

4.1 Introduction

The use of prior information about an object's shape is the essence of model-based vision. This chapter describes a method to track known rigid objects in 3D which is based on Harris' RAPiD Tracker, proposed in [Har92a].

This chapter begins with a description of this method using the notation of Thompson *et al.* [TRMM01] (Section 4.2) and follows by describing some experiments to evaluate the accuracy of this tracker for a calibrated multiple view situation (Section 4.3). Section 4.4 describes a method that associates colour information with the RAPiD tracker to estimate the pointing direction of a hand from a wearable camera's viewpoint. This chapter finishes with a summary in Section 4.5. The main contributions of this chapter have been published in [dMM06].

4.2 RAPiD tracker

RAPiD (Real-time Attitude and Position Determination) is a model-based three dimensional tracking algorithm for a known rigid object executing arbitrary motion [Har92a]. The basic assumption of this tracker is that the change between the current estimated pose and the actual pose is small enough (i) to allow linearisation of the solution and (ii) to make the problem of matching edges straightforward. It is also assumed that the cameras are calibrated and the object to be tracked is specified in a Euclidean coordinate system.

Deriving and using a linear pose update was not the only contribution of Harris' RAPiD tracker. A number of early model-based trackers (e.g. [Gen92, Low92, Sul92]) recovered image features explicitly, such as straight lines, throughout each image, and then adjusted the pose of the model so that the distance between projected and observed features was minimised. Harris [HS90, Har92a] made video-rate tracking possible on meagre general purpose hardware by being far more parsimonious with the use of the image data. He searched perpendicularly from just a few control points on projected lines to locate nearby image edges and then adjusted the pose to minimise the summed squared-distances, a method to become the norm in active contours. Although processor speeds have since increased a hundredfold, economy still remains an issue as the the number of control points to handle complex and multiple objects has increased similarly.

4.2.1 Scene and projected image motion

A 3D object is described in its own coordinate frame 0 as a set of control points $\mathbf{X}^0 = [X^0, Y^0, Z^0]^\top$. These points may be genuine points on the object, but more usually they are parametrised locations on fixed crease or albedo edges, or are generated on the fly as extremal edges of a curved object, as sketched in Figure 4.1. To allow multiple and possibly moving cameras to be treated equally, the object's pose will be one or other representation of the rotation and translation $\{\mathtt{R}_0^W, \mathbf{t}_{0W}\}$ that take points in frame 0 into points in a fixed world frame W. The position of each camera C$_k$ is defined similarly by $\{\mathtt{R}_W^C, \mathbf{t}_{WC}\}_k$.

so the pose of the object in a camera coordinate frame is represented by $\mathbf{X}^{C_k} = \mathtt{R}^W_{C_k}\mathbf{X}^W + \mathbf{t}_{WC_k}$.

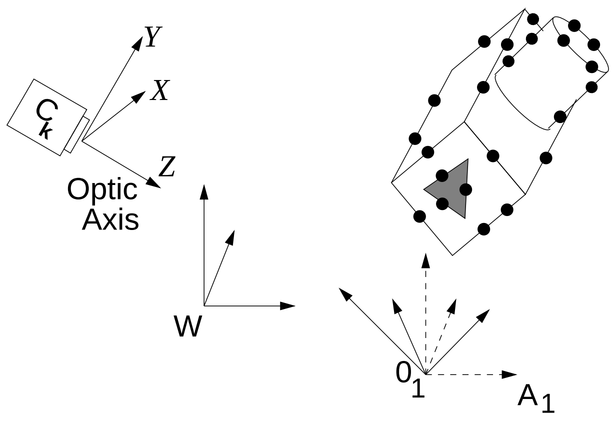


Figure 4.1: Each object is modelled within its own coordinate frame 0 as a set of control points lying on edges, which may be crease, albedo or extremal edges. C_k, W and A_n are the k-th camera, the world and n-th object aligned frames, respectively.

As a matter of convenience, let A be a frame that is aligned with W but has its origin coincident with the object frame 0: $\mathbf{X}^A = \mathtt{R}^W_0\mathbf{X}^0$. Then, the pose of an object $\mathbf{X}$ at some instant is described in the world frame by

$$\mathbf{X}^W = \mathtt{R}^W_0\mathbf{X}^0 + \mathbf{t}_{0W} = \mathbf{X}^A + \mathbf{t}_{0W}, \tag{4.1}$$

The object's angular and rectilinear velocities are defined in the frame A and represented by $\boldsymbol{\omega}$ and $\boldsymbol{v}$ respectively. The object's instantaneous velocity in the world frame is then:

$$\begin{aligned} \dot{\mathbf{X}}^W &= \left(\left[-\mathbf{X}^A\right]_\times \mid \mathtt{I}_{3\times 3} \right) \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{pmatrix} \\ &= \mathtt{H}\,\mathbf{s}\,. \end{aligned} \tag{4.2}$$

The antisymmetric matrix $[\mathbf{a}]_\times$ is such that $[\mathbf{a}]_\times\mathbf{b}$ is the vector product $\mathbf{a} \times \mathbf{b}$. The velocity in the world frame is transformed into a camera frame as

$$\dot{\mathbf{X}}^C = \mathtt{R}^C_W\mathtt{H}\mathbf{s}\,. \tag{4.3}$$

The projection into a normalised image (i.e., one corrected by the intrinsic calibration to have focal length and aspect ratio unity, and origin at the optic centre) is $\mathbf{x} = \mathbf{X}^C/Z^C$, and so the projected motion

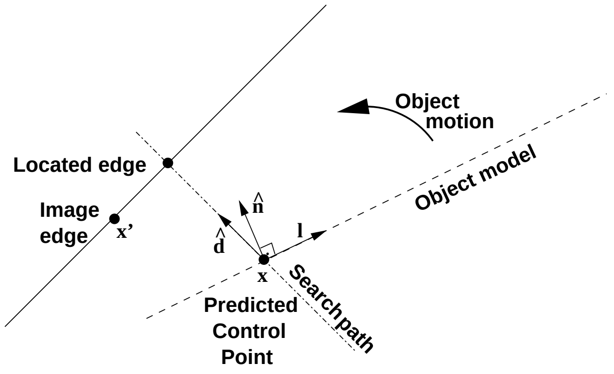


Figure 4.2: The search from the predicted control point $\mathbf{x}$ is along a convenient direction $\hat{\mathbf{d}}$ close to the edge normal. For fast image search, $\hat{\mathbf{d}}$ may be taken as one of the eight cardinal directions.

is

$$\begin{aligned} \dot{\mathbf{x}} &= (1/Z^C)(\dot{\mathbf{X}}^C - \mathbf{x}\dot{Z}^C) \\ &= (1/Z^C)\,[\mathtt{I}_3 - \mathbf{x}[001]]\,\mathtt{R}_W^C\mathtt{H}\mathbf{s}\,, \end{aligned} \tag{4.4}$$

where $\mathtt{I}_3$ is the 3×3 identity matrix and $\mathbf{x}[001]$ is an outer product.

4.2.2 Measurements of edge-normal motion

Full motion vectors $\dot{\mathbf{x}} = (\mathbf{x}' - \mathbf{x})$ could be used to recover the screw $\mathbf{s}$ with three or more point to point matches. But it is more usual to match control points to image lines or curves (with relatively large curvature radius), and the resulting aperture problem requires measurement of the projection of $\dot{\mathbf{x}}$ onto a direction $\hat{\mathbf{n}}$ normal to the edge on which the control point lies [TRMM01], as shown in Figure 4.2. The measurement equation for each control point i becomes

$$\begin{aligned} \frac{1}{Z_i^C}\hat{\mathbf{n}}_i^\top\,[\mathtt{I}_3 - \mathbf{x}_i[001]]\,\mathtt{R}_W^C\mathtt{H}_i\mathbf{s} &= \hat{\mathbf{n}}_i^\top\dot{\mathbf{x}}_i \\ \text{or} \qquad \mathbf{f}_i\mathbf{s} &= d_i\,, \end{aligned} \tag{4.5}$$

Harris [Har92a] pointed out that if the pose change is small, the predicted and located lines (or curves) are close to parallel and the perpendicular distance $\hat{\mathbf{n}}^\top\dot{\mathbf{x}} \approx d\hat{\mathbf{n}}^\top\hat{\mathbf{d}}$, where d is the distance between the $\mathbf{x}$ and the located edge along any unit vector $\hat{\mathbf{d}}$. For fast image search, $\hat{\mathbf{d}}$ is taken as one of the eight cardinal directions which is closest to $\hat{\mathbf{n}}$.

Each edge measurement builds up a row of $\mathtt{F}$ and an element of $\mathbf{d}$ in this equation:

$$\mathtt{F}\mathbf{s} = \mathbf{d} \tag{4.6}$$

and each distance measured gives a one-dimensional motion constraint, so it is required to stack at least six measurements to the equation above before solving using some variant of least squares (further details are given in Section 5.4).

Note that this system is straightforwardly extended for multiple views. For each camera C, a different set of projected control points $\mathbf{x}_i^C$ is created and Equation 4.5 is computed. The kinematic screw vector $\mathbf{s}$ is specified in the aligned object frame A, so it is the same for all cameras. Thus, each control point from each camera adds a row to the system of Equation 4.6.

4.2.3 Control points and visibility calculation

To avoid ambiguity, if two or more edges of equal gradient intensity are found along the search path, the nearest to the prediction is chosen. The control points along curves or lines are not fixed in object coordinates. The spacing between projected control points is fixed in the image, and this is what dictates the number of control points per line of the object, and thus their position. This avoids making repeated or overlapping measurements along object lines that are nearly parallel to the camera's optic axis, a repetition which would skew the weighting.

Although visibility calculations are performed for rendering in the hardware of graphics accelerators, it is not straightforward to match the resulting rendered object with the track-able representation of individual control points – an explicit relation between their position in the image $\mathbf{x}$ and their position in the 3D object $\mathbf{X}$ is needed. For this reason, visibility calculations are implemented as part of the process to compute control points. Since the number of control points is much smaller than the number of pixels in the object, this process is not too expensive.

Polyhedral objects

For convex polyhedral objects, visibility calculation can be computed for whole control lines rather than control points individually. A control line is visible if at least one of its neighbouring faces has its outward normal facing the camera.

For concave objects and multiple objects of any shape, some extra processing has to be done. Each control point from lines that were considered visible in the step described above has to be tested against the other faces of the object in order to verify if one of them is occluding it. This is done by ray tracing:

if the line that links the control point and the centre of the camera crosses one of the faces of the object, this control point is occluded.

To avoid ambiguity caused by lines of the same face that are projected too close to each other, when only one of the neighbouring faces of a line faces the camera, the angle between the normal of this face and a vector that goes from the control line to the centre of the camera is verified. If this angle is close to 90°this line is not tracked. In the example of Figure 4.3, the edges of the back lines on the top of the object were not tracked for this reason[1].

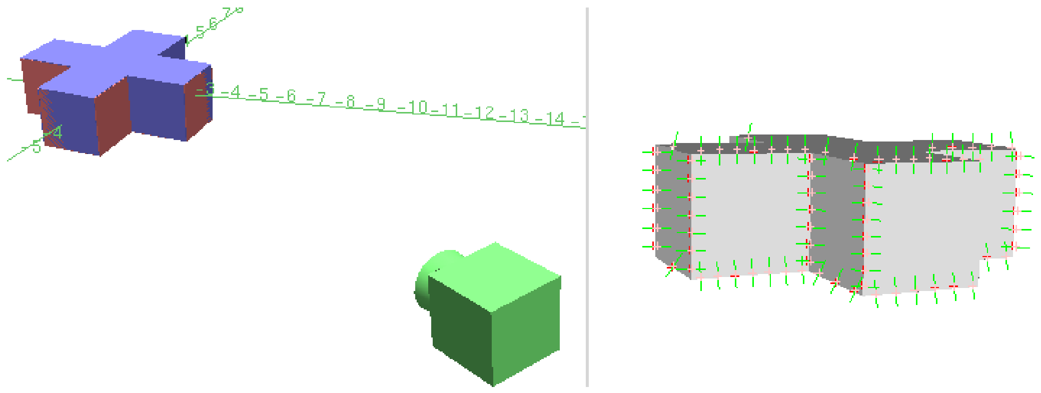


Figure 4.3: Tracking a synthetic synthetic cross-shaped object. **Left**: 3D view showing the camera and the object: the object in blue is at the at the synthesised ground truth pose, this is superposed by the object at the estimated pose in red. **Right**: camera view with the projection of the predicted control points (pink crosses), the normals to the lines (green lines), and the located control points (red crosses).

Rings, spheres and cones

Planar circles are parametrised by position, radius and orientation. The number of control points is computed according to the size of the projected circle in the image and this is used to establish the angular spacing in which a control point must be created on the circle. Discs are composed by an inner and outer circle and a zone in between them that occludes objects behind the disc. To verify if a given control point is occluded by a disc, first it is verified if the control point is behind the disc's plane. Next, it is checked if the point is in the "shadow" of the disk. For that, the angle between the camera centre, the disk centre and its bounding contour is compared with the angle between the control point being tested, the camera centre and the disc centre. An additional test is done to verify if the control point is within

[1]For detailed description of computational geometry methods involved in the visibility calculation, see [O'R98].

the “hole” of the disk.

For spheres, cones and quadric objects, a method to compute occluding contours is described in [Ste04]. However this involves factorisation of matrices for each object component. In this thesis, a simpler method was employed by assuming that the bounding contours are in the intersection between the object and a plane that crosses the part centre. For spheres, this plane is orthogonal to the camera axis. This approximation saves some computation at little expense in the accuracy if the camera is not too close to the sphere. The localisation of control points is then computed in the same process as that of circles. To check occlusion, three tests are used, the computationally cheapest is first: (i) Is the point within the sphere’s radius? (ii) Is it closer to the camera than the sphere centre? (iii) Is it further than the sphere and in the “shadow”?

For truncated cones, a plane on the cone axis is used and the bounding contours are defined by two lines and two circles. The control points are created following the normal process for polyhedrals and circles. Occlusion handling is performed using these tests: (i) if the point is beyond the ends of the cylinder it is not occluded; (ii) if the point is within the cylinder radius it is definitely occluded; (iii) if the point is closer to camera than the leading edge of the cylinder it cannot be occluded; (iv) if the ray to the point passes through the cylinder contour and the distance from the camera is greater than the cylinders, it is occluded.

4.2.4 Robust methods

The use of robust methods based on RANSAC (Random Sampling Consensus) [FB81] to improve the performance of polyhedral tracking was explored in [AZ95, HM99, TRMM01]. In this thesis, robust methods are used in two steps of RAPiD: to select collinear points (robust collinearity), and to compute the whole object’s pose (robust pose). RANSAC requires that the standard deviation of measurements is known a priori to discriminate between inliers and outliers. Here, this value is not known, so robust estimation is performed using Least Median of Squares (LMedS) [RL87].

For robust pose, this consists of selecting minimal sets of measurements (six in the case of estimating $\mathbf{s}$) chosen randomly to compute model parameters. These parameters are then used to compute the deviation between the fitted estimate and all the measured control points $e_i = |d_i - d_i^{fitted}|$. This is iterated, and the solution with the smallest median of e is used to estimate the standard deviation of the

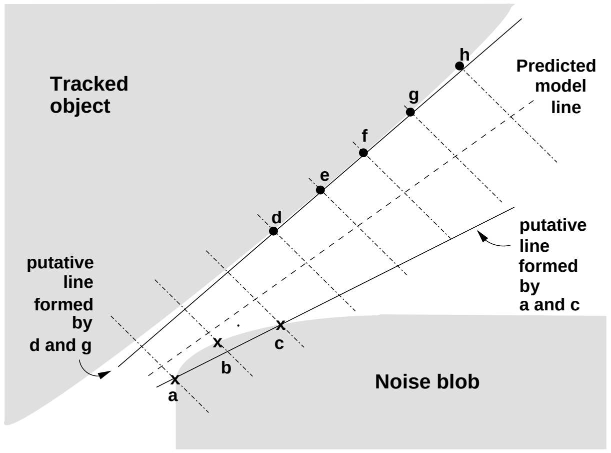

Tracked object
Predicted model line
h
g
f
e
d
putative line formed by a and c
putative line formed by d and g
x
a
x
b
x
c
Noise blob

Figure 4.4: Robust collinearity: rejecting outliers.

data. This obviously requires that at least 50% of the measurements are inliers. The standard deviation is estimated exploiting the fact that $\frac{1}{\phi^{-1}(0.75)}\sqrt{med|e_i|}$ is an asymptotically consistent estimator of σ when e_i are distributed like $N(0,\sigma^2)$, where ϕ is the cumulative distribution function for the Gaussian pdf, thus

$$\sigma = \frac{1}{\phi^{-1}(0.75)}\sqrt{\text{med}|e_i|} = 1.48\sqrt{\text{med}|e_i|} \tag{4.7}$$

Then the measurements are split between inliers and outliers according to [RL87]:

$$i \in \begin{cases} \text{inliers} & \text{if } |e_i| \leq 1.96\sigma \\ \text{outliers} & \text{otherwise} \end{cases} \tag{4.8}$$

The final estimate of s is obtained using all the inliers. The 1.96 coefficient was derived in [TRMM01] and it is typically of order 0.03 in the ideal image with focal length unity. Using the Rousseeuw formula in reverse, they found that this translate to $|e_i| \sim 1.2$ pixel in a physical image using cameras with focal length of around 3000 pixels, which is also a typical value for most of the experiments presented here. This is a commensurate with the edge search mechanism, which operates only to ± 1 pixel accuracy.

The expected confidence P that a valid minimal set of m features will be selected after I trials when the fraction of valid data is ψ is estimated by

$$P = 1 - (1 - \psi^m)^I \tag{4.9}$$

It is desired that P be as close to 1 as possible. The values were chosen empirically, as explained later.

Robust collinearity [AZ95, HM99] aims to avoid edges mismatches when locating control lines. In situations like that shown in Figure 4.4, background noise or even other parts of the object being tracked can cause mismatches on locating a control line (edges a, b and c). The robust collinearity method consists of picking pairs of points at random to define a line, and computing the perpendicular distance from each edge location to the line. The standard deviation is computed and used to calculate a threshold on the perpendicular distance that is used to determine the inliers and outliers. In the example of Figure 4.4, the control points d–h are correctly matched to the image line, but the points a, b and c are mismatched to another edge. Robust collinearity detects these control points as outliers, and they should not be considered in the calculation of the change in pose.

The combination of robust collinearity and robust pose is important because robust pose requires more measurements in the minimal set, which means that more iterations are needed. If a number of outliers is eliminated with robust collinearity, the estimate of the fraction of valid data ψ for robust pose is increased. A sample situation in which robust collinearity alone is not enough occurs when the object has two lines projected near each other, and one of them has weak edges in the image. Robust collinearity may fit all the points to the same line, then robust pose eliminates the mismatching points that should have matched the weak line. This combination has been explred in detail in [KD05].

4.2.5 Convergence and filtering

Thompson *et al*. [TRMM01] show that because of the approximate nature of the linearisation, it is useful to iterate the solution within each image, allowing the pose of the object to converge to a more accurate results. This approach has been used in some of the experiments detailed later.

During the iterations of robust pose, some minimal sets lead to hypotheses of $\mathbf{s}$ with unrealistically fast motion. To save computation and avoid unstable motion estimates, these hypotheses are eliminated without being checked against all the measurements. An alternative to improve the stability of the motion estimate is to damp Equation 4.6 by modifying it to:

$$\begin{pmatrix} \mathtt{F} \\ \lambda\mathtt{I} \end{pmatrix} \mathbf{s} = \begin{pmatrix} \mathbf{d} \\ \mathbf{0} \end{pmatrix} , \qquad (4.10)$$

where λ is a damping factor [WL88]. As before, a least squares solution method is applied to estimate $\mathbf{s}$. This formulation assumes regularised entries in $\mathbf{s}$.

To deal with fast (but smooth) motions, or low frame rates, Harris [Har92a] adopted the Kalman filter (e.g. [Bro83]) to maintain an estimation of the motion and improve the tracking results using prediction of the pose.

The experiments presented in this chapter are interested on verifying the accuracy of the pose estimation assuming that the frame rate is high enough so inter-frame motion is small. For this reason, damping and filtering have not been used. Another reason for this choice is that the intended application of tracking hands interacting with objects shares some features with teleoperation (e.g. contact and abrupt motions). In teleoperation, the application of a filter with a motion model such as constant velocity has proven less satisfactory [MRT98].

4.3 Multiple view experiments

In this section, experiments with multiple cameras are described. Section 4.3.1 shows experiments with synthetic images and Section 4.3.2 describe an experiment with real images.

4.3.1 Synthetic images

To evaluate the RAPiD tracker it is necessary to compare its pose recovery results with ground truth data. A convenient way to obtain accurate and reliable ground truth data is to build an artificial three dimensional scene such as that shown in Figure 4.5. In this scene, the object was placed in the centre of the world coordinate system (position $[0, 0, 0]$), and three cameras were placed in regular intervals of a circle on the plane XZ, centred in $[0, 0, 0]$.

Tracking experiments with six polyhedral objects were performed: a cube, a parallelepiped, a cross (shown in Figure 4.3), an L-shaped object, a pointed object (shown in Figure 4.5) and an object composed of two separated cubes. The background was black and the objects were textured with a uniform colour. The artificial light sources were placed at positions that allow the appearance of edges in the object's surface creases.

The experiments were performed on 50 frames long sequences, one sequence per object. In these sequences, the first pose of the object is known by the tracker. In each frame, a rotation of 1.8^o about the axis of vector $[1, 1, 1]^\top$ indicated in Figure 4.5 was applied to the object.

The parameters of the Harris tracker were chosen so that real-time processing performed (i.e. each frame could be processed in less than $\frac{1}{30}$s). In a 1.8GHz Pentium 4 machine the following parameters were adopted: 12 pixels of spacing between control points in the images, 21 pixels of search path for

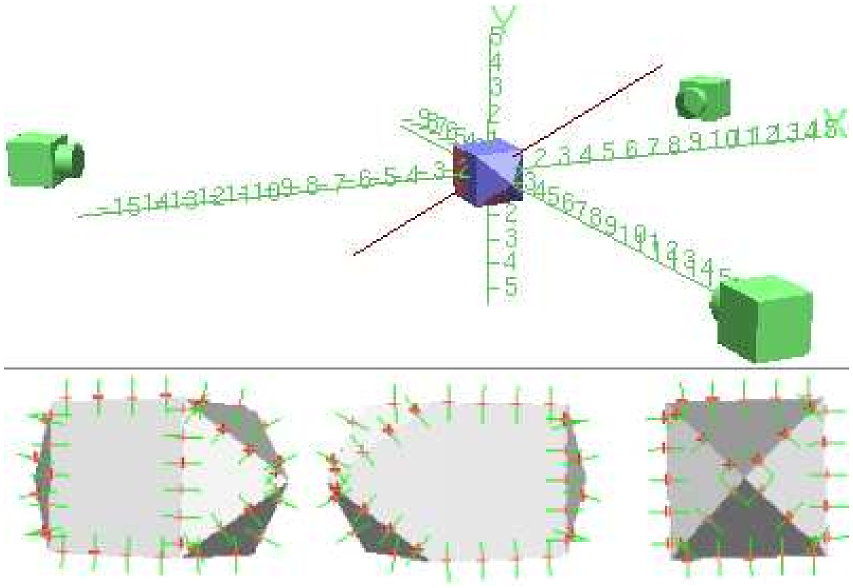


Figure 4.5: **Top**: artificial scene showing the cameras, the pointed block object and its rotation axis (dark red line). **Bottom**: views of the three cameras, with the same symbols used in Figure 4.3.

control points. The robust collinearity method was iterated 3 times and the robust pose 10 times. These values result from assuming $\psi = 80\%$ of inliers and $P = 95\%$ of confidence for robust pose. For robust collinearity, 3 iterations are enough to give $P = 99\%$ of confidence assuming that $\psi = 60\%$ of the data are inliers (see Equation 4.9).

The experiments were performed using 400×300 pixel images and cameras with focal length of 700 pixels. The distance between each camera and centre of the object was 18 metric units, and the diameter of the objects ranged between 3.46 (cube) and 7.48 (parallelepiped) metric units. This means that parts of the biggest objects exceeded the field of view of some cameras for a range of rotations[2].

For 300 frames, the mean processing time per frame was 13.4ms, with standard deviation of 6.1ms and maximum of 31.9ms (occurred when 241 control points were used to track the cross-shaped object). Since the ground truth motion is the same for all the objects the evaluation results have been computed together. The accuracy evaluation quantities of Section 3.2.4 have been used. The results are shown in in Figures 4.6, 4.7 and 4.8 for translation, angle of rotation and axis of rotation, respectively.

Table 4.1 gives an overview of the results obtained in these tracking sequences. Translation error is

[2]Note that $fd/Z + x > h$, where f is the focal length, d is the diameter of the object, x is the position of the object in the image (in this case, the centre of the image) and h is the height of the image. This also happens in the horizontal direction of the image for the longest objects.

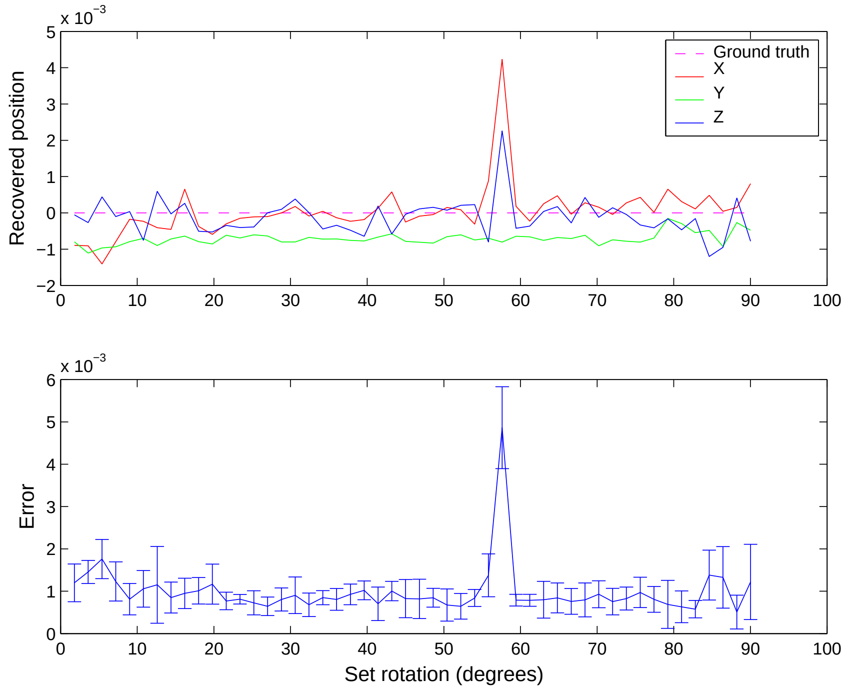


Figure 4.6: **Top**: mean of the recovered translations (X, Y and Z) for synthetic images (in a metric unit normalised by distance between the cameras and the objects) versus set rotation (in degrees), in comparison with the ground truth data. **Bottom**: mean of the error (solid line) and inaccuracy (bars).

		$\Delta_{\mathbf{t}}/dist$ ($\times 10^{-3}$)	$\Delta\theta^o$	$\Delta\alpha^o$
Error	mean	1.0	0.7	1.7
	std	0.6	0.9	2.7
	max	4.7	3.7	15.5

Table 4.1: Synthetic tracking results: mean, standard deviation and maximum error in the estimate of translation $\Delta_{\mathbf{t}}/dist$, angle $\Delta\theta^o$ and axis of rotation $\Delta\alpha^o$.

normalised by the distance between the camera and the object so if this distance is 1m, the mean error of the position estimative is $\sim$ 1mm, and the expected error of the orientation estimative is $\sim 1.7^o$. Since these synthetic images created for the tests described here do not simulate the noise that is usually present in real images, the error obtained is attributed to the linearisation done and to the pixelation due to the relatively low resolution used.

The quirk in the estimate of X and Z that happens at rotation of 59°(shown in Figure 4.6) hapenned because this is a critical pose for the object models and the rendering setup used. All these objects are made predominantly by cubes, and at that pose, some of the albedo and crease edges become invisible

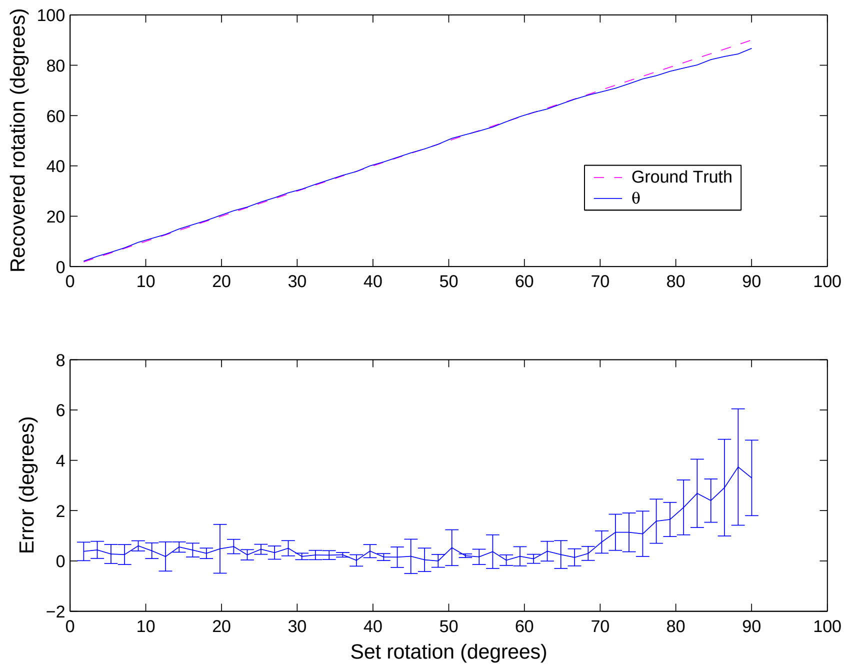


Figure 4.7: **Top**: recovered rotation (θ) versus set rotation (in degrees) around the fixed rotation axis in comparison with the ground truth data. **Bottom**: mean error and standard deviation.

due to the lack of contrast caused by the position of the light source. At that pose, some of the planes of the objects are close to aligned with the the axis of two cameras in the set, also reducing the number of located control points.

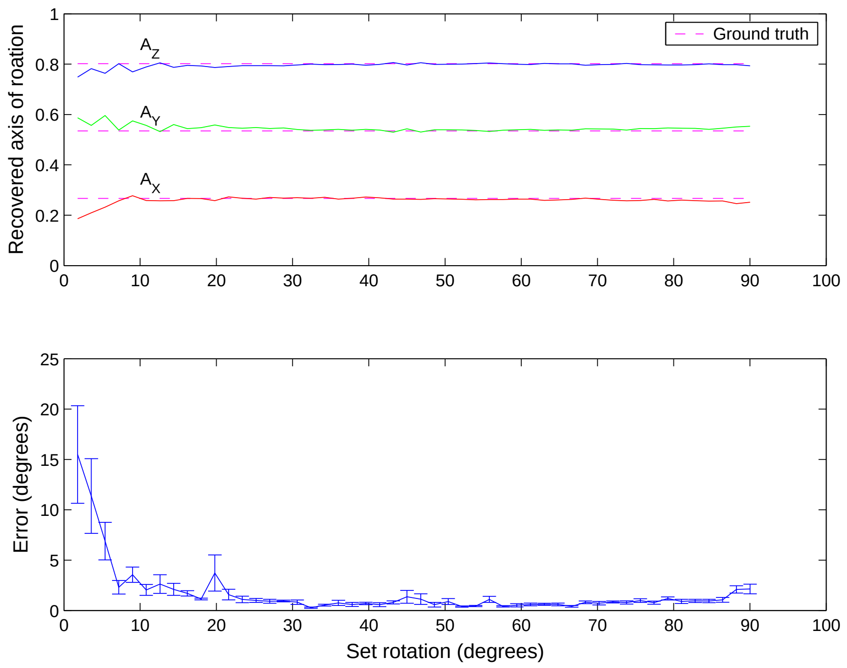


Figure 4.8: **Top**: recovered axis of rotation versus set rotation (in normalised coordinates). **Bottom**: mean error α and standard deviation (in degrees).

4.3.2 Real images

To evaluate the tracking performance for real images, we performed experiments with a sequence grabbed from the calibrated cameras in the desktop environment shown in Figure 3.6. The tracked object was a sheet of paper with the pattern described in Section 3.2 printed on it. Images of resolution 640×480 were used.

In this 140 frames video sequence, the tracked object is placed on the cover of a book that is $\sim$ 12mm thick. The sequence starts with the book on the centre of the coordinate frame, which means that the initial position of the tracked object is about $[0, 0, 12]$mm (see Figure 4.9-left). Shortly after, the cover of this book is opened up to about 10^o (see Figure 4.9-centre); then there is a small pause and the book is closed, back to the original pose. After that, a rotation about the Z axis is applied, followed by a pause when the angle of rotation was $\sim 45^o$ (see Figure 4.9-right), and a new motion combining this rotation and a translation in both X and Y is started just before the end of the sequence.

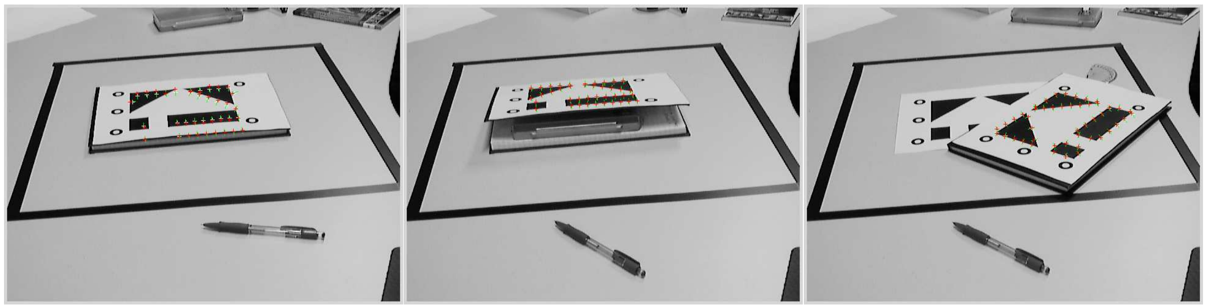

frame 1 frame 34 frame 134

Figure 4.9: Key frames viewed from camera 2 (see Figure 3.6) with the control points.

The tracker was applied with real-time parameters: the spacing between control points was such that the average number of located control points was of 128 (adding the three views). The mean of the time per frame was 15ms. The estimated poses shown in Figures 4.10, 4.11 and 4.12 visually agree with the set rotations above. For a quantitative evaluation, these results are compared with results obtained using the more expensive set of parameters that have been used for camera pose estimation in Section 3.2.4, with the difference that now multiple views are combined. With those parameters, the average number of located control points was 698. The results obtained with these more expensive parameters are taken as “ground truth”.

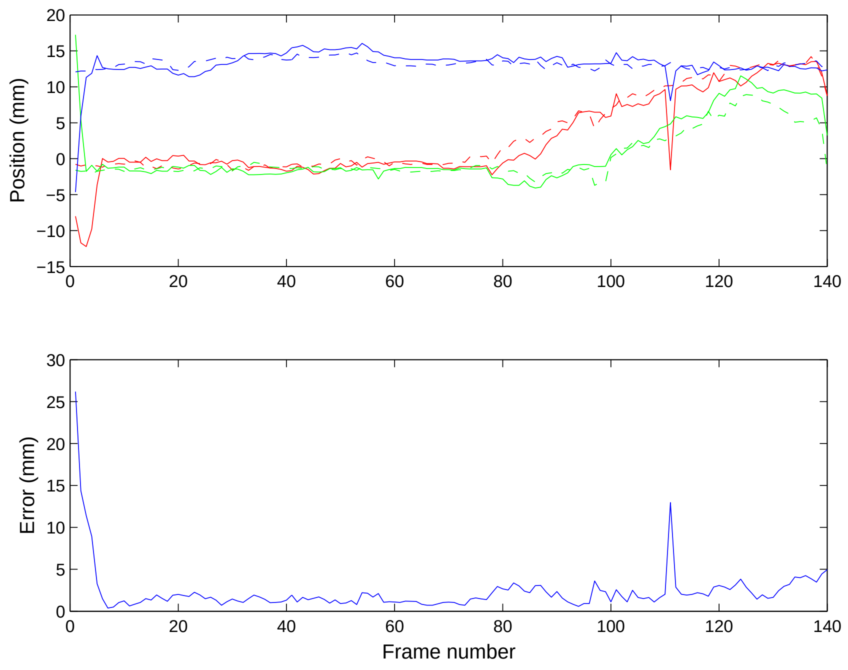


Figure 4.10: **Top**: recovered translations (continuous lines) for a sequence of real images in comparison with the ground truth data (dotted lines). X, Y and Z are represented in red, green and blue, respectively. **Bottom**: estimation error.

A robustness test regarding to error on initialisation is also presented in experiment. The tracker was initialised with the object placed on position $[0, 0, 0]$, i.e., as if the sheet of paper was laying on the desk, and not on a book that is on the desk. This is why the control points were misplaced in the view of the first frame shown in Figure 4.9. Note (from Figures 4.10, 4.11 and 4.12) that only 6 frames were enough to recover the correct pose of the object. A quirk in the tracking result happens at around frame 110 for the estimate of X and Z. This was probably caused by change in the cardinal direction of the edges search paths, but this did not affect subsequent results and other DOFs.

	$\Delta_{\mathbf{t}}$mm	$\Delta\theta^o$	$\Delta\alpha^o$
Mean	2.29	0.51	8.40
STD	2.81	0.68	19.80
Max	26.19	5.89	125.90

Table 4.2: Tracking error for a sequence of real images, showing the mean, the standard deviation and the maximum error in the estimate of translation $\Delta_{\mathbf{t}}/dist$, angle $\Delta\theta^o$ and axis of rotation $\Delta\alpha^o$.

Table 4.2 shows the mean, standard deviation and maximum error for the whole sequence (including

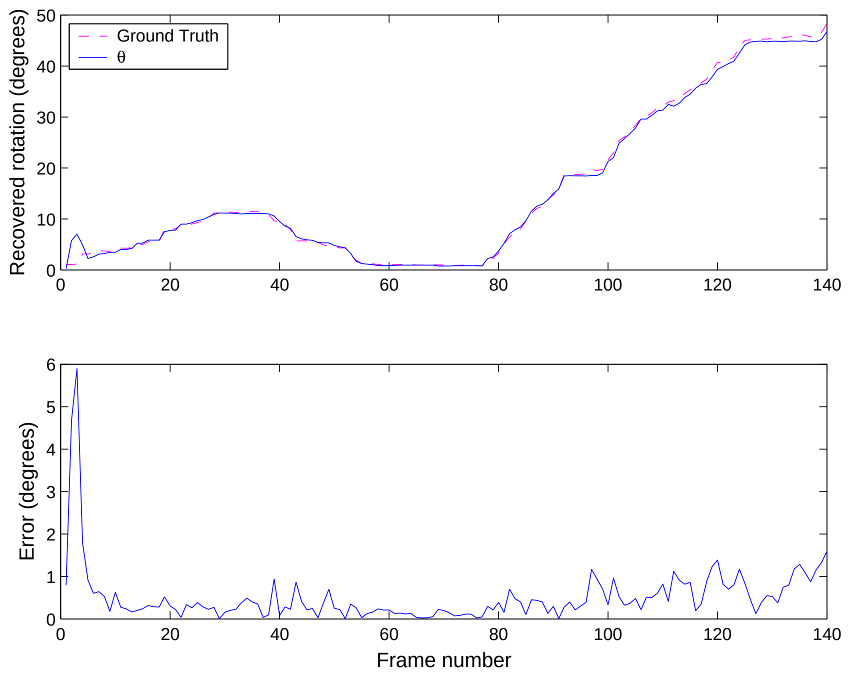


Figure 4.11: **Top**: recovered rotation (θ) in comparison with the ground truth data. **Bottom**: angle (θ) estimation error in degrees.

the error in the first frames). These values were computed from a single tracking attempt. Note that the estimated axis of rotation (angle α) sometimes presented a very high error, but the overall error was not too large because the estimated angle of rotation θ for the frames where this occurred was not large (smaller than 7°).

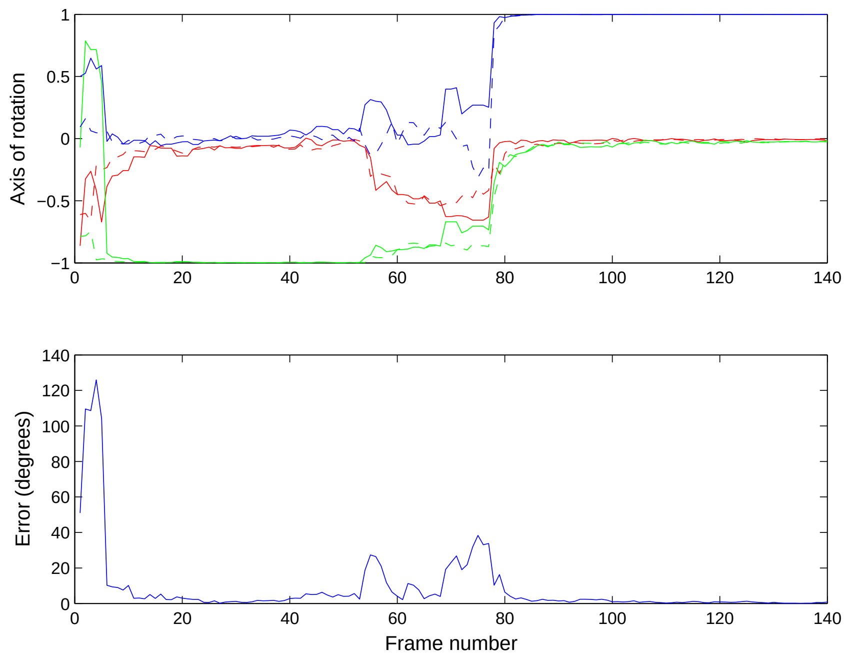


Figure 4.12: **Top**: recovered axis of rotation (continuous lines) and ground truth (dotted lines). A_X, A_Y and A_Z are represented by red, gree and blue, respectively. **Bottom**: orientation (α) error in degrees. The error in α can be very large when the angle of rotation (θ) is small.

4.4 Detection and tracking of pointing hand from a wearable camera

In previous section, it was shown that RAPiD was evaluated as a method to track rigid objects using multiple calibrated cameras in a desktop environment. In this section, a single-view application of this method is described. RAPiD is combined with an image-based shape detector which uses the skin colour detector of Chapter 3 to locate and track a specific hand shape from a wearable camera's viewpoint. Unlike the experiments described before, the camera is obviously not static. In fact, the camera is an active vision system that is worn on the shoulder of the user. The goal of this work is to aid the user interface using the localisation and tracking of deictic gestures.

4.4.1 Introduction

Recent technology allows the implementation of robotic systems that are light enough to be worn without inconvenience to the user. This leads to a wide range of applications from assistive technologies to entertainment and portable communication. Wearable active cameras provide views of the environment which are rich in information about the wearer's location, interactions and intentions. But the images from them present severe challenges because neither the sensor nor its underlying "platform" is stationary. Compounding these difficulties, most researchers use cameras that are more or less rigidly mounted to one or other body part — head, shoulder, chest and hand have all been used — making the imagery highly dependent on posture.

Mayol [May04] developed prototypes for a miniature wearable active camera, and argued that mounting it at the shoulder gives an optimum location measured against field of view, independence from the wearer's movements, and, important in wearable applications, social acceptability. The ability to redirect the camera also allows switching between sensing contexts: one context may be focused on the manipulative space; another may be the horizon, aligned with gravity; and a third may be fixated on an independently moving object. Such devices require a range of sensing and perceptual modalities. In [MTM00] inertial and visual cues are used to stabilise gaze by detecting user and image motion. In [TMdM02b] slaving the device from head motion is investigated.

In the wearable domain, hand gesture recognition is a natural replacement for keyboard and mouse-based input. In [SWP98a], for example, a hat-mounted camera is used for a sign language recognition task, and interestingly performs better than a wall mounted one, while in [KOKS01] a bare hand is used as a cursor-and-click device for interacting with menus displayed on a head mounted display. Pointing gestures are the main form of non-verbal communication, presenting a major complement to speech in human to human communication [PSH97].

The interest of this section is on using the view from the wearable camera to detect and track pointing gestures in order to determine the focus of attention and redirect the camera. In order to allow natural user interface, it is necessary to use real-time algorithms. To that end, a coarse-to-fine method for shape detection was proposed. This is invariant to translation and rotation, but retains the ability to identify position and orientation of the pointing hand. Using a cyclic finite state machine, this detection method is combined with RAPiD to refine the pose estimate and add depth information about the position and orientation of the hand. Such parameters enrich the ability of the wearable camera to perform a saccade to the pointed area in 3D.

4.4.2 The wearable camera system

The wearable active camera consists of a miniature camera mounted at the end of a serial chain of three motorised axes. As shown in Figure 4.13, the device is mounted on a collar and lies just above the shoulder of the wearer, its location was found optimal against a number of criteria. Full details about the device's kinematics and spatial layout are given in [May04].

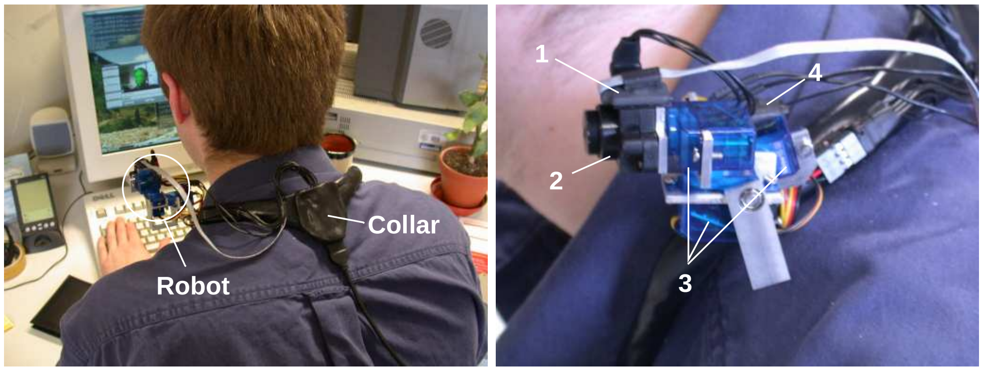


Figure 4.13: Wearable Visual Robot: (1) 2-axis accelerometer, (2) CMOS colour camera, (3) three motorised axes, (4) wireless video transmitter. The wearable interface box containing the data transceiver, micro-controllers and batteries is worn at the hip.

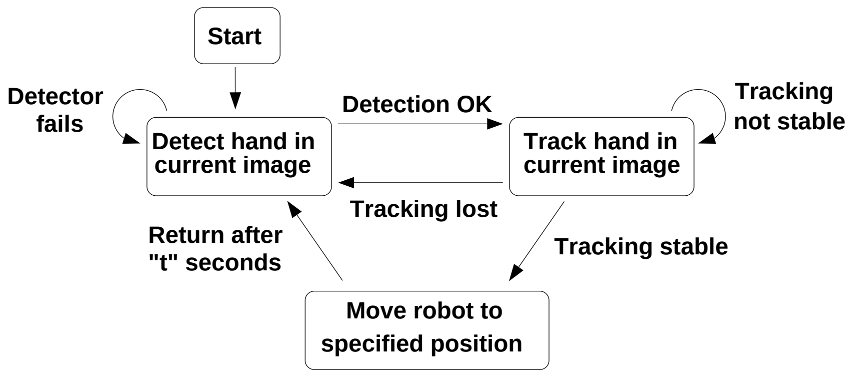


Figure 4.14: A finite state machine to combine detection and tracking of pointing hand.

4.4.3 Locating pointing gestures robustly

The hand detection algorithm is a coarse-to-fine matching method that is able to find the hand and also to estimate its pointing direction in the image plane without the need for scanning all the pixels. This is combined with the RAPiD tracker in a finite state machine shown in Figure 4.14.

Preprocessing

The first step in detection consists of skin colour detection, which is done using the method described in Section 3.3. If most of the background captured by the wearable camera is dark, then the automatic contrast normalisation of the camera can produce some saturated blobs in the hand image for Caucasian users, destroying the colour information in those regions. To finesse this problem, white saturated pixels were classified as skin. This reduces the false negative classification rate at a cost of increasing the false positive rate. But this is not critical because the hand detection method takes the global shape into account. Figure 4.15 shows the result of this method for a challenging image.

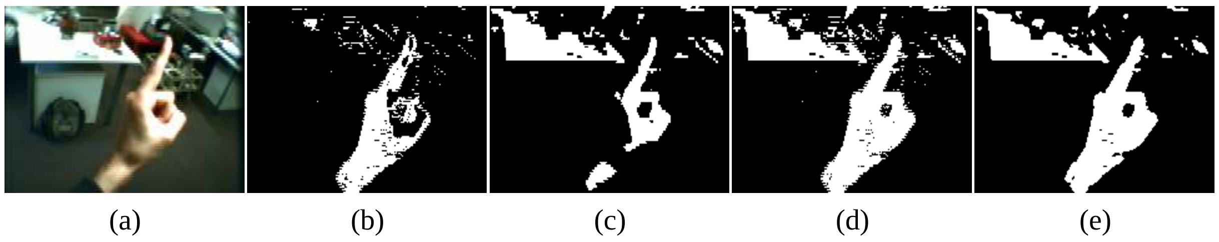


Figure 4.15: a) Original colour image, which has saturated areas and video interlacing artifacts; b) skin detection result; c) threshold result; d) combined (OR) image; e) filtered result obtained with the median filter with a 3×3 image.

Since the shape detection and tracking methods are based on control points, rather than global images, skin detection and filtering can be applied only to the regions of interest to reduce the computational cost.

Hand shape detection

Techniques for finding objects of a known shape include the use of 2D correlation, image moments, and specific spatial filters [dCJ01]. The first two methods work well when noise is small and when objects do not vary too much. But several kinds of distortion happen often in hand images: the hand can appear as a non-contiguous object due to occlusion and shadows; it can be in different orientations; other objects with similar colour and size can be present; and small variations in the hand shape can occur. To cope with these factors and with image noise generated by the wireless video transmission, a robust shape detector is needed.

Since the camera is located on the user's shoulder, the variation in the scale of the hand in the image is not expected to be very large, at least in the first frame of reference of the pointing gesture sequence. The detector uses the local shape descriptor presented in [MDTM04]. Given an image location, $r = 5$ rings with different radii centred at this location evaluate the skin classification value $\mathcal{C}$ of the image $\mathcal{I}(\cdot)$ at every π/K radians (in this experiment, $K = 32$), as shown in Figure 4.16. A positive value ($\mathcal{C}(\mathcal{I}(i,j)) = 1$) in the curve indicates skin, a negative value ($\mathcal{C}(\mathcal{I}(i,j)) = -1$) indicates background. For rotation invariance, the descriptor builds a feature vector $\mathbf{p}$ where each element consists of a similarity measure between each possible pair of response curves, i.e.,

$$\mathbf{p} = [h_{1,2}, h_{1,3}, h_{1,4}, h_{1,5}, h_{2,3}, h_{2,4}, h_{2,5}, h_{3,4}, h_{3,5}, h_{4,5}] \tag{4.11}$$

where $h_{m,n}$ is the similarity between the $2K$-dimensional curves m and n. Since the values of $\mathcal{C}(\mathcal{I}(\cdot))$ are either 1 or -1, $h_{m,n}$ is computed by

$$h_{m,n} = \frac{1}{2K}\sum_{k=1}^{2K} \mathcal{C}_{m,k}\mathcal{C}_{n,k} \; . \tag{4.12}$$

Note that $\mathbf{p}$ is invariant to rotation since it is a descriptor calculated with the shape itself, and invariant to column permutations.

A template $\overline{\mathbf{p}}$ is generated from a training image in which the user clicks on the metacarpophalangeal joint of the index finger and on the index finger tip[3]. This determines the centre and the orientation of

[3]For the nomenclature of hand bones and joints, see Figure 1.1.

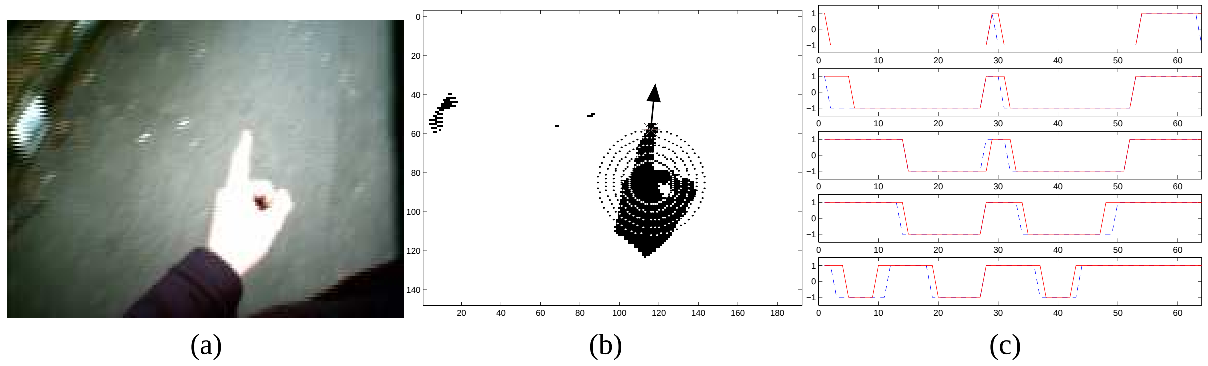


Figure 4.16: Extracting and matching the shape descriptor: (a) outdoor view of the hand; (b) the shape detector locates the pointing gesture and its direction; (c) values extracted from the rings, where the 1 indicates skin area and -1 indicates background; template values are showed by dashed blue lines and solid red lines show the current signal after best alignment. (From [MDTM04], with permission.)

the template, respectively. Upon application, a new sample vector $\mathbf{p}$ is compared with the template $\overline{\mathbf{p}}$ to determine the similarity $g(\mathbf{p}, \overline{\mathbf{p}})$ to the shape under search, determined by

$$g(\mathbf{p}, \overline{\mathbf{p}}) = \frac{1}{S} \sum_{i=1}^{S} p_i \overline{p}_i \ , \tag{4.13}$$

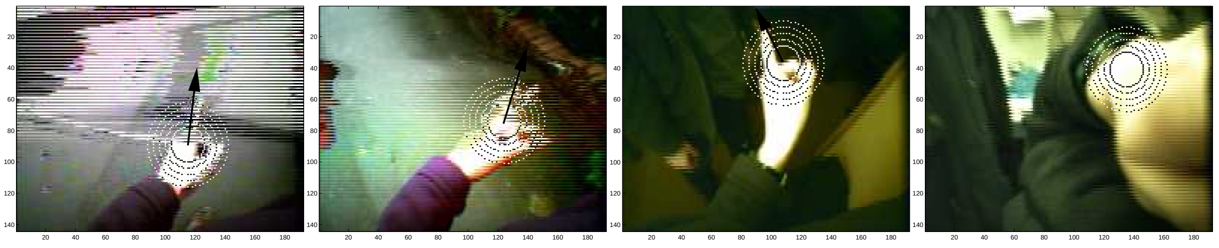
where $S = r!/2!(r-2)!$, where r is the number of rings used (here $S = 10$). Note that $g(\cdot) \in [0, 1]$. Thus, the detector is a function that tries to find the position of $\mathbf{p}'$ in the image $\mathcal{I}(\cdot)$, such that

$$\mathbf{p}' = arg \max_{\mathbf{p}} g(\mathbf{p}, \overline{\mathbf{p}}) \ . \tag{4.14}$$

The spacing between rings and the number of rings was determined experimentally. The best trade-off between accuracy and computational power for 192×144 images was obtained using 5 rings spaced from each other by 4 pixels. The innermost ring has radius of 11 pixels.

In order to speed up the detector, a coarse-to-fine search method was used. In the first stage, a gross search is done and the similarity is evaluated only once in each 27 pixels in the vertical and horizontal directions. Next, a fine search is done centred on all skin colour pixels in the neighbourhood of the best location found in the gross search. Once the position that maximises $g(\mathbf{p}, \overline{\mathbf{p}})$ is found, the hand orientation is estimated by searching for the orientation θ of the template $\overline{\mathbf{p}}$ that maximises the similarities $h_{\overline{m},m'}$ between the rings of the template and the located image descriptor.

To save computational time, the matching score $g(\mathbf{p}, \overline{\mathbf{p}})$ is evaluated before moving to a finer stage. If it falls below a threshold, it is considered that no pointing hand has been located in the image and the system waits for the next frame. The same happens after the finest search in order to decide whether to move to the tracking stage or not. Figure 4.17 shows that the detector functions under quite different and severe image noise.



(a) $g(\mathbf{p}, \overline{\mathbf{p}}) = 0.86$ (b) $g(\mathbf{p}, \overline{\mathbf{p}}) = 0.85$ (c) $g(\mathbf{p}, \overline{\mathbf{p}}) = 0.81$ (d) $g(\mathbf{p}, \overline{\mathbf{p}}) = 0.72$

Figure 4.17: Challenging images and detector response values: (a) outdoor noise image where hand is non-contiguous (finger striped), (b) ghostly finger; (c) indoor image with change in shape (sleeve retracted); (d) Non gesturing hand. The video noise in (a) and (b) is encountered at the limits of the wireless transmitter's range.

4.4.4 Hand tracking

The shape detector initialises three degrees of translational and rotational freedom that most affect image appearance. The other 3 DOF are set to default values, and all are passed to an implementation of RAPiD. The idea is that the user tells the robot that (s)he is performing a pointing gesture by starting with the hand at a roughly standard distance from the camera. Next the user can adjust the depth of the pointing direction and this is identified by the tracker.

Since the aim here is to track a single pointing gesture, a rigid model of the hand is enough. In order to reduce the computational cost, a simple planar model was used, so self-occlusion handling is not necessary. This model comprises straight edges along which control points $\mathbf{X}^0$ are distributed in the model coordinate frame. Since this is a monocular system, the world coordinate frame can coincide with the camera coordinate frame, as shown in Figure 4.18.

Since the images are binarised on skin colour, finding edges is trivial. But as the finger is narrow, some care has to be taken not merely to chose the edge closest to the control point. In Figure 4.19, for example, this would be a mismatch. The edge detector restricts the direction of the edge to be dependent on the searching direction. The hand model used here is a polygon such that all the lines may lie in between hand and background pixels. Therefore, considering the clockwise direction, the search

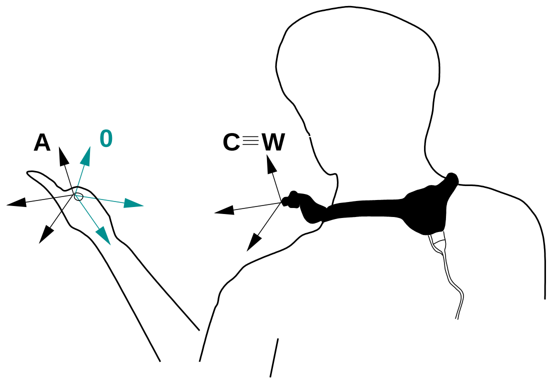


Figure 4.18: The hand and camera coordinate frames.

is performed from right to left. The first value change from 1 (skin) to 0 (background) is taken as the located edge. This also prevents the tracker fitting to background edges.

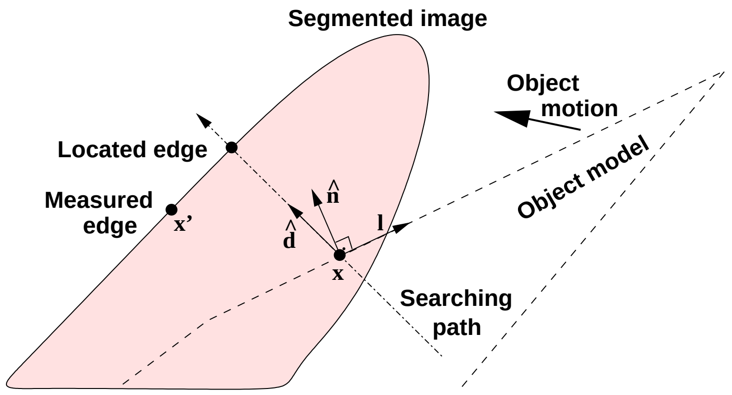


Figure 4.19: The search is made similarly to Figure 4.2, but here the search is directional and only considers skin-to-background edges.

The size of path for edge searching $2L$ is set to a value that is proportional to the proximity between the hand and the camera, because the speed of the hand in the image is likely to be proportional to this proximity. Thus, $L = K/t_Z^{i-1}{}_{0C}$, where $t_Z^{i-1}{}_{0C}$ is the distance between the camera and the hand in the previous frame of the video sequence. The constant K is set to $K = 5t_Z^{i=0}{}_{BC}$, where $t_Z^{i=0}{}_{BC}$ is the default translation in depth that is used in the first iteration of the tracker after the detector is executed.

Figure 4.20 shows a skin colour segmented image overlapped by a projection of the five-line planar hand model showing the control points. Although the model used does not have a realistic appearance, the experiments have shown that modelling the finger as a triangle increases the motion constraints along the finger axis. This also makes it more robust to rotations in depth. Such additional constraints compensates the lack of edges on the wrist, which were not included to avoid requiring that the user wears a long-sleeved shirt or a bracelet. The simplicity of this model speeds up projection calculations.

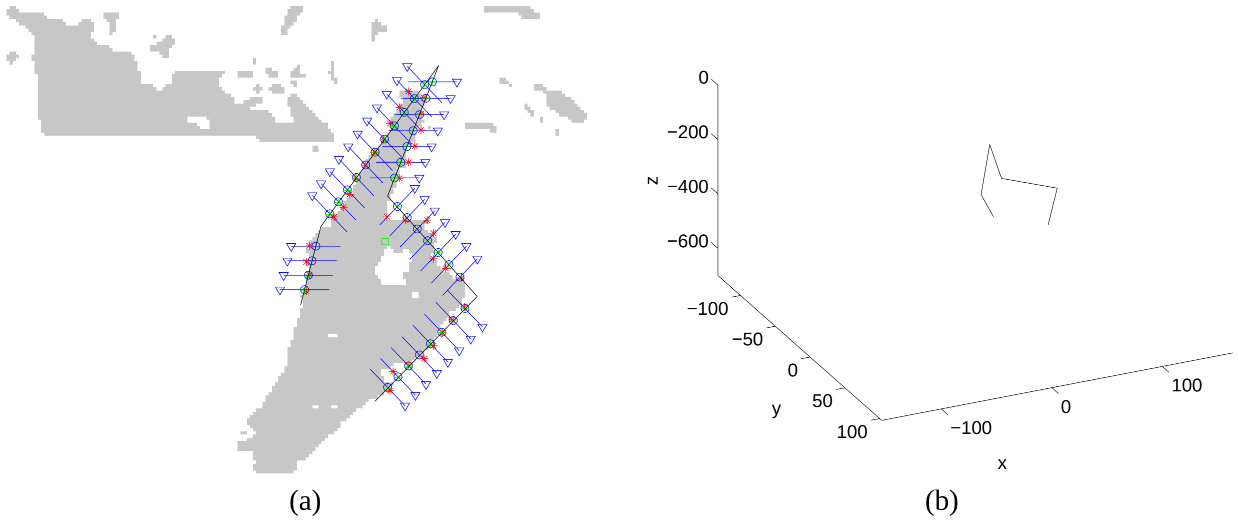


Figure 4.20: (a) Projection of the hand model (black line), search paths (segments with a triangle indicating the end of the search), control points (circles) and located edges (‘*’). (b) Representation in the camera coordinate frame of the hand model projected in the image in (a). The units are in millimetres and the Z axis is the camera axis.

4.4.5 Monitoring tracking

To monitor the tracking performance, the norm of RAPiD’s residual vector $||\mathbf{d}||$ (before pose update) could be used. But outliers and unmatched control points are not included in the residual which this means that the value of $||\mathbf{d}||$ does not reflect the success of the tracker. Thus, the choice was made for a cost function that depends on the actual distance between the located edges $\mathbf{r}$ and the projected lines $\mathbf{l}$ of the model after the pose update. Using homogeneous coordinates, each point can be modelled as a vector $\mathbf{x} = (x, y, 1)^\top$, and the lines are defined by $\mathbf{l} = \mathbf{x}_m{}' \times \mathbf{x}_n{}'$, which is equivalent to the following determinant:

$$\mathbf{l} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ x'_m & y'_m & 1 \\ x'_n & y'_n & 1 \end{vmatrix}, \tag{4.15}$$

where $\mathbf{x}_m{}'$ and $\mathbf{x}_n{}'$ are two points that lie in $\mathbf{l}$. The distance $m_{p,q}$ between line $\mathbf{l}_p$ and point $\mathbf{r}_q$ can be computed by:

$$m_{p,q} = \frac{\mathbf{r}_q^\top \mathbf{l}_p}{\sqrt{l_{x_p}^2 + l_{y_p}^2}} \tag{4.16}$$

The cost function is then defined by the sum of all the distances m between all the found edges and their respective lines:

$$\mathcal{C} = \frac{1}{\mathcal{W}D} \sum_{\forall p,q} m_{p,q}, \tag{4.17}$$

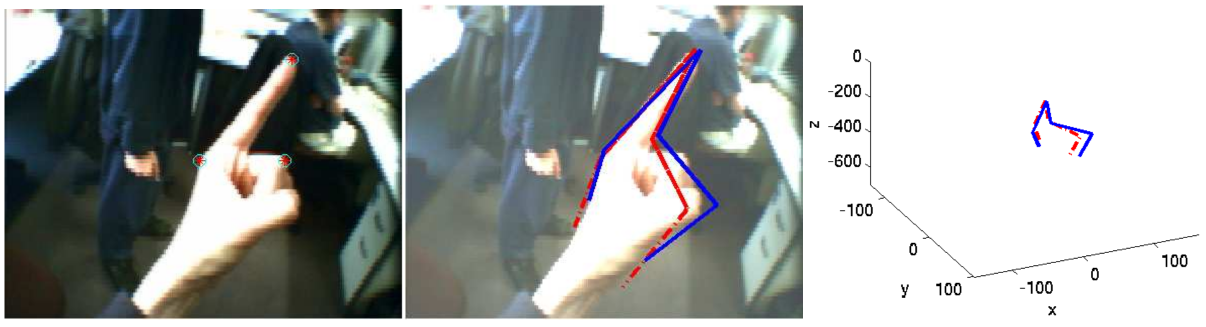
where D is the total number of control points in the whole model, and $\mathcal{W}$ is the worst case constant, defined by $\mathcal{W} = 2L$, which is the number of pixels in the path for searching edges. When no edge $\mathbf{r}_p$ is located in the search path for a control point, $m_{p,q}$ is set to $\mathcal{W}$.

The cost function result is employed to determine if the tracker has lost the hand and the detector needs to be called. The function is also used to verify if the tracking results are good enough to be used to perform a camera movement toward the target. A second condition for that is the stability of the hand in the space. If the change of pose $||\mathbf{s}||$ is below a given threshold for 1 second, the camera can be redirected to the target direction.

4.4.6 Results

The experiments described here were performed on a video sequence of 1104 frames grabbed from the wearable camera in an office environment with no illumination control and with a cluttered background. An approximate of the ground truth trajectory was generated from mouse clicks on three points of the hand: on the index finger tip, on the index finger MCP knuckle, and on the middle finger PIP joint knuckle. The Nelder-Mead simplex algorithm [NM65] was used to minimise the geometric error [HZ01] between mouse clicks and pose hypotheses to estimate ground truth. Figure 4.21 shows a sample image with the mouse clicks used to estimate the ground truth data and the pose estimation results in comparison to tracking results.

The plots in figure 4.22 show the pose estimation results (thick blue curves) with time (in frames) in comparison with the ground true estimative (thin red curves) for four degrees of freedom. When the cost function indicated a bad pose estimate, the hand detector was invoked. The circles illustrate the frames where this happened. Cost function results are shown in Figure 4.23.



(a) (b) (c)

Figure 4.21: Ground truth data estimation and tracking result. (a) Original image with the points used to estimate ground date; (b) Projection of the model with pose parameters obtained with the ground truth data and with RAPiD. (c) Model in the 3D pose estimated by the "ground truth" data and by tracking method. In both (b) and (c), solid blue indicates RAPiD result and dashed red indicates ground truth estimate.

These results show that the estimates of parameters parallel to the image plane (X, Y and θ_z) are good match to the "ground truth" data, but the same is not observed for the depth parameters (e.g. Z). However the estimate of ground truth data was not reliable for depth parameters because only three mouse-clicked points in a single view were used, without sub-pixel accuracy. It was difficult to choose more points to be clicked, as the hand texture is plain. A better estimate of the ground truth would be obtained if multiple views were available for the same sequence. The above can be verified in the video sequence that demonstrate the results.

The same video sequence was used to evaluate the application for redirecting the wearable camera to an object of interest. The results are plotted in Figure 4.24, which, for clarity, shows only the estimated pose and the "ground truth" in the frames where the re-direction process was called. The wearable camera's movement and location of object of interest is assumed to take 1s, after which the wearable camera moves back to the hand detection context.

4.5 Summary and conclusion

This chapter described the RAPiD rigid object tracker and its implementation for multiple view tracking, using the notation of [TRMM01]. Occlusion handling and dynamic generation of control points was discussed for objects made of simple geometric primitives like planes, lines, circles, spheres and truncated cones. Evaluations with synthetic and real images have validated the implementation as a real-

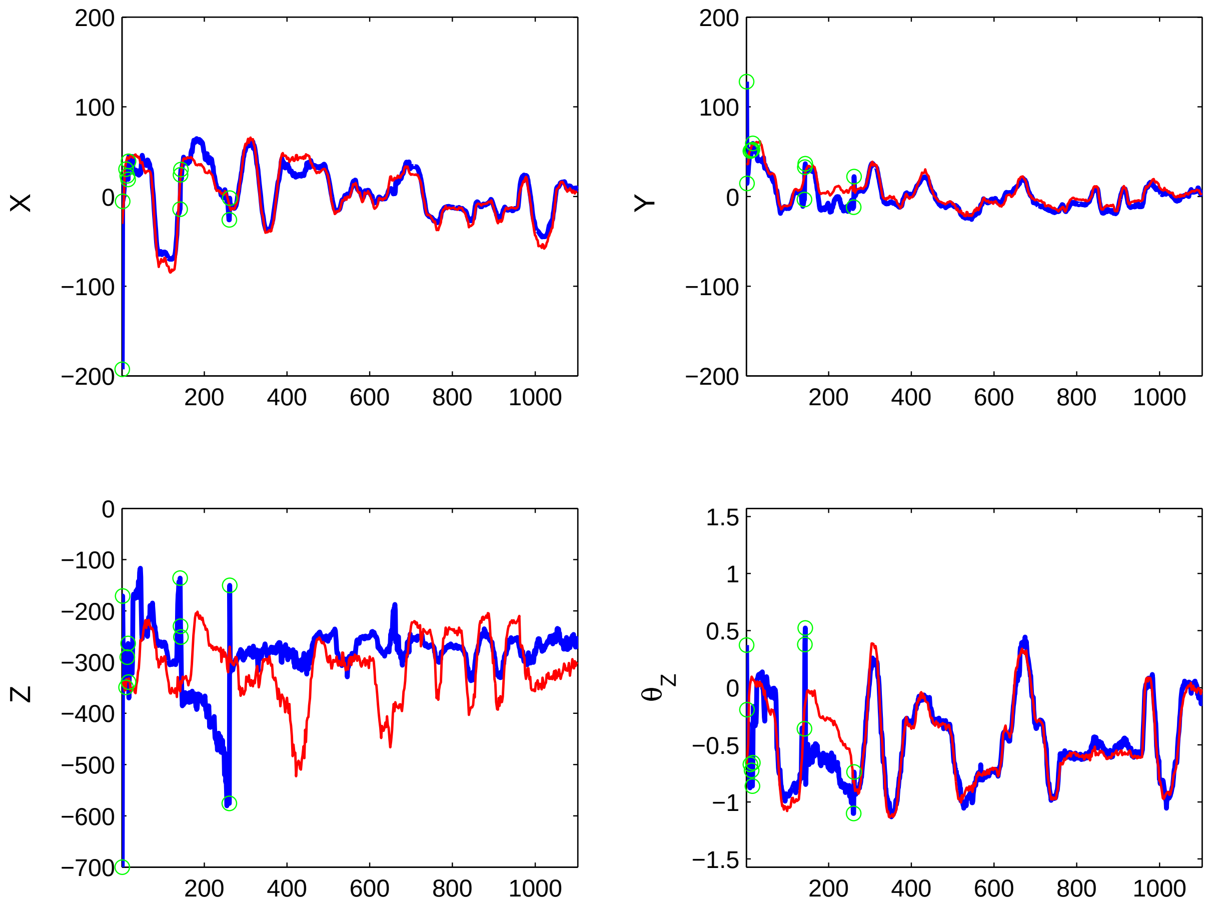


Figure 4.22: Results of the integrated system (thick blue curves) showing the detector calls (circles) and the ground truth estimate (thin red curves). The space is measured in millimetres, angle in radians and the time in frames. The estimated values of X, Y, Z and θ_Z are plotted against time, in frames.

time tracking method. The merits of this method for real-time multiple views implementation are the main point of interest and for this reason this method was chosen as the basis for the development of an articulated tracking system, described in next chapter.

An application of RAPiD and the skin colour detection method described in the previous chapter was described. This is method for detecting and tracking a specific hand shape — pointing — with applications of estimating the focus of attention or controlling the gaze direction of a wearable active camera. This enhances user-robot interaction and enables the recognition of an important non-verbal communication gesture.

This method combined a 2D shape detector and the RAPiD 3D tracker using a finite state machine. Criterion functions for both the detector and the tracker were used to automatically monitor their result

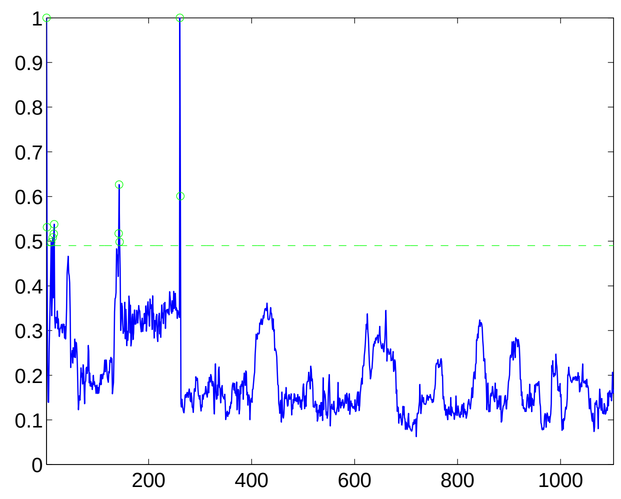


Figure 4.23: Cost function results for the experiment shown in Figure 4.22 plotted against time (in frames). The dashed line is the threshold used to indicate whether the tracker is lost, and the circles indicate when the hand detector was called.

in order to change the state in the finite state machine. The detection method provides an initial estimate of the planar pose parameters which are then refined with 3D information by RAPiD.

The experiments have shown that a simple rigid planar model of the hand lead to acceptable tracking results with low computational cost.

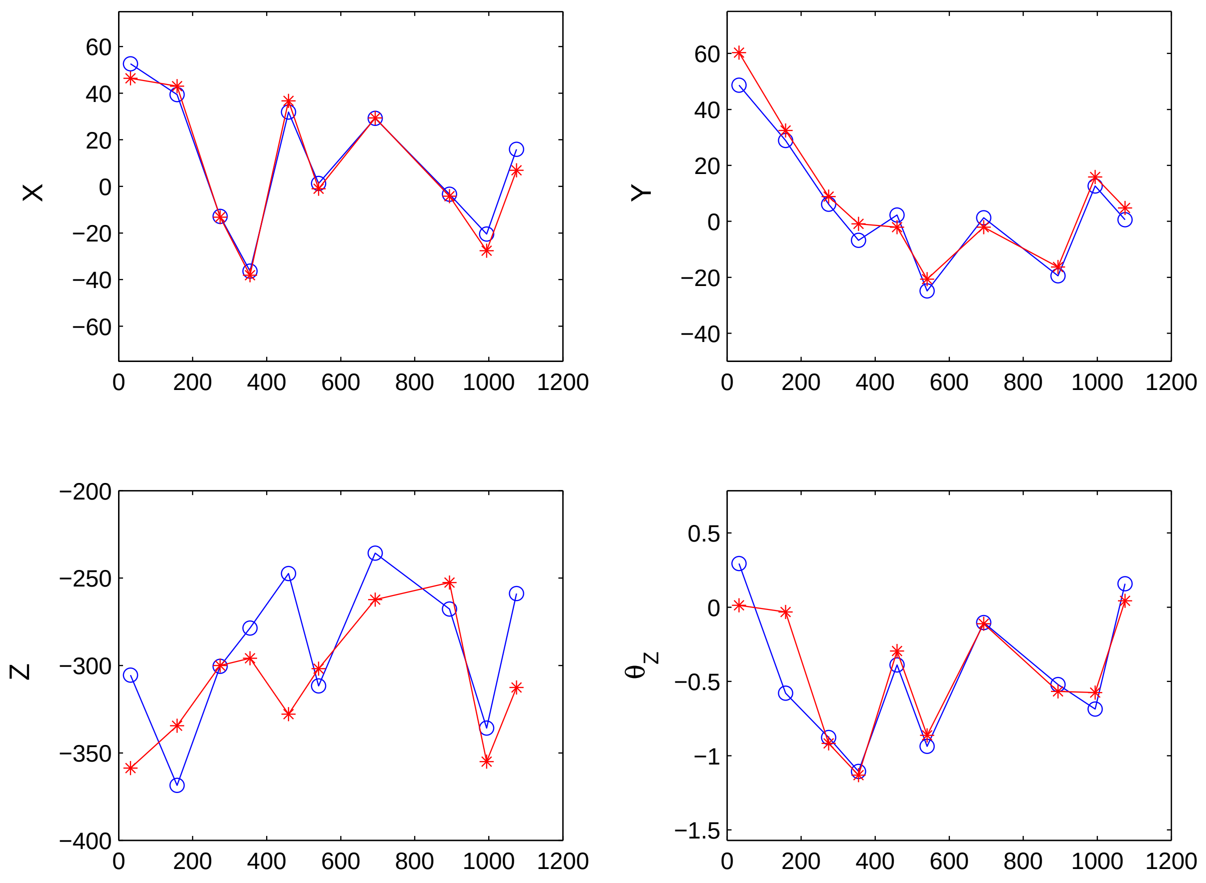


Figure 4.24: Pose estimations ('*') and estimated ground truth ('o') when the re-directing process was called. The estimated values of X, Y, Z and θ_Z are plotted against time, in frames.

5

An articulated RAPiD tracker: ART

This chapter describes a novel extension of Harris' RAPiD rigid object tracker to track articulated objects in 3D. It generates a linear system for pose update in terms of a minimal set of variables. A subpart of the object is chosen as its basis with six degrees of freedom and the pose of the remaining subparts are described in terms of the joints connecting from this basis in a kinematic tree. Experimental demonstrations of this system are given in a video-rate implementation using imagery from multiple cameras.

5.1 Introduction

With the aim of performing full-DOF tracking of hands, this chapter develops a method to track generic articulated objects in real-time. Hands can be modelled as kinematic chains, which are assemblage of links and joints. In robotics, *forward kinematics* is the process of calculating the position in space of the end of a linked structure given the angles of all the joints. This process is based on performing transformations from the basis coordinate frame to the end of the chain. This is what is normally done to update the pose of an articulated object when a new pose vector is available. In this straightforward process, only one solution, i.e., position of the end of the chain, is obtained (for acyclic kinematic systems).

Inverse kinematics does the reverse: given the end point of the structure, the goal is to find the joints angles necessary to reach it [Cra89]. In general, this is solved by an optimisation process that locates the parameters that minimise the distance between the target endpoint and its current position. Depending on the kinematic chain and on the position of the target point, there might be zero, one or multiple solutions.

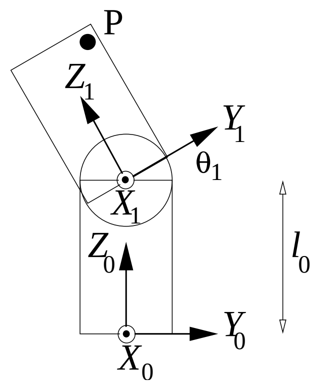


Figure 5.1: A simple articulated object to illustrate the basics of ART.

Section 2.3.3 presented an overview of inverse kinematics methods applied to 3D hand tracking using accurate measurements of fingertip locations obtained, for instance, colour markers. The problem becomes more challenging for marker-less tracking of articulated objects, as reviewed in Section 2.3.4.

Based on the success of the RAPiD tracker for real-time tracking of rigid objects, this chapter describes an extension of this method for articulated objects, dubbed ART. Section 5.2 develops this extension with a simple two parts example and extrapolates to complete chains. Section 5.3 describes the algorithm used to implement this method for kinematic trees and discusses the case of closed loop kinematic graphs. Section 5.4 evaluates some methods to solve linear systems in order to compute the pose update. Details of a hand model and occlusion handling are presented in Section 5.5. Experiments are described in Section 5.6 and the chapter ends with a summary in Section 5.7.

5.2 Extending RAPiD to articulated objects

Consider two subparts labelled 0 and 1, connected by a pure revolute joint with joint angle θ_1 located at $\boldsymbol{\ell}_0$ in subpart 0's frame, as shown in Fig 5.1. A point P at $\mathbf{X}^1$ referred to the local frame attached to subpart 1 of the articulated mechanism is at

$$\mathbf{X}^0 = \mathtt{T}_1^0\mathbf{X}^1 = \begin{pmatrix} \mathtt{R}(\theta_1) & \boldsymbol{\ell}_0 \\ \mathbf{0}^\top & 1 \end{pmatrix}\mathbf{X}^1 \tag{5.1}$$

in part 0's frame. For generic rotations, the matrix $\mathtt{R}$ is composed from the angle θ and axis $\bar{\boldsymbol{\alpha}}$ using Rodrigues' formula [Cra89]:

$$\mathtt{R} = \mathtt{I}_{3\times3} + \sin\theta[\bar{\boldsymbol{\alpha}}]_\times + (\cos\theta - 1)(\mathtt{I}_{3\times3} - \bar{\boldsymbol{\alpha}}\bar{\boldsymbol{\alpha}}^\top)\,, \tag{5.2}$$

where $[\bar{\boldsymbol{\alpha}}]_\times$ is the cross product matrix formed from $\bar{\boldsymbol{\alpha}}$.

As point P is stationary in frame 1, differentiation with respect to time gives

$$\dot{\mathbf{X}}^0 = \dot{\theta}_1 \begin{pmatrix} \mathtt{R}'(\theta_1) & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{pmatrix} \mathbf{X}^1 = \dot{\theta}_1 \mathtt{U}_1^0 \mathbf{X}^1 , \tag{5.3}$$

where $\mathtt{R}'$ is the element-by-element derivative of $\mathtt{R}$ with respect to θ_1. For a generic rotation matrix, $\mathtt{R}'$ is computed by differentiating (5.2):

$$\mathtt{R}' = \cos\theta [\bar{\boldsymbol{\alpha}}]_\times - \sin\theta (\mathtt{I}_{3\times 3} - \bar{\boldsymbol{\alpha}}\bar{\boldsymbol{\alpha}}^\top) \tag{5.4}$$

Similarly for purely prismatic joints

$$\mathtt{T}_1^0 = \begin{pmatrix} \mathtt{I}_3 & \boldsymbol{\ell}_0 + \theta_1 \hat{\mathbf{u}}_1 \\ \mathbf{0}^\top & 1 \end{pmatrix}, \ \mathtt{U}_1^0 = \begin{pmatrix} \mathtt{O}_3 & \hat{\mathbf{u}}_1 \\ \mathbf{0}^\top & 0 \end{pmatrix} \tag{5.5}$$

where $\hat{\mathbf{u}}_1$ is a unit vector and parameter θ_1 is now a length, not an angle.

This is straightforwardly extended to a point on subpart J of a mechanism

$$\mathbf{X}^0 = \mathtt{T}_1^0(\theta_1)\mathtt{T}_2^1(\theta_2)...\mathtt{T}_J^{J-1}(\theta_J)\mathbf{X}^J , \tag{5.6}$$

$$\dot{\mathbf{X}}^0 = \left(\dot{\theta}_1 \mathtt{D}_{J1}^0 + \dot{\theta}_2 \mathtt{D}_{J2}^0 + \ldots + \dot{\theta}_J \mathtt{D}_{JJ}^0\right) \mathbf{X}^J , \tag{5.7}$$

where $\mathtt{D}_{J1}^0 = \mathtt{U}_1^0\mathtt{T}_2^1...\mathtt{T}_J^{J-1}$; $\mathtt{D}_{J2}^0 = \mathtt{T}_1^0\mathtt{U}_2^1...\mathtt{T}_J^{J-1}$; and so on. For a mechanism with $(N+1)$ parts, this expression can be written as a linear sum over all N joint velocities $\dot{\boldsymbol{\theta}} = (\dot{\theta}_1 \ldots \dot{\theta}_N)^\top$

$$\dot{\mathbf{X}}^0 = \left(\mathbf{a}_1 | \mathbf{a}_2 | \ldots | \mathbf{a}_J | \mathtt{O}_{4\times(N-J)}\right) \dot{\boldsymbol{\theta}} = \mathtt{A}_{4\times N} \dot{\boldsymbol{\theta}} , \tag{5.8}$$

where $\mathbf{a}_j = \mathtt{D}_{Jj}^0 \mathbf{X}^J$. As $\dot{\mathbf{X}}^0$ is a direction vector, its fourth component is always zero. Below $\mathbf{a}$ is used as a 3-vector, and a $(3\times N)$ matrix $\mathtt{A}$ is written as $\mathtt{A}_{3\times N} = (\mathtt{I}_3 | \mathbf{0})\, \mathtt{A}_{4\times N}$.

If the pose of base part (0) is given by $\{\mathtt{R}_0^W, \mathbf{t}_{0W}\}$ referred to the world frame, then the instantaneous velocity in the world frame is (returning to non-homogeneous coordinates)

$$\begin{aligned} \dot{\mathbf{X}}^W &= \mathtt{R}_0^W \dot{\mathbf{X}}^0 + \boldsymbol{\omega} \times \mathtt{R}_0^W \mathbf{X}^0 + \boldsymbol{v} \\ &= \mathtt{R}_0^W \mathtt{A}_{3\times N} \dot{\boldsymbol{\theta}} + \boldsymbol{\omega} \times \mathbf{X}^{\mathrm{A}} + \boldsymbol{v} , \end{aligned} \tag{5.9}$$

where $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are the instantaneous global velocity and angular motion as used in Chapter 4.

The key observation is that, just as in Eq. (4.2), the velocity in the world frame can be written linearly as

$$\dot{\mathbf{X}}^W = \mathtt{H}\mathbf{s} \tag{5.10}$$

but now $\mathtt{H}$ has N extra columns at the right

$$\mathtt{H} = \left(\; [-\mathbf{X}^A]_\times \;\middle|\; \mathtt{I}_{3\times 3} \;\middle|\; \mathtt{R}_0^W \mathtt{A}_{3\times N} \; \right) \tag{5.11}$$

and $\mathbf{s}$ is augmented with the joint velocities

$$\mathbf{s} = \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \\ \dot{\boldsymbol{\theta}} \end{pmatrix} . \tag{5.12}$$

Like in RAPiD, edge features sought along near-orthogonal lines are used to reduce the cost of matching. Therefore the construction of the pose update equation from the measurements follows exactly as given in Eqs. (4.5) and (4.6), i.e., the motion parameters are estimated by solving

$$\mathtt{F}\mathbf{s} = \mathbf{d} \, , \tag{5.13}$$

with control points and measurements $\mathbf{d}$ are obtained from all the parts of the articulated body.

5.3 ART algorithm

The articulated RAPiD tracker has been implemented as a video-rate (30Hz) process for multiple articulated objects viewed by one or more cameras. The articulated object is represented as a graph (e.g. a tree) where each node stores data about the position and orientation of each joint, the type of the joints (revolute or prismatic) and the nodes that this node is connected to. Joints with more than one DOF are implicitly represented as a combination of 1 DOF joints. This can could lead to gimbal lock for rotations near $90°$, but this has not been a problem in our experiments because the tracked objects (specially the hand) do not move to this range in joints with more than 1 DOF. Each node has a representation of a rigid object which is the same as in RAPiD. Algorithm 1 summarises the tracking method, and certain of its steps are fleshed out below.

For kinematic chains with branches (i.e. kinematic trees), the coupling of subparts with joints can be represented as a tree. To gather the information to complete each pose update, the tree is explored depth

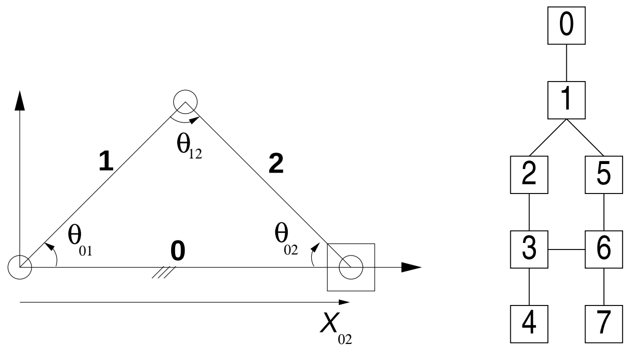
Algorithm 1 Articulated RAPiD tracker (ART) for kinematic trees – one iteration of the pose update.

```
1:  At the base part set cumulative transformation $\mathtt{T}^0_0 = \mathtt{I}_4$.
2:  for each subpart $j$ in a depth-first expansion do
3:      Set $p$ to be $j$'s parent
4:      Compute and store cumulative $\mathtt{T}^0_j = \mathtt{T}^0_p \mathtt{T}^p_j$
5:      Compute and store $\mathtt{D}^0_{jj} = \mathtt{T}^0_p \mathtt{U}^p_j$
6:      for each joint $j' = j - 1$ back to $j' = 1$ do
7:          Compute and store $\mathtt{D}^0_{jj'} = \mathtt{D}^0_{pj'} \mathtt{T}^p_j$
8:      end for
9:      for each camera do
10:         for each visible control point $i$ on subpart $j$ do
11:             Construct $\mathtt{A}_i$, $\mathtt{H}_i$ and thence $\mathbf{f}_i$
12:             Search for image edge, compute $d_i$.
13:             Append row $\mathbf{f}_i$ to matrix $\mathtt{F}$, and $d_i$ to vector $\mathbf{d}$
14:         end for
15:     end for
16: end for
17: Derive $\mathrm{s}$ from $\mathtt{F}\mathrm{s} = \mathbf{d}$ (Eq. 5.13).
18: Update pose and joint angles
```

first. This reduces the amount of computation required to calculate the coordinate frame transformations. Denoting the root and current subparts as nodes 0 and j, respectively, and the parent of the current node as p, the cumulative transformation (Eq. 5.6) at the current node is found as $\mathtt{T}^0_j = \mathtt{T}^0_p \mathtt{T}^p_j$ and stored at the node. To populate Eq. (5.7), the dependency on the joint angle (or length) between parent and current nodes is determined as $\mathtt{D}^0_{jj} = \mathtt{T}^0_p \mathtt{U}^p_j$, and the dependencies on joint angles (or lengths) $\theta_{j'}$ earlier than the current node's parent found as $\mathtt{D}^0_{jj'} = \mathtt{D}^0_{pj'} \mathtt{T}^p_j$. The $\mathtt{D}$ matrices are again stored at the node.

Further computational saving is made on computing the matrix $\mathtt{A}$ (5.8) when there is a prismatic joint. As shown in Equation (5.5), with exception of the first three elements of the rightmost column of $\mathtt{U}^{j-1}_j$, all other elements are zero. This means that $\mathtt{U}^{j-1}_j \mathtt{T}^j_{j+1} \mathtt{T}^{j+1}_{j+2} \cdots \mathtt{T}^{J-1}_J = \mathtt{U}^{j-1}_j$, so there is no need to compute $\mathtt{T}^k_{k+1}$ for k further down the kinematic chain in the calculation of $\mathtt{D}^0_{Jj}$.

Kinematic chains with closed loops are represented following a standard solution for inverse kinematics. In [GA90], the pose change of the end effector (or of any chosen link in or after the loop) is represented following two paths from the base, i.e., using two Jacobian matrices $\mathtt{A}$ and $\mathtt{B}$. Given the two sets of joint parameters $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_b$, the inverse kinematics solution is obtained by making $\mathtt{A}\dot{\boldsymbol{\theta}}_a - \mathtt{B}\dot{\boldsymbol{\theta}}_b = \mathbf{0}$. In the example of Figure 5.2(a), $\boldsymbol{\theta}_a = [\theta_{01}, \theta_{12}, \theta_{02}]^\top$ and $\boldsymbol{\theta}_b = [X_{02}]$. For this simple example, $\mathtt{A}$ and $\mathtt{B}$ can be derived analytically (see e.g. [GA90]).



(a) (b)

Figure 5.2: Examples of closed loop kinematic chain: (a) the planar RRRP mechanism, where circles represent revolute joints and the square represents a prismatic joint. Link 0 is the basis and the position of the prismatic joint is the end effector. (b) A more complex kinematic chain with a loop, where each block represents a link and lines represent joints.

To apply the same idea using ART's representation, each control point from links inside a loop adds two rows **f** for the same measurement d, each **f** following a different path in the loop. For the example of Fig. 5.2(b), control points in the link 7 have two possible representation in the coordinate frame of link 0:

$$\mathbf{X}^0 = \mathtt{T}_1^0(\theta_{0,1})\mathtt{T}_2^1(\theta_{1,2})\mathtt{T}_3^2(\theta_{2,3})\mathtt{T}_6^3(\theta_{3,6})\mathtt{T}_7^6(\theta_{6,7})\mathbf{X}^7 \,, \tag{5.14}$$

$$= \mathtt{T}_1^0(\theta_{0,1})\mathtt{T}_5^1(\theta_{1,5})\mathtt{T}_6^5(\theta_{5,6})\mathtt{T}_7^6(\theta_{6,7})\mathbf{X}^7 \,. \tag{5.15}$$

Each measurement of links 2–7 adds two rows **f** to F and the system can be solved as before. The number of DOFs of the system of Eq. (5.13) remains the same, so the computational complexity of pose estimation is not affected by the presence of loops. Note that it is necessary to use weighted least squares to avoid erroneously increasing the importance of nodes in or below a loop. For generic chains (specially long or non-planar mechanisms), it is troublesome to determine the set of singular poses analytically, so it is necessary to do rank monitoring to solve Equation (5.13). Note that the flow control of Algorithm 1 needs to be modified in order to detect and deal with loop closing.

5.4 Solving the linear system

A critical step of the tracking method is to determine the pose update by solving the linear system of Equation 5.13, specially when robust pose is used. Although it is known that all methods to solve linear systems have asymptotic complexity of $\mathcal{O}(N^3)$, where N is the number of variables [PTVF88], different methods have different coefficients in the polynomials that express their time complexity. This can imply significant differences for the range of N that usually occurs for articulated objects tracking.

In ART, the number of unknowns is smaller than or equal to the number of equations. The robust pose calculations are done using minimal sets of measurements, so the number of equations and unknowns are always the same, which means that the system $\mathtt{F}\mathbf{s} = \mathbf{d}$ can be solved by

$$\mathbf{s} = \mathtt{F}^{-1}\mathbf{d} \tag{5.16}$$

For the final optimisation using all inliers, there are more measurements than degrees of freedom and $\mathtt{F}$ is a rectangular matrix. The system can then be solved using the Moore-Penrose pseudo-inverse of $\mathtt{F}$. That is,

$$\mathbf{s} = (\mathtt{F}^\top\mathtt{F})^{-1}\mathtt{F}^\top\mathbf{d}\ . \tag{5.17}$$

However, linear systems do not need to be solved necessarily as in (5.16) or (5.17). A matrix decomposition method that does not require full matrix inversion can be used. A usual way of solving these systems is by means of singular value decomposition (SVD), as it allows to diagnose how close to degenerate (or how close to singular) is the linear system [PTVF88]. But SVD is not the most efficient method to solve linear systems. If a singularity check is not performed, other matrix decomposition methods can be applied, providing faster computations. The processing time of the following methods have been evaluated: SVD, QR, Cholesky and Eigendecomposition (see [PTVF88] for details about these methods). Another point that was considered is that since $\mathtt{F}^\top\mathtt{F}$ is symmetric, a significant computational saving can be achieved by avoiding redundant multiplications to compute half of the matrix.

To evaluate our implementation of these variations, sections of test were written and the result shown in Figures 5.3, 5.4 and 5.5, the lines labelled as *optimised* show the results using a modified matrix multiplication method that avoids redundant computations. The lines labelled as *full* show the results obtained using the matrix decomposition method to solve the generic system (5.13). The lines labelled as *on FtF* are those that used (5.17) to solve the system by inverting the square matrix $\mathtt{F}^\top\mathtt{F}$. The graphs show the average time (in milliseconds) after 1000 repetitions of each experiment. For each matrix dimension, a linear system is created from random Real numbers and all the methods are applied to

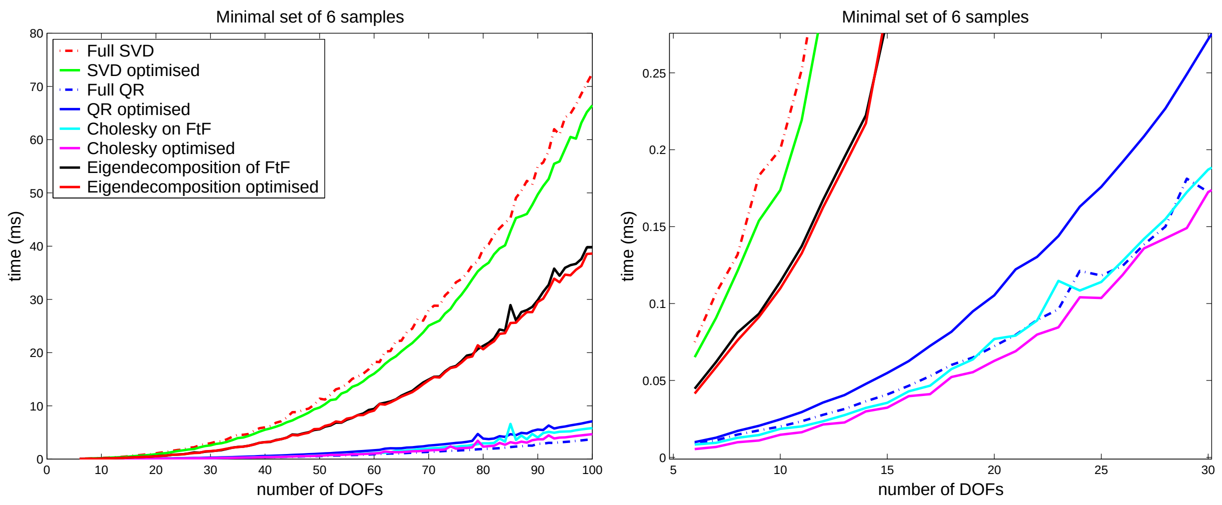


Figure 5.3: Results for a typical situation for robust pose: the number of unknowns (DOF) is equals the number of equations (measurements). **Left**: full graph showing all the methods; **right**: zoom on the 4 fastest methods for dimensions between 6 and 30 (more typical range for hand tracking).

solve the same system. These computations were performed on a 1.8GHz Pentium 4 machine. The numbers were represented using 64bits double precision. [1]

The results show that Cholesky decomposition (using the *optimised* $\mathtt{F}^\top\mathtt{F}$ calculation) provides the best results, specially when there are more equations than unknowns. The gap between the optimised and non-optimised versions is very large for Cholesky because this method assumes that the matrix is symmetric and simply does not check elements above the diagonal. So the upper part of the $\mathtt{F}^\top\mathtt{F}$ matrix is not even copied from the bottom part, saving time with memory access. Note that, in Figure 5.3 *Full QR* performed better than *QR optimised*. This is possibly because of the fact that if the matrix is already square, there is no benefit of using the *optimised* $\mathtt{F}^\top\mathtt{F}$ calculation for pseudo-inverse for QR. Apart from that, the *optimised* versions gave faster results than the other versions of the methods. Therefore, Cholesky decomposition with the *optimised* $\mathtt{F}^\top\mathtt{F}$ calculation was chosen as the standard for the tracking experiments.

If the precision of the results is not critical, a 32 bits implementation can be considered for machines with 32 bits processors (which is the case of the machine used in these experiments). By comparing the graphs of Figure 5.6 with those of Figure 5.5, one can note that, in some cases (e.g. *Full* SVD) the

[1]The graphs show some recurring quirks in processing time which are probably due to due to system's memory management issues, because the machine used was running as a single user Linux with the X interface switched off, so cpu time sharing was not an issue. But these quirks do not affect the comparative analysis.

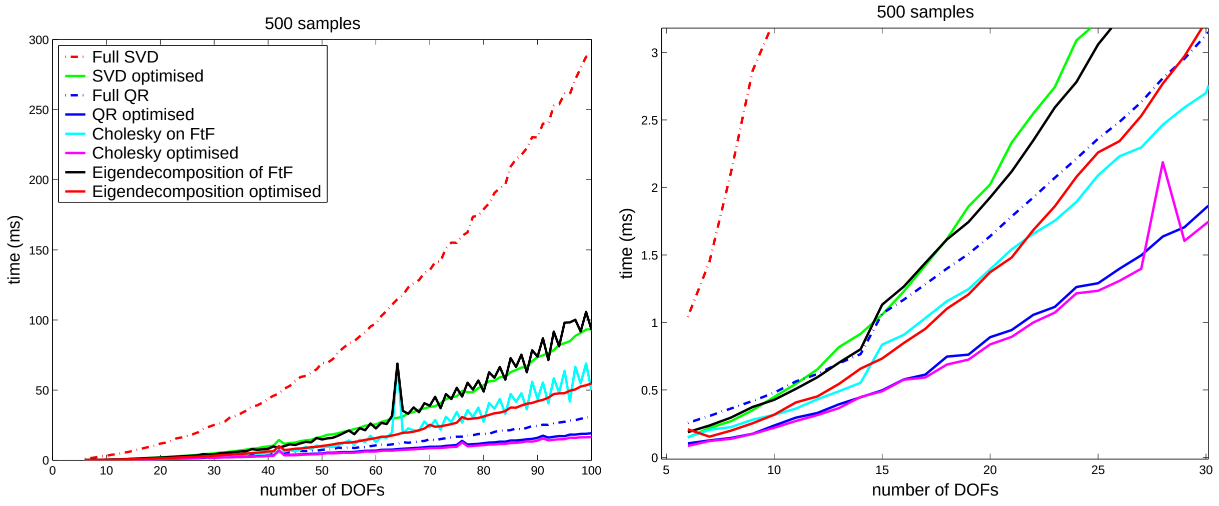


Figure 5.4: Results for 500 equations, an average situation for tracking with three cameras.

computation time drops down to around 50% of the time taken if double accuracy is used. The deviation among the linear system solutions obtained by different methods is $\leq 1.1 \times 10^{-7}$. Note that there are two issues with single precision number. The first is that accumulated roundoff errors in the solution process can swamp the true solution. The second is that for most ANSI C compilers, float variables are automatically converted to double before any operation is attempted. Therefore, for some compiler and library versions, the overhead of float to double (and vice-versa) conversions can make processing 32 bits variables take more time than processing 64 bits variables [PTVF88].

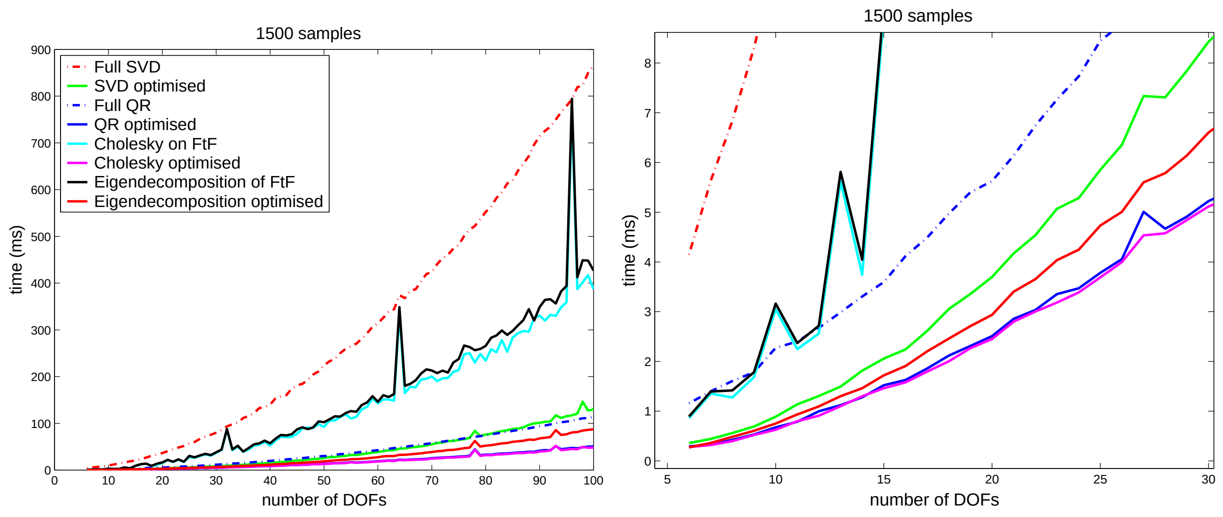


Figure 5.5: Results for a situation that occurs when the tracking accuracy is prioritised using a large number of measurements: 1500 equations.

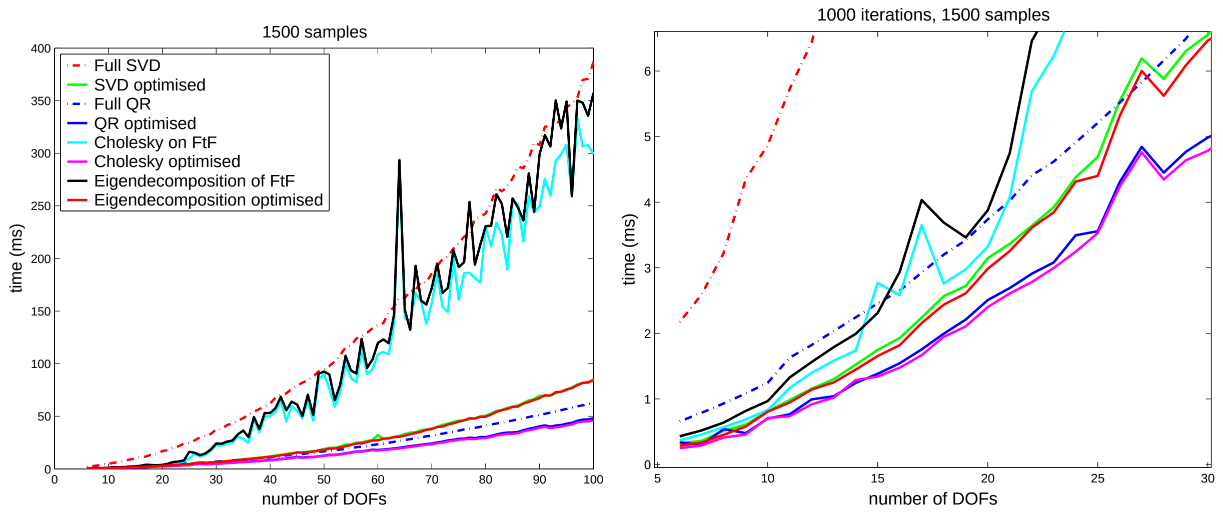


Figure 5.6: Same experiment of Figure 5.5 but using single precision float numbers (32 bits).

5.5 A 3D hand model

For the hand tracking experiments, a hand model was build including palm, thumb, fingers and, for some experiments, the forearm. The proportions of this hand model are based on measurements taken from a single subject, but as in the implementation of RAPiD, all the parameters are set in an XML-based text file, so they can be easily changed if measurements from other subjects are available. The model uses a combination of 20 truncated cones, 21 spheres and 6 planes, as shown in Figures 5.7 and 5.8.

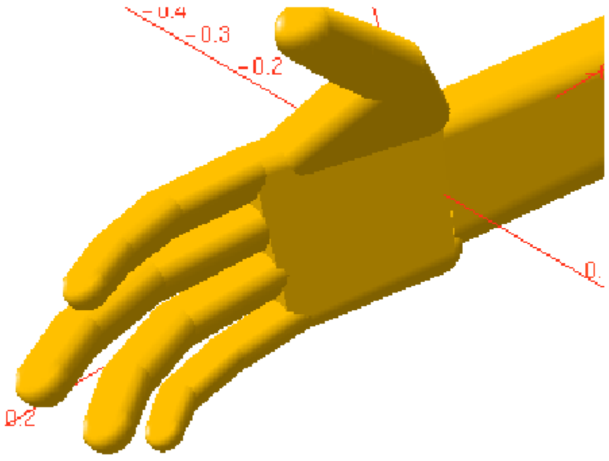
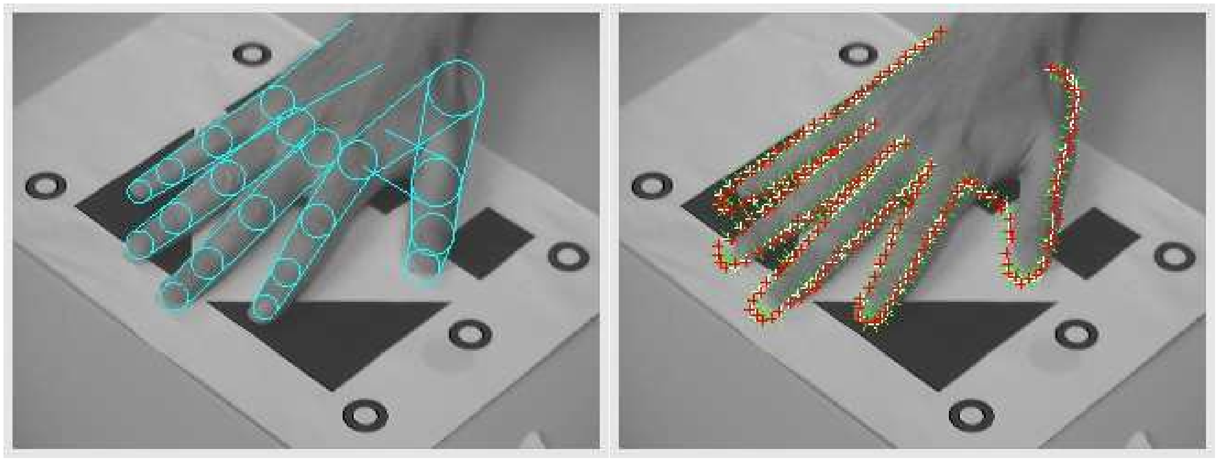


Figure 5.7: Hand model used in the tracking experiments. The model has 22 DOF of internal joint angles and 6 DOF of global pose parameters.



(a) (b)

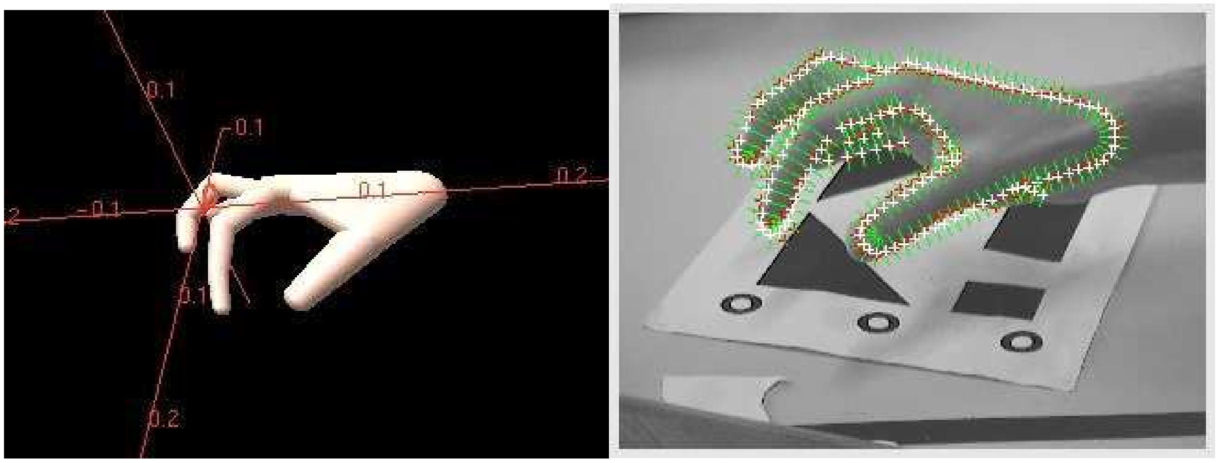
Figure 5.8: (a) Wire frame projection of the hand model on a hand image. (b) Projected control points to be used for pose update.

The palm is rigid, and each finger is modelled as a planar mechanism with 3 DOF for flexion and 1 DOF for abduction and adduction with the palm. The same model is used for the thumb, but its plane is not parallel to the fingers' planes. This gives a total of 20 internal DOF plus 2 DOF for the wrist, and 6 DOF of global pose parameters. Thus the hand motion vector $\mathbf{s}$ is 28 dimensional.

5.5.1 Occlusion handling

Self-occlusion handling is a critical task for complex articulated objects. Rehg and Kanade [RK95a] deal with it through a top-down approach using knowledge of the model to verify the registration of templates. A state space partitioned into regions of fixed visibility order of fingers is used. It is assumed that a finger can not occlude another and be occluded by it at the same time, which complies with the fact that fingers are modelled as planar kinematic chains and there are tight limits on adduction/abduction. A window function attached to each finger segment masks the contribution of segments that are occluded. This method saves computation, but it fails for poses such as "crossed fingers".

In ART, each control point is first checked against its own rigid body first, as described in Section 4.2.3. Next, the resulting visible control points are checked against other parts of the kinematic graph. Some computation is saved by not checking occlusion against object parts that are behind the control point. Figure 5.9 shows an example of hand pose with a significant amount of self-occlusion and the control points generated for that pose.



(a) (b)

Figure 5.9: Hand at a pose with significant amount of self-occlusion (a), and projected control points generated at that pose (b).

A few tens to few hundreds of control points contribute rows to the measurement system. When using multiple cameras, the measurement equations are gathered into a single system to solve for the pose, as in Chapter 4 and in [BM98, TRMM01].

5.6 Experiments

To demonstrate ART, output from video rate (30 Hz) experiments on increasingly complex objects is shown in Figure 5.11–5.15. The imagery was captured from three calibrated cameras viewing a 0.5×0.5 m^2 working area on a desk from near orthogonal directions as shown in Figure 3.6. Videos demonstrating these tracking results are available at http://www.robots.ox.ac.uk/$\sim$teo/art/

In the experiments of Figures 5.13 and 5.15, mismatching is reduced using robust methods. Least median of squares is used to remove non-collinear outliers amongst measured control points which should belong to the same line [TRMM01], and guided MLESAC [TM02] [TM05] is used to select control points in the robust generation of the solution to Eq. (5.13). To improve the confidence of the method, the implementation tries to select at least one control point per visible object part. To illustrate the importance of robust pose for tracking, Figure 5.10 shows a synthetic sequence in which tracking fails if no outlier rejection method is used and succeeds otherwise. The same figure also illustrates that if this articulated object is tracked as two single unconstrained objects, tracking fails.

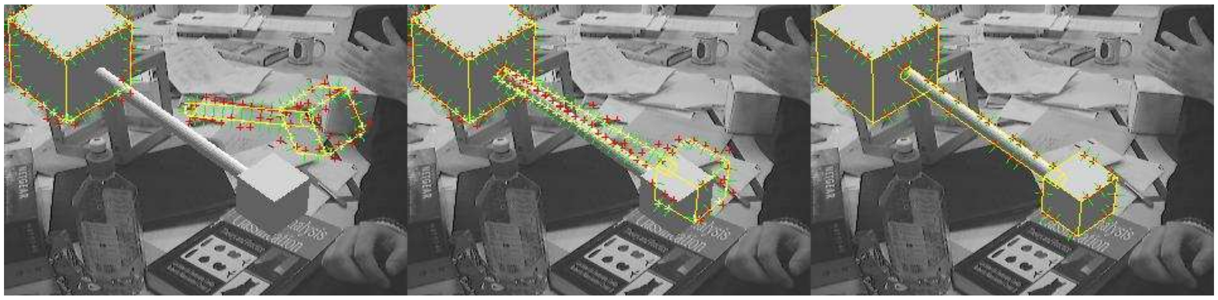


Figure 5.10: Tracking a synthetic articulated object made of two blocks linked by a 2 DOF revolute joint. In this sequence, the object rotates around an axis centred on the large block and the joint is kept static. The three panels show intermediate tracking results obtained by: tracking the two parts individually (left); tracking both parts as a single articulated object (centre); and also using robust pose (right).

Figure 5.11 shows three frames cut from a sequence where a book modelled with three planes (front, back and spine), two hinges, and eight DOF is tracked. The white crosses show the projection of the

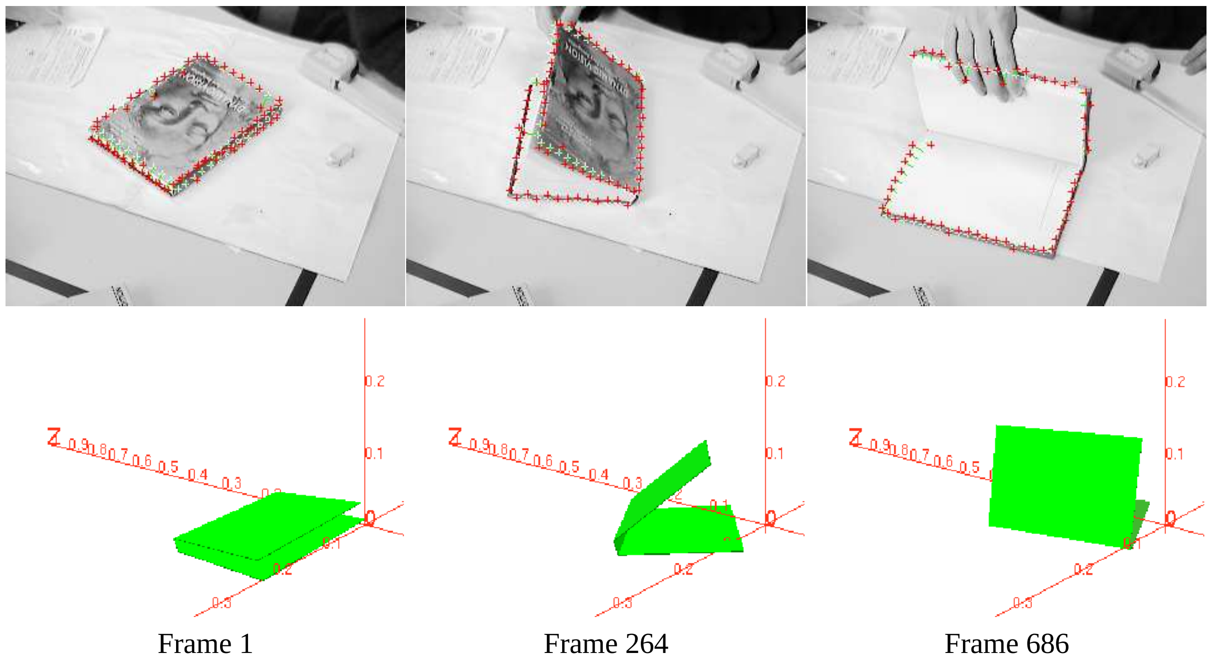


Figure 5.11: Tracking a book using an 8 degrees of freedom model in ART. The images (top) are all from the same camera of the three. The white crosses are the predicted control point positions, and the red are the corresponding measured image positions. The graphical output in the bottom shows the fitted pose in the world coordinate frame, generated from a viewpoint opposite to that of the camera of the top row.

model's control points in the image, and the red crosses show the located edges. Here, two iterations of the linear update were used per frame. Tracking continues in the presence of potential distraction caused by the presence of occlusions, texture in the object, and the proximity of model and image edges.

Although articulation has been described above in terms of single objects with subparts, the method can be more generally used to apply motion constraints between objects. For example, to track a ball while it rolls on a moving table, one can model the table as the base part, and the ball as connected to it using two 1-DOF prismatic joints. Figures 5.12 and 5.13 show more complicated examples. In Figure 5.12, two entities: a plane and an articulated object made of two cylinders are tracked. In Figure 5.13, four entities, a plane, a mug, a ball, and two articulated cylinders, are tracked. In both cases, the set is tracked as *one* articulated object, with 10 DOF and 14 DOF, respectively.

In Figure 5.12, the 3D pose of the basis object (the plane) has 6 DOF, the 2D pose of the articulated object with respect to the planar object is represented with 3 DOF, and 1 DOF represents the joint of the articulated object. In Figure 5.13, the 14 DOF comprises 6 DOF for the planar object, 2 each for ball and mug with respect to the plane, and 4 for jointed cylinders.

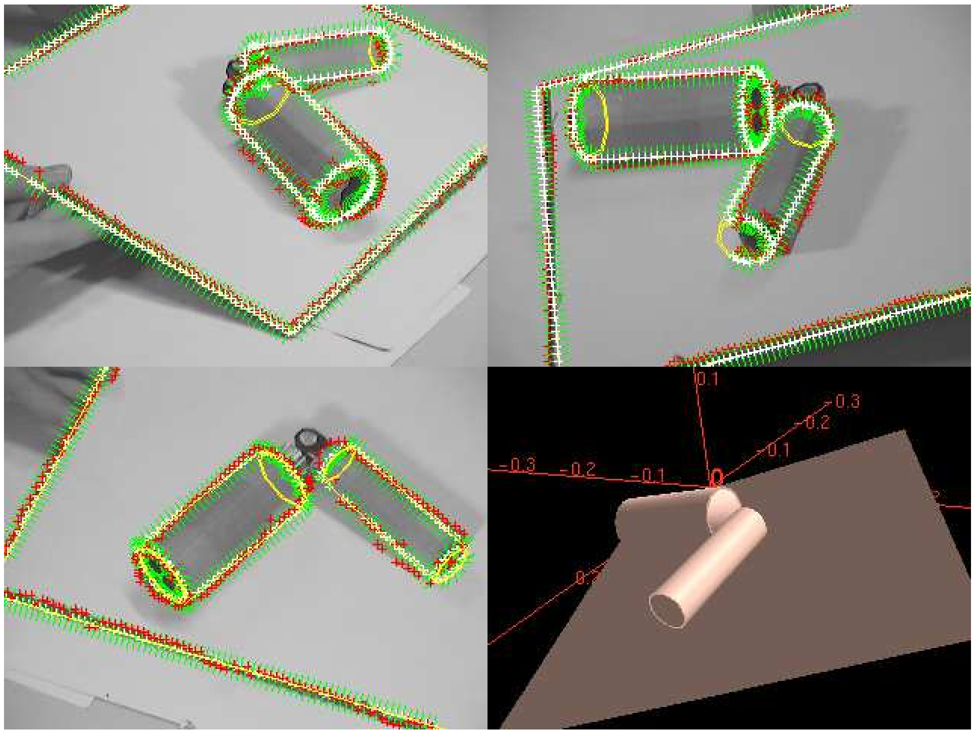


Figure 5.12: Two objects being tracked simultaneously: a planar object and an articulated object that rests on the planar object. The first three images show the views of the three cameras used with superimposed control points and the bottom right image is a 3D display showing the world coordinate frame and the object model in the pose that matches the acquired images.

Figures 5.14 and 5.15 show hand tracking experiments. In the former, the hand is kept flat and performs abduction movements. The latter figure shows a sequence of tracking a hand and a box, the hand with a single articulation for the fingers. The hand and the box are modelled as a single 7 DOF kinematic tree (3 for the hand, plus 1 for the fingers and 3 for the box with respect to the hand), constrained to the table plane. When the hand grasps the box, the degrees of freedom between hand and box are switched off and the whole set is tracked as a rigid object, reducing the dimensionality of the problem. To switch off DOFs, elements of $\mathbf{s}$ and columns of $\mathtt{F}$ are removed from linear system $\mathtt{F}\mathbf{s} = \mathbf{d}$ before computing $\mathbf{s}$.

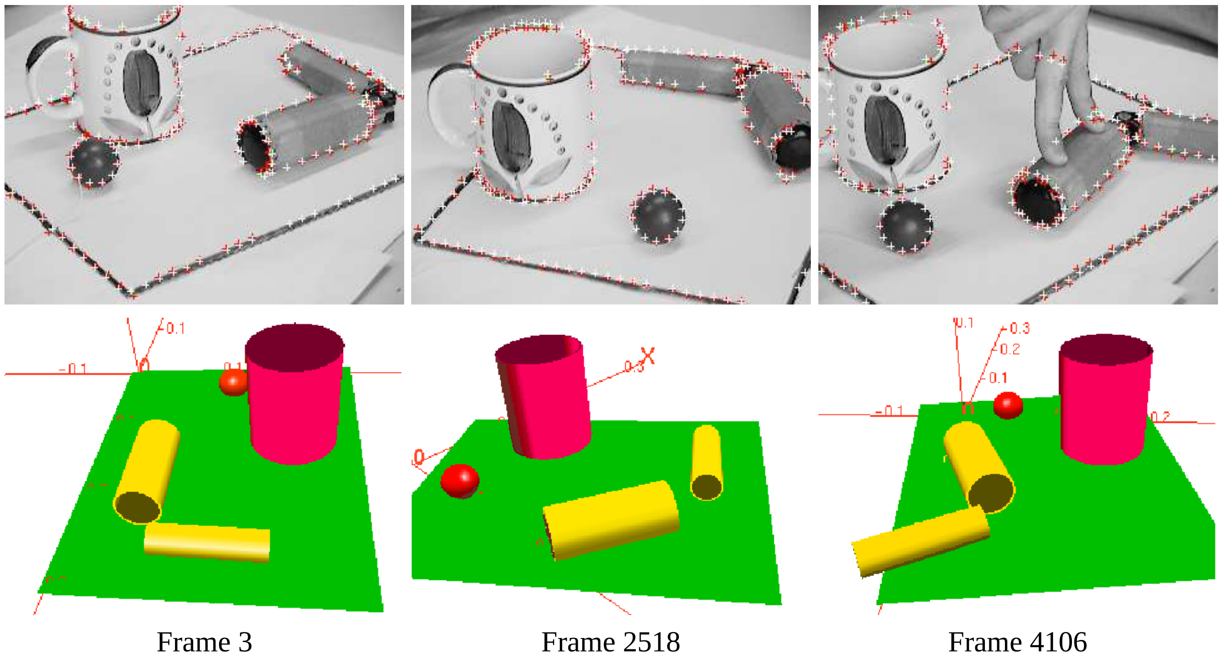


Figure 5.13: Tracking four objects as a 14-DOF articulated, or "motion-constrained", object. Also shown are graphical views generated from the corresponding contemporary object poses.

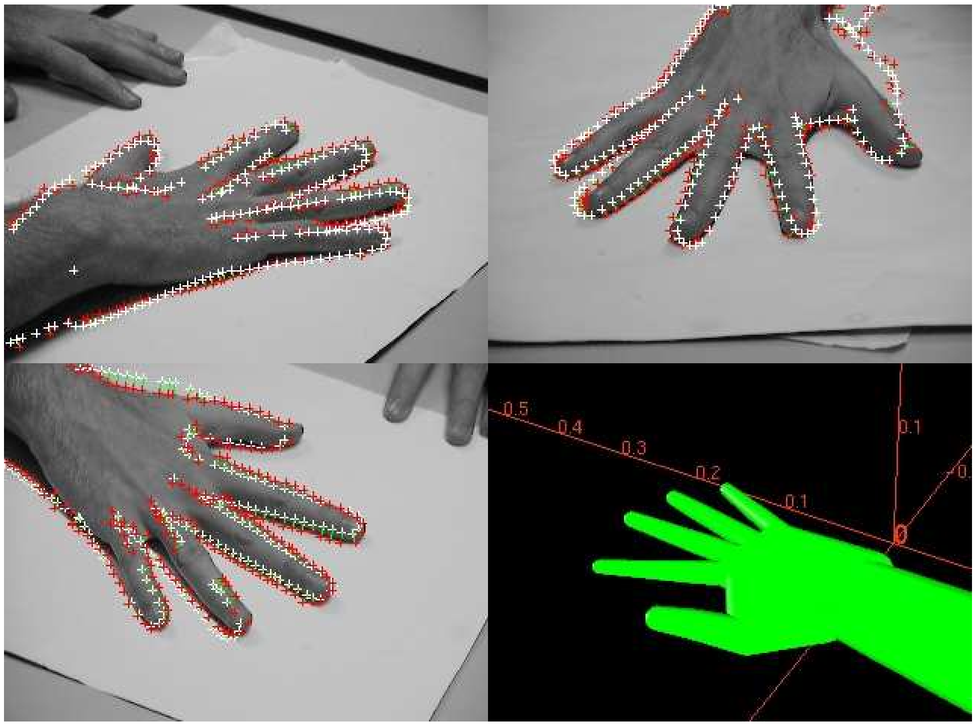
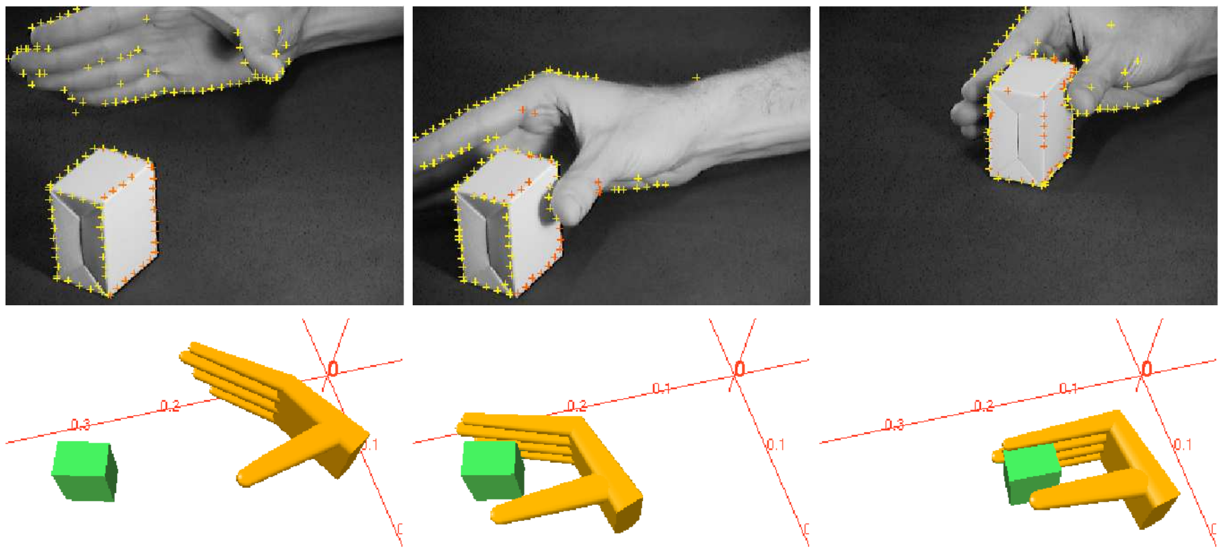


Figure 5.14: Three views of a hand with predicted and located control points superimposed, and the synthesised hand at the position estimated using these images during a tracking sequence.



Frame 4 Frame 301 Frame 463

Figure 5.15: Tracking a hand grasping a box. The graphical images show the view from above.

5.7 Summary

With the aim of building a real-time method to track articulated objects, this chapter examined how to extend a rigid object tracker that has proven successful as a real-time multi-view method (RAPiD). Although rigid and articulated objects are different in terms of how image measurements are associated to motion parameters, the articulated version of RAPiD estimates motion parameters maintaining the same formulation as the original method. The control points are sought in the image in the same way and a linear system is used to estimate the motion parameters with the same formulation as that of the original method.

The articulated RAPiD tracker (ART) is able to track general articulated objects including branched kinematic trees and closed loop kinematic graphs. Details that contributed to enable a real-time implementation were given. These include: a method to process the articulated trees which uses information computed for previous nodes, preventing redundant calculations; and the choice of a method to solve systems of linear equations efficiently.

Satisfactory tracking results have been achieved for man-made objects like books and tools. It was also shown that multiple objects can be tracked as a single articulated object with phantom joints that can represent motion restrictions that may happen, for instance, due to contact.

A 3D hand model has been implemented and tracking experiments were described with the hand moving on its own and in interaction with another object. In both cases, some constraints have been applied to the finger movements. Despite the demonstrated success, the inaccuracy of the model has caused difficulty to track all DOFs of the hand reliably. This can be improved by means of data-driven dimensionality reduction methods as discussed in Section 2.3.7.

Next chapter describes an alternative representation of articulated objects for tracking and presents comparisons with ART. Further details about robustness and computational complexity are also examined.

6

Linear recovery of articulated pose change: comparing pre- and post-imposed constraints

This chapter contrasts two methods of imposing constraints during the tracking of articulated objects. The first method develops constraints using the conventional kinematical approach described in Chapter 5. The second method is that of Drummond and Cipolla [DC02], which tracks the subparts of an articulated object individually, and hence uses the maximal set of variables, but then imposes the motion constraints using Lagrange multipliers.

This chapter shows that these methods, despite their very different formulations, are functionally equivalent in terms of the pose results recovered. Further comparisons between the methods are drawn in terms of computational speed and algorithmic simplicity and robustness, and it is the last area which is the most telling. The comparative results suggest that using built-in constraints is well-suited to tracking individual articulated objects, whereas applying constraints afterwards is most suited to problems involving contact and breakage between articulated (or rigid) objects, where the ability quickly to test tracking performance with constraints turned on or off is desirable.

6.1 Introduction

The ability to track multiple and articulated modelled objects is an important one, not least in the areas of autonomous and teleoperated robotics, visual surveillance and human motion analysis. It is an area which

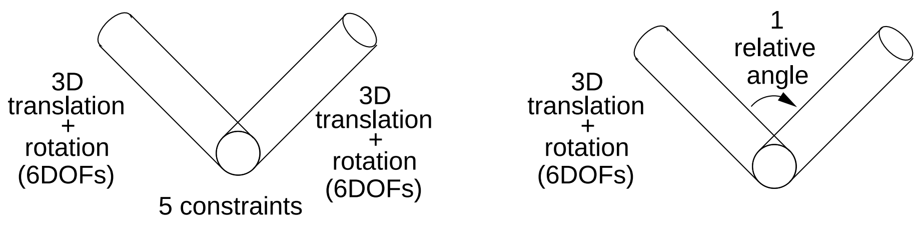


Figure 6.1: Two representations to track articulated objects illustrated with an object with two links and a revolute joint: post- and pre-imposition of constraints.

is still proving challenging more than twenty years after Hogg [Hog83] demonstrated visual tracking of a walking person, modelled using 3D cylinders à la Marr and Nishihara [MN78]. The taxing issues remain those of how to represent the objects and their composited pose, how to associate observable image data with the correct part of the object, by what computational means economically to adjust the high dimensional state vector to improve the fit to current observations, and, lastly, how to overcome fundamental ambiguities in the observations.

As discussed in Chapters 1 and 2, a large variety of ways of addressing these issues have been proposed. The work reported in this chapter is motivated by a desire to understand better the interaction of the human hand with objects at the transition between articulated, independent motion and constrained, possibly rigid, combined motion. An issue of especial concern is how to represent articulated pose to detect a transition from articulated to rigid motion and vice versa. Perhaps each link or subpart should be tracked independently, thereby introducing redundant degrees of freedom, and constraints imposed later in a lower dimensional subspace. Alternatively the available kinematic constraints might be imposed up-front within the tracking process. In support of the latter approach, Rehg *et al.* [RMK03] note that the kinematics define the state of the scene and define the mapping from scene to image. Figure 6.1 illustrates the two approaches.

Exemplars of both classes of method have been reported in a number of application areas. In the former class, and applied to tracking a number of mechanisms, is the work of Drummond and Cipolla [DC00, DC02]. They track subparts independently and then apply constraints using Lagrange multipliers. A similar approach based on kinematic sets is described by Comport *et al.* [CMC04, CMC06]. The object parts parameters and articulated constraints are optimised using an iterative method. In the area of hand tracking is the work of Wu *et al.* [WHY03] who impose constraints on independent subparts via a

Markov network. Hel-Or and Werman [HOW96] fuse constraints and measurements using an extended Kalman filter (EKF) by treating constraints as measurements with zero uncertainty. The kinematic chain approach is applied to tracking a robotic arm by Nickels and Hutchinson [NH01], who use an EKF to recover a state vector of arm joint angles and velocities from point measurements; however they simplify matters by assuming a fixed and known base pose. In hand tracking, Rehg and Kanade [RK95b] put joint angles and pose into a Newton non-linear minimization. In full body tracking, Bregler and Malik [BM98] develop a linear relationship between instantaneous motion and pose change; and Sidenbladh *et al.* [SBF00] use the kinematics in a generative model of image appearance. In each case, however, the surrounding observation and computation methodologies are sufficiently different to make immediate comparison difficult, and no detailed comparison is available in the literature.

Two works in the different classes that appear most similar in other respects are those of Bregler and Malik [BM98] and Drummond and Cipolla [DC02]. Both develop linear expressions for the pose updates of articulated objects using exponential representations of motion and Lie algebra. The differences — in addition of course to the way that articulation constraints are imposed — are the use of different image measures, viz. warped image patches and edges, respectively, and the use of different image projections, viz. scaled orthography and full perspective.

This chapter attempts to make a fair comparison of the two constraint approaches. First, edge data and perspective projection are used for both. Second, both methods are re-implemented using Harris' earlier RAPiD tracker as a common base. In RAPiD [HS90, Har92a], Harris proposed an object pose update which is linear in the elements of the kinematic screw. Here it is shown that RAPiD, described in Chapter 4, is entirely equivalent to Drummond and Cipolla's rigid body tracker based on Lie algebra [DC99], which forms the basis of their articulated tracker [DC02]. The screw is synonymous with the exponential twist, and so for articulated tracking with kinematic constraints it was not needed to slavishly to re-implement Bregler and Malik, but instead the extension of RAPiD to articulated objects described in Chapter 5 is used (here referred to as ART). The implementations therefore share much of their code.

Section 6.2 reviews the way in which scene and image motion are described in Drummond and Cipolla's tracker (here referred to as DCT), and shows that for rigid objects this method is entirely equivalent to RAPiD tracker. Section 6.3 reviews how constraints are imposed in DCT and, for comparison with ART, gives detail of the solution for both kinematic chains and branching trees. Section 6.4 gives a comparison of the two approaches in terms of accuracy, efficiency, and robustness. Finally, Section 6.5 of this chapter draws conclusions on the applicability of the pre-constrained and post-constrained methods in the light of the earlier findings. The main contributions described here have been published in [dTM06].

6.2 Scene and projected image motion

RAPiD was originally formulated using inhomogeneous coordinates, whereas DCT used homogeneous coordinates. To provide continuity with previous work, this chapter must move between both, but will do so mostly without comment. In both RAPiD and DCT, a single rigid object (or rigid subpart of an articulated object) is described in an object frame 0 by the coordinates $\mathbf{X}$ of each of a set of control points — points which may be genuine points on the object, but which more usually are parametrized locations on fixed crease or albedo edges, or are generated on the fly as extremal edges of a curved object, as described in Chapter 4.

In DCT the aim is to recover the 6-vector $\boldsymbol{\alpha}$ of coefficients of the generators of SE(3) describing the change of homogeneous transformation between object and camera. To draw proper comparison with RAPiD, this change is specified in the aligned frame, and so it is the 4×4 transformation from object to world frames that is updated after movement, from $\mathtt{T}_A^W \mathtt{T}_0^A$ to $\mathtt{T}_A^W \mathtt{M} \mathtt{T}_0^A$, where $\mathtt{M} = \exp\left(\sum_i \alpha_i \mathtt{G}_i\right)$. To conform with the conventional screw order, the first and last three generators of the Lie group from [DC02] were switched; in turn they are associated with angular velocities about the X-, Y- and Z- axes, and with translational velocities in the X-, Y- and Z-directions

$$\mathtt{G}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathtt{G}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathtt{G}_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$
$$\mathtt{G}_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathtt{G}_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathtt{G}_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.1}$$

As the change in pose is small, $\mathtt{M}$ is approximated by $\mathtt{M} \approx \mathtt{I} + \sum_i \alpha_i \mathtt{G}_i$, and the velocity in world coordinates can be written

$$\begin{aligned}
\dot{\mathbf{X}}^W &= \begin{pmatrix} \mathtt{I} & \mathbf{t}_{0W} \\ \mathbf{0}^\top & 1 \end{pmatrix} \left(\sum_i \alpha_i \mathtt{G}_i \right) \begin{pmatrix} \mathtt{R}_0^W & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathbf{X}_0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} \mathtt{I} & \mathbf{t}_{0W} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} 0 & -\alpha_3 & \alpha_2 & \alpha_4 \\ \alpha_3 & 0 & -\alpha_1 & \alpha_5 \\ -\alpha_2 & \alpha_1 & 0 & \alpha_6 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{X}_A \\ 1 \end{pmatrix} \\
&= \left(\left[-\mathbf{X}^A \right]_\times \,\middle|\, \mathtt{I}_{3\times 3} \right) \boldsymbol{\alpha} \,. \qquad (6.2)
\end{aligned}$$

Comparison of Eq (6.2) with Eq (4.2) shows that recovering $\boldsymbol{\alpha}$ is identical with recovering the screw $\mathbf{s}$. The equivalence is maintained whichever frame is used to specify motion.

DCT similarly recovers $\boldsymbol{\alpha}$ from image motion, and for completeness it is shown that the measurable image motion derived from the homogeneous expressions in [DC02] is identical with Eq. (4.4). In normalized image coordinates [DC02] has

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \mathtt{R}_W^C \mathbf{X}^W = \mathbf{X}^C \,, \quad \text{and} \quad \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \mathtt{R}_W^C \dot{\mathbf{X}}^W \,, \qquad (6.3)$$

from which the inhomogeneous image motion is derived as

$$\begin{aligned}
\dot{\mathbf{x}} = \frac{1}{w} \begin{pmatrix} \dot{u} - \frac{u}{w}\dot{w} \\ \dot{v} - \frac{v}{w}\dot{w} \\ 0 \end{pmatrix} &= \frac{1}{Z^C} \left[\mathtt{I}_3 - \mathbf{x}[001] \right] \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} \\
&= \frac{1}{Z^C} \left[\mathtt{I}_3 - \mathbf{x}[001] \right] \mathtt{R}_W^C \dot{\mathbf{X}}^W . \qquad (6.4)
\end{aligned}$$

Inserting $\dot{\mathbf{X}}^W$ from Eq. (4.2) into Eq. (6.4) again yields Eq. (4.4).

Thus, for rigid objects, the two methods are functionally equivalent. (Note however that in [DC02] $\boldsymbol{\alpha}$ is recovered for each object in the *object's* frame. The screw recovered, $\boldsymbol{\alpha}^0$, is then within a linear transformation of that in the aligned frame, $\boldsymbol{\alpha}^0 = \mathcal{T}_A^0 \boldsymbol{\alpha}^A$, as clarified later.)

The edge-normal motion is thus related to the state vector $\boldsymbol{\alpha}$ just as it is for the state vector $\mathbf{s}$, described in Section 4.2.2.

6.3 Enforcing constraints after measurement

This section reviews Drummond and Cipolla's method of applying motion constraints to articulated objects *after* making measurements on independent subparts. Their approach is to adjust the optimum screws (now denoted by $\boldsymbol{\alpha}$) obtained for each individual subpart so that the constraints are satisfied and the overall fitting cost is minimized. The solution is reached using Lagrange multipliers.

6.3.1 The cost of (mis-)fitting single object data

For each individual subpart, the measurement system

$$\mathtt{F}\boldsymbol{\alpha} = \mathbf{d} \tag{6.5}$$

in [DC02] is similar to that developed in Eq. (4.6) but the pose adjustment takes place in the object frame, not the aligned frame. The measurements $\mathbf{d}$ are identical with those derived earlier, but the rows $\mathbf{f}$ of $\mathtt{F}$ are generated now using

$$\dot{\mathbf{X}}^W = \begin{pmatrix} \mathtt{R}_0^W & \mathbf{t}_{0W} \\ \mathbf{0}^\top & 1 \end{pmatrix} \left(\sum_i \alpha_i \mathtt{G}_i \right) \begin{pmatrix} \mathbf{X}_0 \\ 1 \end{pmatrix} . \tag{6.6}$$

Without constraints, the optimal least squares solution for each subpart would be used to update that subpart's pose. However, when the constraints are applied, each value of $\boldsymbol{\alpha}$ is modified to $\boldsymbol{\beta}$, and it is necessary to know the additional cost of fitting. For any *individually* suboptimal solution $\boldsymbol{\beta}$, the sum-squared fitting cost is $S_\beta = (\mathtt{F}\boldsymbol{\beta} - \mathbf{d})^\top (\mathtt{F}\boldsymbol{\beta} - \mathbf{d})$ whose minimum is $S_\alpha = (\mathtt{F}\boldsymbol{\alpha} - \mathbf{d})^\top (\mathtt{F}\boldsymbol{\alpha} - \mathbf{d})$. A little manipulation gives the well-known quadratic form for the extra cost

$$S_\beta - S_\alpha = (\boldsymbol{\beta} - \boldsymbol{\alpha})^\top [\mathtt{F}^\top \mathtt{F}] (\boldsymbol{\beta} - \boldsymbol{\alpha}) . \tag{6.7}$$

6.3.2 Developing and imposing constraints

Consider first a simple example of two parts with translational velocities given by vectors of coefficients $\boldsymbol{\beta}_p$ and $\boldsymbol{\beta}_q$ defined in their local Cartesian frames p and q. (That is, the first's velocity is $\beta_{px}\hat{\mathbf{x}}_p + \beta_{py}\hat{\mathbf{y}}_p + \beta_{pz}\hat{\mathbf{z}}_p$, and similarly for the second.) The constraint that the two y-velocities are equal can be written as

$$(\boldsymbol{\beta}_p - \boldsymbol{\beta}_q^p)^\top \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = 0 , \tag{6.8}$$

where the superscript p denotes a value referred to the p-th frame, and absence of superscript denotes a value referred the native object frame given by the subscript. Drummond and Cipolla observed that exactly the same can be done using the $\mathtt{G}$-matrix basis, again provided the $\boldsymbol{\beta}$ values refer to the same frame. So, to impose constraints between two subparts p and q, they wrote

$$(\boldsymbol{\beta}_p - \boldsymbol{\beta}_q^p)^\top \mathbf{c}_{p,q}^k = 0 \tag{6.9}$$

where, to constrain the associated quantity, each $\mathbf{c}_{p,q}^k$, $k = 1, \ldots, K_{p,q}$ is a 6-vector drawn from the set[1]

$$\mathbf{c} = \begin{array}{cccccc} \omega_x & \omega_y & \omega_z & v_x & v_y & v_z \\ \begin{pmatrix}1\\0\\0\\0\\0\\0\end{pmatrix}, & \begin{pmatrix}0\\1\\0\\0\\0\\0\end{pmatrix}, & \begin{pmatrix}0\\0\\1\\0\\0\\0\end{pmatrix}, & \begin{pmatrix}0\\0\\0\\1\\0\\0\end{pmatrix}, & \begin{pmatrix}0\\0\\0\\0\\1\\0\end{pmatrix}, & \begin{pmatrix}0\\0\\0\\0\\0\\1\end{pmatrix} \end{array} . \tag{6.10}$$

As frames p and q are related by $\mathbf{X}^p = \mathtt{T}_q^p \mathbf{X}^q$, the screw $\boldsymbol{\beta}_q^p$ is given by

$$\boldsymbol{\beta}_q^p = \mathrm{Ad}(\mathtt{T}_q^p)\boldsymbol{\beta}_q \ . \tag{6.11}$$

The adjoint transformation, abbreviated below to $\mathcal{T}_q^p \equiv \mathrm{Ad}(\mathtt{T}_q^p)$, is derived in Appendix B. The reason for considering the pose update in each subpart's own frame in this method now becomes clear: it is that the vectors $\mathbf{c}$ become easy to specify and they are independent of pose (for most constraints).

Drummond and Cipolla's method seeks the optimum solution for the articulated object as that which minimizes the additional cost of sub-optimally fitting the individual subparts, subject to all relevant constraints being satisfied. Writing $\mathtt{C} = \mathtt{F}^\top \mathtt{F}$, they consider the problem as one minimizing the sum over all parts

$$\min_{\boldsymbol{\beta}_p \, , \, \boldsymbol{\beta}_q^p} \left[(\boldsymbol{\beta}_p - \boldsymbol{\alpha}_p)^\top \mathtt{C}_p (\boldsymbol{\beta}_p - \boldsymbol{\alpha}_p) \ + \ (\boldsymbol{\beta}_q^p - \boldsymbol{\alpha}_q^p)^\top \mathtt{C}_q^p (\boldsymbol{\beta}_q^p - \boldsymbol{\alpha}_q^p) \right] \tag{6.12}$$

subject to the constraint set

$$[\boldsymbol{\beta}_p - \boldsymbol{\beta}_q^p]^\top \mathbf{c}_{p,q}^{k=1\ldots K_{p,q}} = 0 \ . \tag{6.13}$$

The relationship between $\mathtt{C}_p^q$ and $\mathtt{C}_q$ is detailed in Appendix B. However, we note that the second term of the minimization is actually invariant to changes in frame, and the problem is more efficiently written in terms of the $\boldsymbol{\beta}$'s in their native frames, so that the second cost term becomes

$$(\boldsymbol{\beta}_q - \boldsymbol{\alpha}_q)^\top \mathtt{C}_q (\boldsymbol{\beta}_q - \boldsymbol{\alpha}_q) \tag{6.14}$$

[1]The constraints are not restricted to this set. If the joint jas a pivot that is not at the origin, its contraint vector would be different.

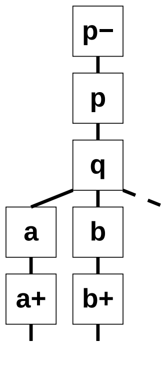


Figure 6.2: A general branching model.

and the constraints are modified to

$$[\boldsymbol{\beta}_p - \mathcal{T}_q^p \boldsymbol{\beta}_q]^\top \mathbf{c}_{p,q}^{k=1\ldots K_{p,q}} = 0\ . \tag{6.15}$$

Depending on the physical constraints, each set contains $0 \le K_{p,q} \le 6$ individual constraint equations, where the lower and upper equalities indicate, respectively, complete freedom and complete rigidity between the two subparts.

6.3.3 Handling general articulated graphs in DCT

In [DC02], Drummond and Cipolla describe the solution for two subparts connected by a hinge, and in [DC01] they sketch a solution for a chain of subparts which is also demonstrated with a kinematic tree (human body). However, for comparison with the articulated RAPiD tracker, it is necessary properly to understand how DCT might handle branching in the kinematic chain. A solution that handles both non-branching chains and branching trees is developed here.

Consider $N+1$ subparts arranged in an articulated tree. The DCT problem becomes one of finding

$$\min_{\text{All } \boldsymbol{\beta}_q} \sum_q (\boldsymbol{\beta}_q - \boldsymbol{\alpha}_q)^\top \mathtt{C}_q (\boldsymbol{\beta}_q - \boldsymbol{\alpha}_q) \tag{6.16}$$

subject to sets of motion constraint equations. Each subpart q with children generates one constraint set for each of its children q^+:

$$[\boldsymbol{\beta}_q - \mathcal{T}_{q^+}^q \boldsymbol{\beta}_{q^+}]^\top \mathbf{c}_{q,q^+}^k = 0 \quad 1 \le k \le K_{q,q^+}\ . \tag{6.17}$$

Now consider the subpart q as shown in Fig. 6.2, with ancestors p, p^-, etc, and with possible multiple lines of descendants a, a^+, etc, b, b^+ etc. The constraint sets referencing $\boldsymbol{\beta}_q$ involve the parent of q and each of its children, if they exist. That is,

$$\text{If parent } p \text{ exists :} \quad \left[\boldsymbol{\beta}_p - \mathcal{T}^p_q \boldsymbol{\beta}_q\right]^\top \mathbf{c}^k_{p,q} = 0 \tag{6.18}$$

$$\text{If 1st child } a \text{ exists :} \quad \left[\boldsymbol{\beta}_q - \mathcal{T}^q_a \boldsymbol{\beta}_a\right]^\top \mathbf{c}^k_{q,a} = 0 \tag{6.19}$$

$$\text{If 2nd child } b \text{ exists :} \quad \left[\boldsymbol{\beta}_q - \mathcal{T}^q_b \boldsymbol{\beta}_b\right]^\top \mathbf{c}^k_{q,b} = 0 \tag{6.20}$$

and so on if q has further children.

That part of the Lagrange system depending of differentiation w.r.t. $\boldsymbol{\beta}_q$ is therefore

$$2\mathtt{C}_q(\boldsymbol{\beta}_q - \boldsymbol{\alpha}_q) - \sum_{m=1}^{K_{p,q}} \lambda^m_{p,q} {\mathcal{T}^p_q}^\top \mathbf{c}^m_{p,q} + \sum_{m=1}^{K_{q,a}} \lambda^m_{q,a} \mathbf{c}^m_{q,a} \left(+ \sum_{m=1}^{K_{q,b}} \lambda^m_{q,b} \mathbf{c}^m_{q,b} + \dots \right) = \mathbf{0}_6 \ , \tag{6.21}$$

where the first term derives from the cost, and the remainder from the constraints, and where the λ's are the Lagrange multipliers. The first summation is omitted if q has no parent, the second if q has no children, and further terms of the sort shown in brackets are added if there are further children. Rearranging,

$$\boldsymbol{\beta}_q = \boldsymbol{\alpha}_q + \sum_{m=1}^{K_{p,q}} \lambda^m_{p,q} \frac{1}{2} \mathtt{C}_q^{-1} {\mathcal{T}^p_q}^\top \mathbf{c}^m_{p,q} - \sum_{m=1}^{K_{q,a}} \lambda^m_{q,a} \frac{1}{2} \mathtt{C}_q^{-1} \mathbf{c}^m_{q,a} \left(- \sum_{m=1}^{K_{q,b}} \lambda^m_{q,b} \frac{1}{2} \mathtt{C}_q^{-1} \mathbf{c}^m_{q,b} - \dots \right) . \tag{6.22}$$

Similar expressions can be written for the other $\boldsymbol{\beta}$'s, and replacing all the $\boldsymbol{\beta}$'s in the (p,q) constraint set (Eq. 6.18) gives

$$\begin{aligned} &\sum_{m=1}^{K_{p^-,p}} \lambda^m_{p^-,p} {\mathbf{c}^k_{p,q}}^\top \mathtt{C}_p^{-1} {\mathcal{T}^{p^-}_p}^\top \mathbf{c}^m_{p^-,p} - \sum_{m=1}^{K_{p,q}} \lambda^m_{p,q} {\mathbf{c}^k_{p,q}}^\top \left[\mathtt{C}_p^{-1} + \mathcal{T}^p_q \mathtt{C}_q^{-1} {\mathcal{T}^p_q}^\top \right] \mathbf{c}^m_{p,q} \\ &\quad + \sum_{m=1}^{K_{q,a}} \lambda^m_{q,a} {\mathbf{c}^k_{p,q}}^\top \mathcal{T}^p_q \mathtt{C}_q^{-1} \mathbf{c}^m_{q,a} \left(+ \sum_{m=1}^{K_{q,b}} \lambda^m_{q,b} {\mathbf{c}^k_{p,q}}^\top \mathcal{T}^p_q \mathtt{C}_q^{-1} \mathbf{c}^m_{q,b} + \dots \right) \\ &\qquad = 2{\mathbf{c}^k_{p,q}}^\top \left[\mathcal{T}^p_q \boldsymbol{\alpha}_q - \boldsymbol{\alpha}_p \right] \quad k = 1, \dots, K_{p,q} \ . \end{aligned} \tag{6.23}$$

Replacing all those in the (q,a) constraint set gives

$$\begin{aligned} &\sum_{m=1}^{K_{p,q}} \lambda^m_{p,q} {\mathbf{c}^k_{q,a}}^\top \mathtt{C}_q^{-1} {\mathcal{T}^p_q}^\top \mathbf{c}^m_{p,q} - \sum_{m=1}^{K_{q,a}} \lambda^m_{q,a} {\mathbf{c}^k_{q,a}}^\top \left[\mathtt{C}_q^{-1} + \mathcal{T}^q_a \mathtt{C}_a^{-1} {\mathcal{T}^q_a}^\top \right] \mathbf{c}^m_{q,a} \\ &\quad + \sum_{m=1}^{K_{a,a^+}} \lambda^m_{a,a^+} {\mathbf{c}^k_{q,a}}^\top \mathcal{T}^q_a \mathtt{C}_a^{-1} \mathbf{c}^m_{a,a^+} \left(+ \sum_{m=1}^{K_{q,b}} \lambda^m_{q,b} {\mathbf{c}^k_{q,a}}^\top \mathtt{C}_q^{-1} \mathbf{c}^m_{q,b} + \dots \right) \\ &\qquad = 2{\mathbf{c}^k_{q,a}}^\top \left[\mathcal{T}^q_a \boldsymbol{\alpha}_a - \boldsymbol{\alpha}_q \right] \quad k = 1, \dots, K_{q,a}. \end{aligned} \tag{6.24}$$

If there are multiple children, b and so on, expressions similar to Eq.(6.24) can be written for the (q,b) constraint set by swapping $a \leftrightarrow b$, and so on.

These expressions are more compactly expressed as

$$\begin{aligned} \mathtt{P}_{p,q}\boldsymbol{\lambda}_{p^-,p} + \mathtt{Q}_{p,q}\boldsymbol{\lambda}_{p,q} + \mathtt{R}^a_{p,q}\boldsymbol{\lambda}_{q,a}\left(+\mathtt{R}^b_{p,q}\boldsymbol{\lambda}_{q,b} + \mathtt{R}^c_{p,q}\boldsymbol{\lambda}_{q,c} + \ldots\right) &= \mathbf{l}_{p,q} && (6.25)\\ \mathtt{P}_{q,a}\boldsymbol{\lambda}_{p,q} + \mathtt{Q}_{q,a}\boldsymbol{\lambda}_{q,a} + \mathtt{R}^a_{q,a}\boldsymbol{\lambda}_{a,a^+}\left(+\mathtt{S}^b_{q,a}\boldsymbol{\lambda}_{q,b} + \mathtt{S}^c_{q,a}\boldsymbol{\lambda}_{q,c} + \ldots\right) &= \mathbf{l}_{q,a}\,. && (6.26)\end{aligned}$$

Again, the bracketed terms are used for additional children, and a further equation of the form of Eq. (6.26) generated for each additional child, with $a \leftrightarrow b$, $a \leftrightarrow c$, etc.

The k-th row and m-th column of the various quantities are

$$\begin{aligned} \mathbf{l}_{p,q}(k) &= 2{\mathbf{c}^k_{p,q}}^\top \left[\mathcal{T}^p_q\boldsymbol{\alpha}_q - \boldsymbol{\alpha}_p\right] && (6.27)\\ \mathtt{P}_{p,q}(k,m) &= {\mathbf{c}^k_{p,q}}^\top \mathtt{C}^{-1}_p {\mathcal{T}^{p^-}_p}^\top \mathbf{c}^m_{p^-,p} && (6.28)\\ \mathtt{Q}_{p,q}(k,m) &= -{\mathbf{c}^k_{p,q}}^\top \left[\mathtt{C}^{-1}_p + \mathcal{T}^p_q\mathtt{C}^{-1}_q{\mathcal{T}^p_q}^\top\right]\mathbf{c}^m_{p,q} && (6.29)\\ \mathtt{R}^a_{p,q}(k,m) &= {\mathbf{c}^k_{p,q}}^\top \mathcal{T}^p_q\mathtt{C}^{-1}_q\mathbf{c}^m_{q,a} && (6.30)\\ \mathtt{S}^b_{q,a}(k,m) &= \mathbf{c}^k_{q,a}\mathtt{C}^{-1}_q\mathbf{c}^m_{q,b} && (6.31)\end{aligned}$$

6.3.4 Specific cases of graphs

Now consider specific cases through examples shown in Fig. 6.3. Below we detail the implications of each of these types of topographies in the solution with DCT. General graphs can be modelled by combining elements of each of these cases.

Non-branching kinematic chains

When the subparts are arranged as a linear chain, labelled here from 0 to N, as illustrated in Figure 6.3(a), the system to be solved for the $\boldsymbol{\lambda}$'s becomes block tridiagonal,

$$\begin{pmatrix} \mathtt{Q}_{01} & \mathtt{R}^2_{01} & 0 & 0 & \ldots & 0\\ \mathtt{P}_{12} & \mathtt{Q}_{12} & \mathtt{R}^3_{12} & 0 & \ldots & 0\\ 0 & \mathtt{P}_{23} & \mathtt{Q}_{23} & \mathtt{R}^4_{23} & \ldots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \ldots & \mathtt{P}_{N-2,N-1} & \mathtt{Q}_{N-2,N-1} & \mathtt{R}^N_{N-2,N-1}\\ 0 & 0 & \ldots & 0 & \mathtt{P}_{N-1,N} & \mathtt{Q}_{N-1,N}\end{pmatrix}\begin{pmatrix}\boldsymbol{\lambda}_{01}\\ \boldsymbol{\lambda}_{12}\\ \boldsymbol{\lambda}_{23}\\ \vdots\\ \boldsymbol{\lambda}_{N-2,N-1}\\ \boldsymbol{\lambda}_{N-1,N}\end{pmatrix} = \begin{pmatrix}\mathbf{l}_{01}\\ \mathbf{l}_{12}\\ \mathbf{l}_{23}\\ \vdots\\ \mathbf{l}_{N-2,N-1}\\ \mathbf{l}_{N-1,N}\end{pmatrix} \quad (6.32)$$

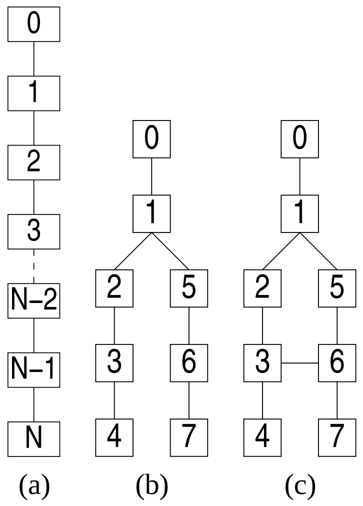


Figure 6.3: Examples of special cases of kinematic chains: (a) a single kinematic chain; (b) a branching kinematic tree; and (c) a closed loop kinematic chain – the same as in Figure 5.2(b).

for which a standard method, $\mathcal{O}(N)$ in the number of links, exists to recover the $\boldsymbol{\lambda}$ values without explicitly building or inverting the matrix (e.g. [PTVF88]). Once the $\boldsymbol{\lambda}_{q,q+1}$ are known, the constrained screws $\boldsymbol{\beta}_q$ are derived for each subpart from Eq. (6.22). Since the constrained screws $\boldsymbol{\beta}_q$ are expressed individually in each object part coordinate frame, there is a risk that rounding errors lead to small breakage in the articulated constraints. These can be corrected by reinforce the constraints by running down the kinematic chain and updating the poses using the adjoint transformations [DC01]. Algorithm 2 summarizes this method of pose update *for a kinematic tree*.

Kinematic trees

When branching occurs, the system loses its block tridiagonal form, and its structure depends of course on the object's structure. By way of example, the structure in Fig. 6.3(b) generates the following system for solution.

$$\begin{pmatrix} \mathtt{Q}_{01} & \mathtt{R}_{01}^{2} & 0 & 0 & \mathtt{R}_{01}^{5} & 0 & 0 \\ \mathtt{P}_{12} & \mathtt{Q}_{12} & \mathtt{R}_{12}^{3} & 0 & \mathtt{S}_{12}^{5} & 0 & 0 \\ 0 & \mathtt{P}_{23} & \mathtt{Q}_{23} & \mathtt{R}_{23}^{4} & 0 & 0 & 0 \\ 0 & 0 & \mathtt{P}_{34} & \mathtt{Q}_{34} & 0 & 0 & 0 \\ \mathtt{P}_{15} & \mathtt{S}_{15}^{2} & 0 & 0 & \mathtt{Q}_{15} & \mathtt{R}_{15}^{6} & 0 \\ 0 & 0 & 0 & 0 & \mathtt{P}_{56} & \mathtt{Q}_{56} & \mathtt{R}_{56}^{7} \\ 0 & 0 & 0 & 0 & 0 & \mathtt{P}_{67} & \mathtt{Q}_{67} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda}_{01} \\ \boldsymbol{\lambda}_{12} \\ \boldsymbol{\lambda}_{23} \\ \boldsymbol{\lambda}_{34} \\ \boldsymbol{\lambda}_{15} \\ \boldsymbol{\lambda}_{56} \\ \boldsymbol{\lambda}_{67} \end{pmatrix} = \begin{pmatrix} \mathbf{l}_{01} \\ \mathbf{l}_{12} \\ \mathbf{l}_{23} \\ \mathbf{l}_{34} \\ \mathbf{l}_{15} \\ \mathbf{l}_{56} \\ \mathbf{l}_{67} \end{pmatrix} \tag{6.33}$$

The structure is block symmetric, and is likely to remain sparse, but this system can no longer use a general $\mathcal{O}(n)$ solver. However, in [DC01], Drummond and Cipolla use a statistics propagation method

Algorithm 2 One iteration of the D&C's Tracker, with modifications for trees.

```
1:  for each subpart $q = 0 \cdots N$ do
2:     for each camera do
3:        for each visible control point $i$ do
4:           Construct $\mathtt{H}_i$ and thence $\mathbf{f}_i$
5:           Search for image edge, compute $d_i$
6:           Add row $\mathbf{f}_i$ to measurement matrix $\mathtt{F}_q$
7:        end for
8:     end for
9:     Derive $\boldsymbol{\alpha}_q$ from $\mathtt{F}_q \boldsymbol{\alpha}_q = \mathbf{d}_q$ (Eq 6.5)
10:    Compute ${\mathtt{C}_q}^{-1}$, from $\mathtt{C}_q = {\mathtt{F}_q}^\top \mathtt{F}_q$
11:    Compute $\mathtt{Ad}(\mathtt{T}^q_{q^+})$ from Eq. (B.9) (except leaves)
12: end for
13: for each constraint set do
14:    Compute $\mathtt{P}$, $\mathtt{Q}$, $\mathtt{R}$, $\mathtt{S}$ and $\mathbf{l}_q$ from Eqs (6.27-6.31)
15: end for
16: Solve tridiagonal (Eq. 6.32) or block symmetric (e.g. Eq. 6.33) system for all $\boldsymbol{\lambda}$
17: for each subpart $q$ do
18:    Derive $\boldsymbol{\beta}_q$ from Equation (6.22) and update part poses.
19: end for
```

and a breadth first search to compute the Lagrange coefficients for each joint sequentialy, and thus in $\mathcal{O}(n)$.

Closed loop kinematic chains

The general solution of Equations (6.25) and (6.26) can also be employed to build the constraint system for closed loop kinematic chains. In the example of Fig. 6.3(c) the constraints $\boldsymbol{\lambda}_{34}$ and $\boldsymbol{\lambda}_{36}$ can be described as links to brother nodes. The same happens between $\boldsymbol{\lambda}_{36}$ and $\boldsymbol{\lambda}_{67}$ for the other side of the branch. Therefore, the constraint system becomes:

$$\begin{pmatrix} \mathtt{Q}_{01} & \mathtt{R}^2_{01} & 0 & 0 & 0 & \mathtt{R}^5_{01} & 0 & 0 \\ \mathtt{P}_{12} & \mathtt{Q}_{12} & \mathtt{R}^3_{12} & 0 & 0 & \mathtt{S}^5_{12} & 0 & 0 \\ 0 & \mathtt{P}_{23} & \mathtt{Q}_{23} & \mathtt{R}^4_{23} & \mathtt{R}^6_{23} & 0 & 0 & 0 \\ 0 & 0 & \mathtt{P}_{34} & \mathtt{Q}_{34} & \mathtt{S}^6_{34} & 0 & 0 & 0 \\ 0 & 0 & \mathtt{P}_{36} & \mathtt{S}^4_{36} & \mathtt{Q}_{36} & 0 & \mathtt{R}^5_{36} & \mathtt{R}^7_{36} \\ \mathtt{P}_{15} & \mathtt{S}^2_{15} & 0 & 0 & 0 & \mathtt{Q}_{15} & \mathtt{R}^6_{15} & 0 \\ 0 & 0 & 0 & 0 & \mathtt{R}^3_{56} & \mathtt{P}_{56} & \mathtt{Q}_{56} & \mathtt{R}^7_{56} \\ 0 & 0 & 0 & 0 & \mathtt{S}^3_{67} & 0 & \mathtt{P}_{67} & \mathtt{Q}_{67} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda}_{01} \\ \boldsymbol{\lambda}_{12} \\ \boldsymbol{\lambda}_{23} \\ \boldsymbol{\lambda}_{34} \\ \boldsymbol{\lambda}_{36} \\ \boldsymbol{\lambda}_{15} \\ \boldsymbol{\lambda}_{56} \\ \boldsymbol{\lambda}_{67} \end{pmatrix} = \begin{pmatrix} \mathbf{l}_{01} \\ \mathbf{l}_{12} \\ \mathbf{l}_{23} \\ \mathbf{l}_{34} \\ \mathbf{l}_{36} \\ \mathbf{l}_{15} \\ \mathbf{l}_{56} \\ \mathbf{l}_{67} \end{pmatrix} \tag{6.34}$$

Again, the constraints system loses its block tridiagonal form and becomes less sparse. Similarly to the ART representation (see Section 5.3), an additional issue is that extra constraints added by loops introduce singularities into the constraints system, demanding rank monitoring to avoid noise imposing spurious rigidity. The iterative solution of [DC01] is an alternative that does not require rank monitoring. Note that the flow control of DCT (Algorithm 2) needs to be modified in order to detect and deal with loop closing.

6.4 Experimental comparison of ART and DCT

6.4.1 Similarity of results

Despite their very different constraint formulations, both the articulated RAPiD tracker developed in Chapter 5 and the Drummond and Cipolla tracker reviewed in Section 6.3 are single-shot linear methods, and, given that relationships between scene and image have been shown to be identical, one should expect them to give the same results. To verify this against known ground truth, a CAD model of two hinged subparts was generated, and both trackers deployed on the resulting imagery as the hinged opened between successive frames. In each experiment increasing amounts of zero-mean Gaussian noise were added to the image displacements measured between control points and image edges. Figure 6.4(a) shows a typical view and match set. The lower trace in Figure 6.4(b) shows the deviation in degrees from the veridical hinge angle recovered using ART. The upper trace shows the rising standard deviation. Figure 6.4(c) shows the same when the individual subparts are tracked without imposition of constraints. Figure 6.4(d) shows that as soon as the constraints are applied in DCT the modified angle becomes all but identical with that recovered using kinematic constraints in (b). Indeed, the results from ART and DCT differed by at most parts in 10^5, effectively at the limits of expected numerical accuracy.

6.4.2 Computational cost

Figure 6.5 shows comparisons of the times for a single update cycle of the core operations of ART and DCT, run on a 1.8 GHz Pentium 4. Times were accumulated over many trials, with each data point taking at least 20 s to collect, giving each datum the same fractional error of order $10^{-2}\%$. Also, in both cases, similar care was taken to avoid unnecessary calculation. The object is taken to have n subparts with p control points per subpart, and is made up of a single articulated chain. Subfigure (a) shows that with up to 10 subparts there is negligible difference between the methods, and that at 30 subparts the time taken by ART is about twice that by DCT. Up to 100 parts the cost in ART is still dominated by $\mathcal{O}(N^2)$, but beyond (not shown) it does become $\mathcal{O}(N^3)$ as expected for the Cholesky decomposition

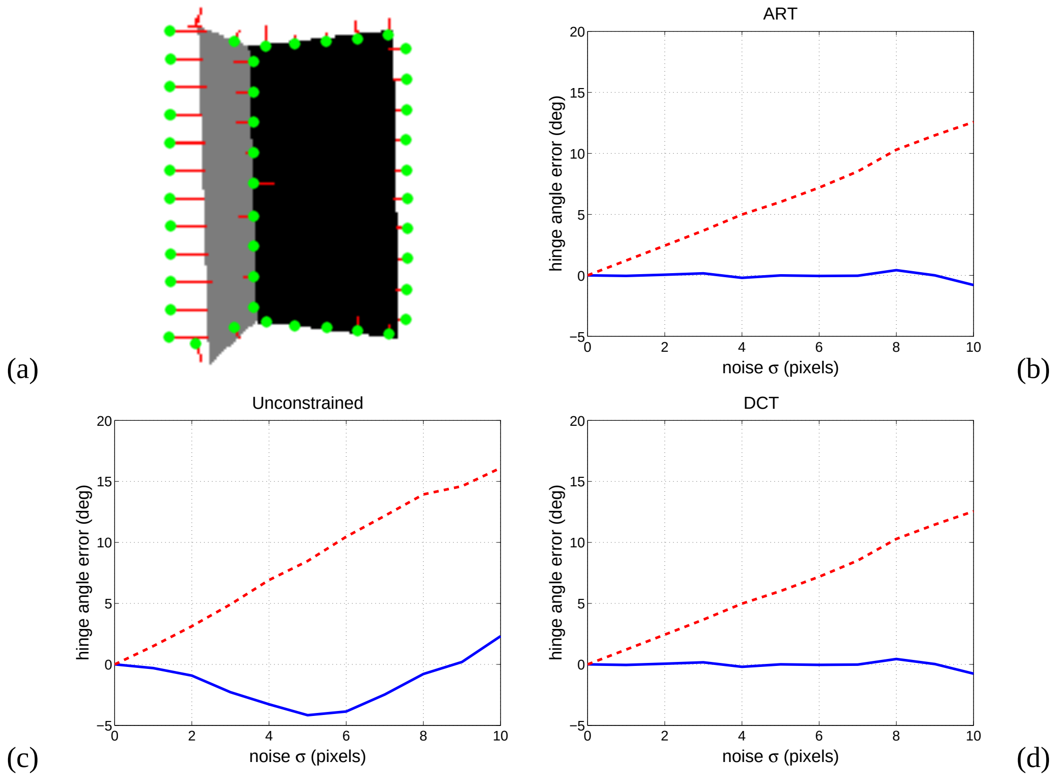


Figure 6.4: Comparison of the numerical performance of the Articulated RAPiD tracker with the DC tracker for two subparts connected by a single revolute joint as increasing Gaussian noise is added to the measured displacements for the object shown in part (a), which is CAD-generated, so exact ground truth is available. Results from: (b) the articulated RAPiD Tracker; (c) unconstrained subpart tracking; (d) after adding the constraints in the DC tracker. In each graph the solid curve is the deviation from ground truth, and the dashed curve is the standard deviation.

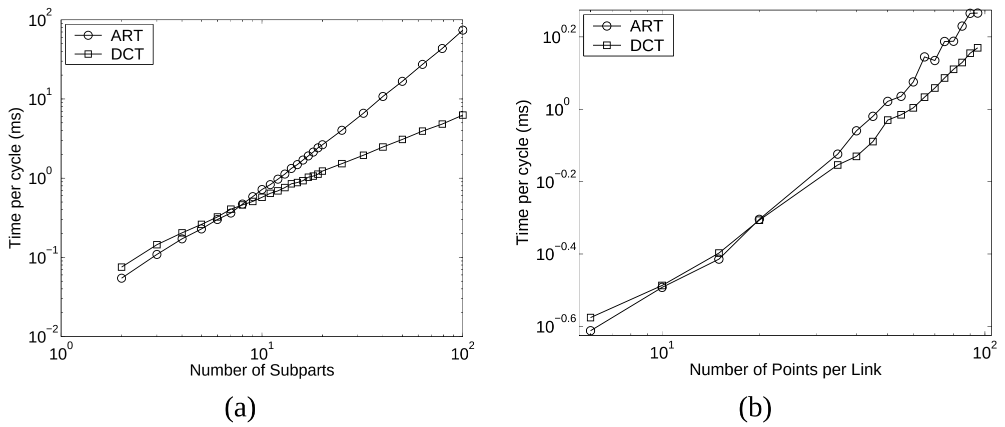
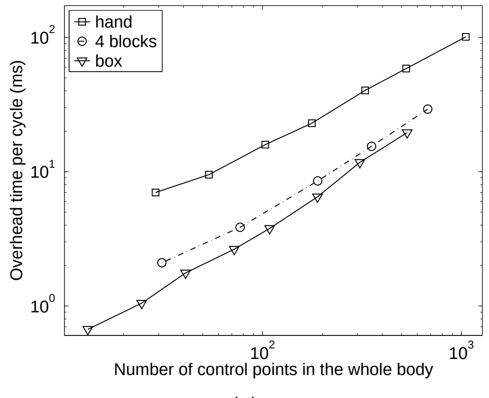
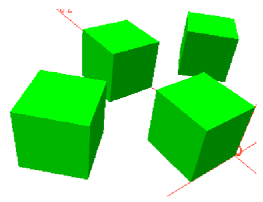


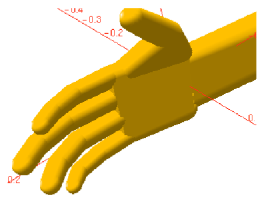
Figure 6.5: Times (in ms) for update cycles of the cores of ART and DCT compared. The comparisons are: (a) As a function of number of subparts N with a fixed number of six control points per part; and (b) as a function of the number of control points p per subpart, with a fixed number of five subparts. In this region ART is predominantly $\mathcal{O}(N^2)$ (but is expected to become order $\mathcal{O}(N^3)$ for higher values of N), while DCT remains $\mathcal{O}(N)$. For fixed N both methods exhibit an order p dependence.



(a)



(b)



(c)

Figure 6.6: (a) Times (in ms) for the shared operations (occlusion testing, image search and measurement, and related OS overhead) in one cycle of ART and DCT, as a function of the average number of visible control points in the whole body. (b,c) show the “4 blocks” and “hand” models related to the upper two traces in part (a).

of the $(n+5)^2$ matrix $\mathtt{F}^\top\mathtt{F}$ that occurs in the least squares solution of $\mathtt{F}\mathbf{s} = \mathbf{d}$. (Recall $\mathbf{s}$ contains the six screw components *and* $n+1$ joint parameters.) Figure 6.5(b) shows that both methods scale predominantly linearly with the number of control points p per link. (Exploration of the apparently noisy results at high p suggests these are a reproducible and uninteresting quirk of memory management in the matrix library used.)

Figure 6.6(a) shows, for the same CPU, the computational time of operations that are shared by both trackers, including occlusion handling, projection of control points, image operations, and related operating system overhead. Three objects of increasing complexity were used: a single part box with 6 DOF, an object with 4 parts and 9 DOF (Figure 6.6b), and a hand with forearm model with 17 parts and 28 DOF (Figure 6.6c). The tests used 320×240 images acquired from three cameras. The search range from control points was ±10 pixels. The number of potential control points was increased in steps but, because of occlusion, not all the control points are always visible, and so the average visible number for a sequence is used as the abscissa. The time increase is dominated by a linear dependence on the number of visible control points.

For all but small problems, DCT is faster than ART and its linear behaviour with number of parts demands its use on large problems, say involving more than 100 subparts. However, for modest numbers of subparts, the timing differences are insignificant compared with the equally shared costs. As examples, the different cores of DCT and ART took 0.45ms and 0.41ms respectively, to run on the four-block object

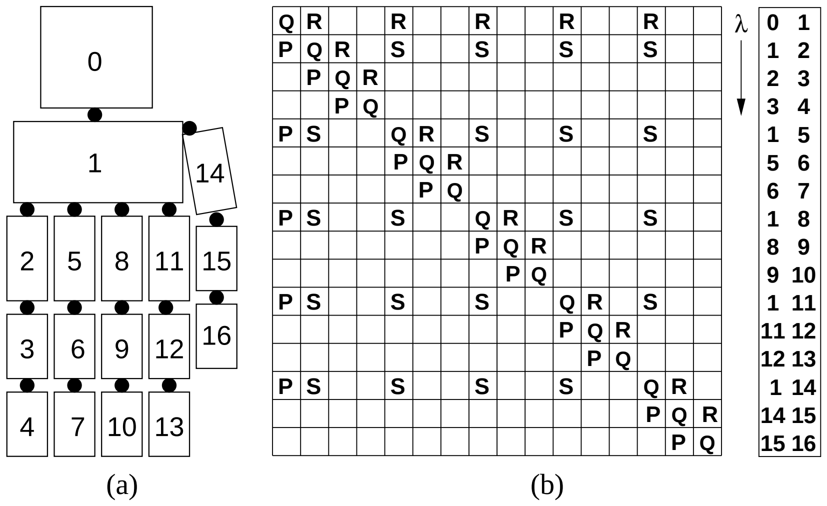


Figure 6.7: The hand model (a), and the block structures of its corresponding matrix and λ subscripts (b).

with 100 control points, but shared operations took some 6 ms; on the hand model with some 170 control points ART's time was 2.5 ms, but the shared operations took over 13 ms.

The core times given for DCT are so far those for simple chains where the tridiagonal system is solved. On a chain with the same number of parts as the hand and, again, 170 control points, DCT took some 1.2 ms, but solving the actual branched system shown in Fig. 6.7 took some 2.5 times longer, and hence taking a little longer than ART. It was found that once any branching occurs, general matrix solvers give a solution time dependent mostly on the number of subparts, not the degree of branching: for example, with 16 parts, two chains of eight takes the same time as four chains of four. However, it was not investigated to what extent careful hand-tuning of the code to a specific object would change this.

6.4.3 Algorithmic robustness

In general, tracking unconstrained subparts is certainly more fragile than tracking subparts collectively in a constrained system. The more degrees of freedom, the greater the likelihood of fitting to noise. Breakage might occur when, for example, the available control points do not provide sufficient constraint — a cylinder with points only along its extremal boundaries is an example (Figure 6.8) — or when a subpart moves quickly at the end of a long linkage (Figure 6.9).

On first consideration this seems to make DCT inherently less robust than ART, but this is not the

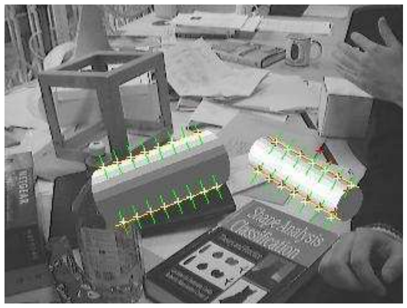


Figure 6.8: A synthetic two-cylinders articulated object with a cluttered background. Since no control points are located on the ends of the cylinders, it is not possible to track the two cylinders without articulated constraints.

case. Assume that, at the start of an update cycle, both trackers have placed their subparts in the same location. They will generate the same control points and therefore generate the same measurements.

Suppose first that there are enough measurements to determine the pose of each subpart, but that the measurements belonging to a particular subpart are quite erroneous. Using ART, the costs are immediately shared and the pose updates of all the subparts will be disturbed somewhat, whereas in DCT the initial pose updates ($\boldsymbol{\alpha}$) will all be better, except that for the erroneous one. However, when the constraints are imposed, the costs of (mis-)fitting with the $\boldsymbol{\beta}$ values are *all* correctly accounted for and minimized. Because the measurements are the same, the solutions must be same.

Suppose now that there are insufficient measurements to determine the pose update of a particular subpart. It is important not to discard those measurements that are available, but instead invent a pose update $\boldsymbol{\alpha}$ for that subpart against which the change in cost when using some $\boldsymbol{\beta}$ instead may be measured, a change which could easily now be a decrease rather than the usual increase. This tactic extends to zero measurements, where the cost change when moving the part is zero. However, what is lost in this case is the ability to separate the recovery of the Lagrange multipliers and the $\boldsymbol{\beta}$ values. Insufficient measurements means that $\mathtt{C} = \mathtt{F}^\top\mathtt{F}$ is rank deficient, and so cannot be inverted for Eq. (6.22). There appears no generalizable alternative to solving the linear system in its entirety, with a stacked vector of $\boldsymbol{\beta}$'s and $\boldsymbol{\lambda}$'s as the unknowns. Alternatively, weak regularisation can be used to compute an approximate of $\mathtt{C}$.

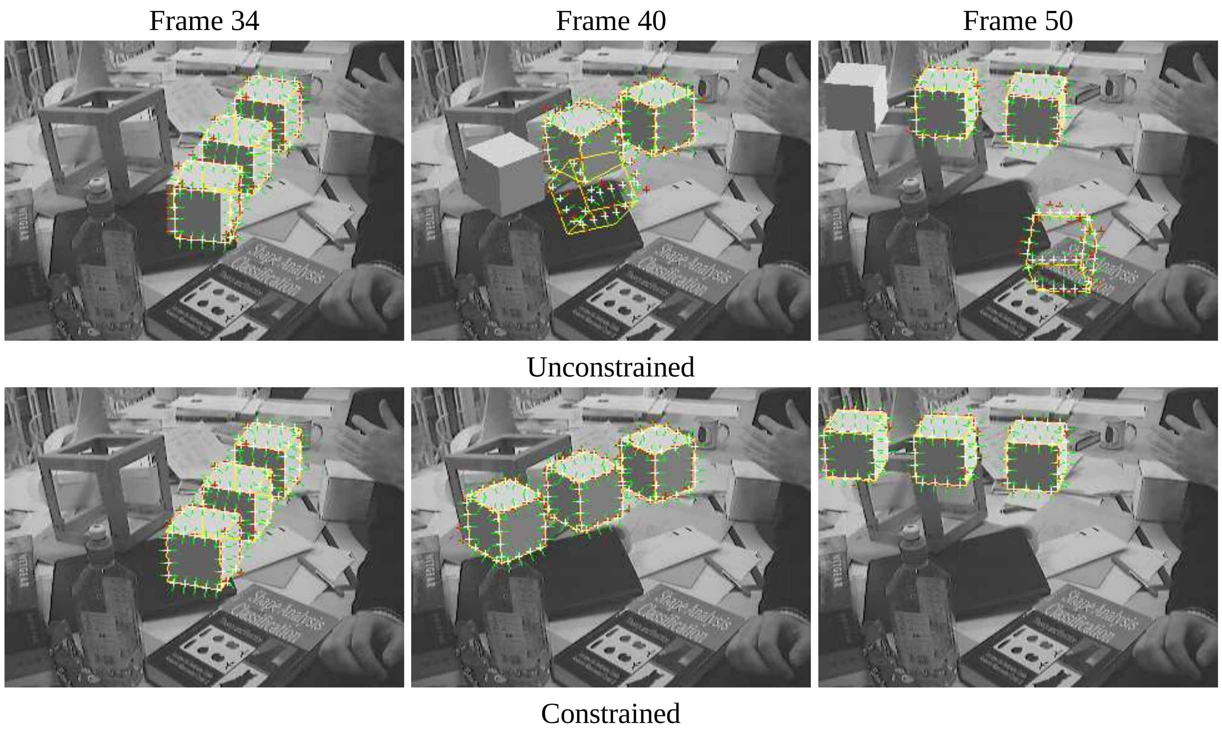


Figure 6.9: An example showing (top) tracking failure when subparts completely unconstrained, contrasted with (bottom) successful tracking the same parts are constrained. The failure arises here because of scene rotation giving rise to rapid motion at the end of the chain of parts. (As the text explains, this is not a comparison between DCT and ART.)

6.4.4 Data robustness

To consider the likely cost of computing the solutions to ART and DCT in a robust manner using a random sampler such as RANSAC [FB81] or LMedS [RL87], the formal relationship already mentioned in Section 4.2.4 is used:

$$P = 1 - (1 - \psi^m)^I \ , \tag{6.35}$$

where P is the confidence that a valid minimal set of m measurements will be selected after I trials when the fraction of valid data is ψ. Although, for a range of problems, experiment indicates that this should be regarded as an underestimate of the number of trials I actually required, the comparative results will be less affected. For N subparts with $(N-1)$ 1 DOF joints, the trials required are

$$I_{\text{ART}} = \frac{\log(1-P)}{\log(1-\psi^{5+N})} \ ; \quad I_{\text{DCT}} = N\frac{\log(1-P)}{\log(1-\psi^6)} \ . \tag{6.36}$$

Multiplying these expressions by the different time costs per iteration of each method, gives the pairs of curves in Figure 6.10 for 10, 20 and 30 subparts, derived at $P = 95\%$ confidence, and each plotted as a function of the percentage of outlying or invalid data, $100(1-\psi)$. Also shown is the locus of the crossover point, as the number of parts is varied. For a modest number of parts and quite low values of corruption, ART's requirements undercut DCT's, but more remarkable is how rapidly a certain fraction of outliers becomes intolerable as the number of subparts rises.

6.5 Discussion and conclusions

This chapter contrasted two different methods of tracking articulated objects in a video sequence. The first method is the straightforward, but novel, extension of Harris' RAPiD tracker to handle articulation (dubbed ART) by explicitly including the kinematics in the linearized pose update equation, described in Chapter 5. The second is the articulated tracker of Drummond and Cipolla (dubbed DCT) which imposes motion constraints on subparts after they have been tracked independently.

To motivate the comparison, it was shown that Harris's original method for recovering the kinematic screw of a single rigid object (described in Chapter 4) is entirely equivalent to Drummond and Cipolla's later formalism based on Lie algebra. It was noted that Bregler and Malik had also described the theory of a kinematics-based tracker using Lie algebra, and it was concluded that ART is equivalent to

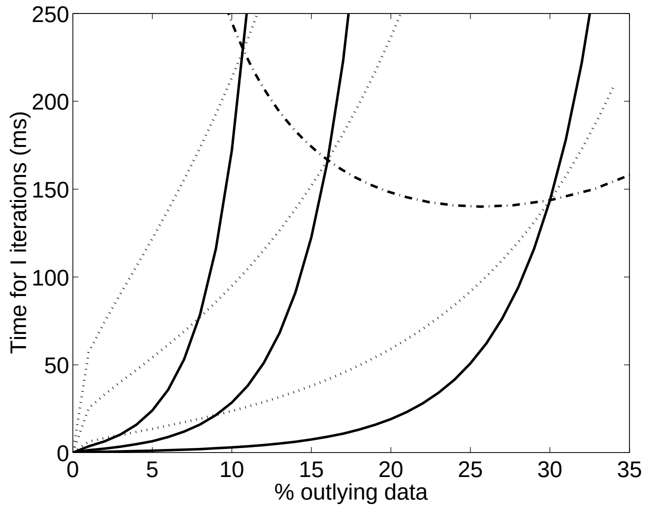


Figure 6.10: Pairs of curves showing the times required to complete the required random sampling trials as the percentage of invalid, outlying, data varies. The pairs of curves are for 30 (left), 20 (centre), and 10 (right) subparts, and in each pair the solid and dotted lines are from ART and DCT, respectively. Also shown is the locus of curve pair intersections.

their formulation, but one suited explicitly to perspective projection. Drummond and Cipolla's method was reviewed, and methods for solving the constraints system of branching kinematic trees and loopy kinematic graphs were given.

Comparative experiments on both methods showed, first, that the results for the pose updates are identical, notwithstanding the constraints being applied in very different ways. Both are done performing linear least squares in a single shot, and so their equivalence is expected.

Second, it was shown empirically that the computational kernel of DCT retains $\mathcal{O}(N)$ complexity in the number of subparts N, whereas ART rises from $\mathcal{O}(N)$ for low numbers, to $\mathcal{O}(N^2)$ for, say, $30 < N < 100$ before eventually succumbing to $\mathcal{O}(N^3)$. For large N, DCT's behaviour is characterized in the best case by N inversions of a fixed size 6×6 matrix and, at least for a non-branching chain, a $\mathcal{O}(N)$ recovery of Lagrange multipliers, whereas ART has to invert a $(6 + (N - 1))^2$ matrix. However, for tens of subparts the absolute difference in core cost is substantially outweighed by the commonly shared costs of model projection, image search, and so on.

Where DCT appears less satisfactory than ART is in the area of algorithmic robustness. One difficulty occurs when, rather than a simple chain of parts, there is branching into a tree structure. The description of the ART algorithm shows it to be, de facto, a simple tree-based method. But in DCT, as noted earlier, at a branch a subpart has more than two sets of motion constraints to satisfy, and the matrix to be solved in Eq. (6.33) no longer has a tridiagonal structure. Instead it acquires a block-symmetric structure dependent on the model's structure, but no general fast solution methods exists for the solution of such systems. Indeed, in [DC01], Drummond and Cipolla adopt a statistics propagation method to compute the Lagrange coefficients for each joint sequentially. Again, when there is insufficient information to solve for the initial pose update of a particular part, the tridiagonal method cannot work, unless regularisation is used.

A more general curiosity in DCT is that the *less* complicated the object's kinematics, the *more* computation has to be done to impose the constraints. This, and the issues raised in the previous paragraph, are outcomes of allowing the degrees of freedom to grow to their maximum and then having to prune them back when those freedoms are not required. While this extra effort is rather neatly tamed if the problem is a well-behaved chain of subparts, exceptions (such as kinematic loops) become hard to manage.

Hands are articulated objects where the motion of each part is highly constrained by the motion of neighbouring parts. These are not simple punctual articulated constraints, they also include dynamic constraints and the motion of each part is also influenced by parts that are not directly adjacent to it (see Section 2.3.7). It is difficult to code such constraints following DCT's model. If a data-driven dimensionality reduction model is to be used, it is more straightforard to code the constraints using minimal representations.

An attempt has been made to characterize the time requirements of both methods to complete trials of a random sampler. Here it does seem that ART has an advantage over DCT, but for both methods the time required for trials increases sharply with rising number of parts, suggesting that it is more prudent to improve the quality of measurement than to rely on random sampling to "clean up" afterwards.

The most significant aspect of Drummond and Cipolla's method is that it makes comparatively easy the switching on and off of constraints, after, and separately from, the expensive process of making image measurements. It appears to provide exactly the mechanism required to account for the motion of objects which make contact and later break apart. As contact approaches between two parts, tracking performance with and without the relevant constraints can be tested, to decide whether the motion was constrained or was still independent.

While the application of constraint hypothesis-and-test to rigid objects is straightforward, a more significant challenge is to apply it to separate articulated objects. The approach favoured by the conclusions of this chapter would be to track the joined articulated entities using ART, and employ DCT at the contact.

7

Regression-based hand pose estimation from multiple cameras

In this chapter an RVM-based learning method is developed for hand pose recovery. The method is based on that proposed by Agarwal and Triggs for whole body pose recovery. However, hand pose recovery appears a more challenging problem than whole body pose estimation due to the greater degree of actual occlusion, and the greater degree of apparent occlusion where finger bounding contours are lost. Further difficulties arises from the acyclic character of usual hand motions that tend to be fast and sudden in images. Also, no a priori positions can be assumed for hands. But we can use the facts that bare hands' textures are fairly uniform and their colour has a relatively small variance in comparison to clothing. The key development proposed here is a combination of multiple views. Such method allows the use a new modification of shape contexts for rotation invariance, reducing the number of required training samples for pose estimation. An experimental comparison of the pose recovery performance using single versus multiple views is reported for synthetic and real imagery. The effects of the number of image measurements and the number of training samples on performance are also taken into account for the comparison.

7.1 Introduction

As mentioned earlier in this thesis, two quite different approaches to the problem of pose estimation of articulated objects are apparent in the literature. The first, and more traditional, is the *generative* approach, in which an estimate of the pose is used to update the model that predicts the appearance, e.g. by projecting a 3D model into the image. Measurements of the deviation between prediction and reality are used to estimate the pose update. The tracking methods studied in chapters 4, 5 and 6 are generative trackers. Such trackers can achieve good qualitative pose estimates at high frame rate, but they need to rely on models that give a good approximation of the tracked object. Furthermore, these trackers need a good estimate of the initial state, and at any time, if its prediction is not a good match to the true state, tracking will fail.

The approach of *discriminative* algorithms has recently been more widely explored for articulated objects [AS02, AASK04, TSTC03, STTC03, Bra99, SVD03, AT04c]. The idea is to recover a direct, but non-physically based, mapping between a (robust) representation of appearance and the model parameters such as joint angles. Inter alia, the approach exploits, as Wu *et al*. [WLH01] note, the fact that the range of typically explored poses of hands is much smaller than the entire range.

As mentioned in Chapter 2, two main approaches to relating image measurements qualitatively to 3D poses: classification-based and mapping-based. The former is computationally expensive and can only output pose estimations that are in the training set. The second is fast and is able to output estimates in a continuous manifold of the parameters space. The key factor of mapping-based approaches is the model used to build the map.

In this chapter, a mapping-based approach is developed for the problem of hand pose estimation. This is based on a multivariate regression method that follows in part Agarwal and Triggs' work on whole body pose estimation. However, the hand pose recovery is in general a more difficult problem, not least because of the far greater degree of actual occlusion, and of "apparent occlusion" where finger bounding contours are lost. For this reason this chapter proposes an extension of the single view method to multiple cameras, an approach which Erol *et al*. [EBN$^+$05] points out has not been widely explored for this problem. An experimental comparison of single and multiple view performance is presented, taking into account variation in the number of image measurements and training samples needed.

A framework of the method described in this chapter is illustrated in Figure 7.1. Once the images are acquired, a pre-processing step is to extract image descriptors is performed on both the training and testing phase. This step, described in Section 7.2, produces a compact description of appearance, for

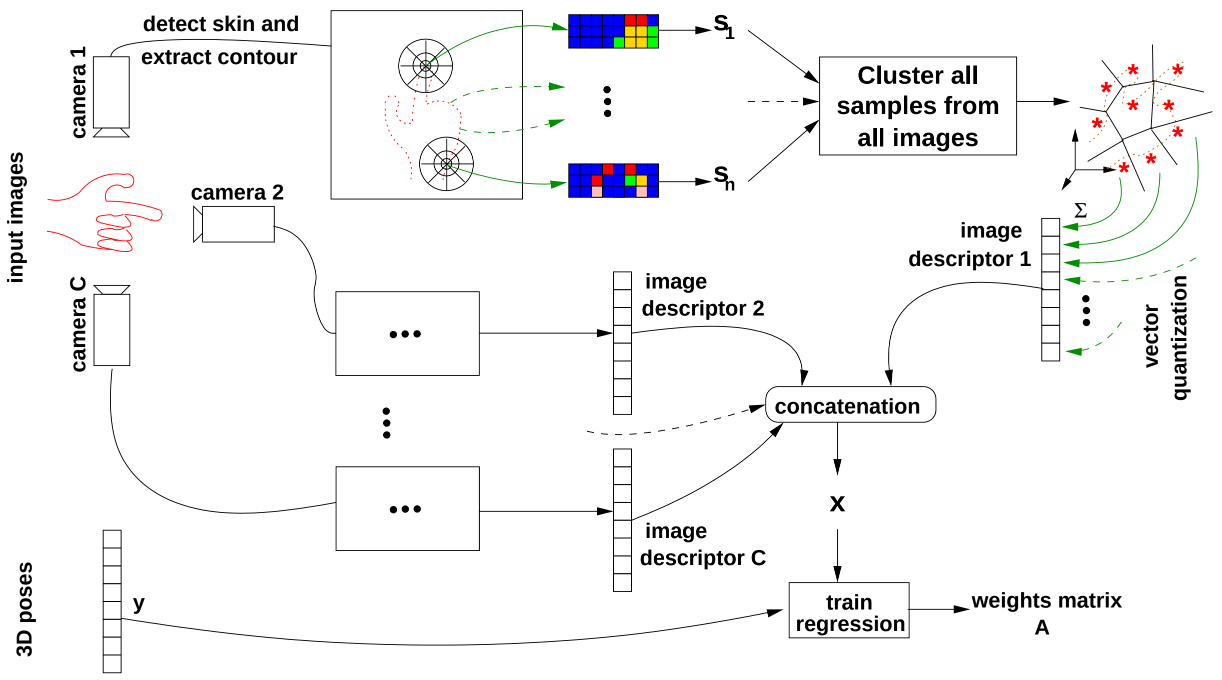


Figure 7.1: Framework showing the combination of the methods employed in the training phase. For each image, first the shape contexts of the silhouette contour are calculated. Next, vector quantisation is performed to produce a compact global image descriptor. The multiple view data are combined by concatenation and an RVM-based regressor is trained using the 3D poses.

which we use, as did Agarwal and Triggs, shape contexts descriptors. A novel modification is introduced for rotation invariance without loss of information about the shape. A compact global image descriptor is obtained though vector quantisation, and multiple view information is combined by concatenation. To train the regressor, it is necessary to gather a set of training pairs of multiple view images and 3D poses, section 7.3 describes the acquisition of training and testing data. The regression method used to learn the mapping between appearance and 3D poses is described in Section 7.4. Section 7.5 describes the experiments and results. A summary and the conclusions are drawn in Section 7.6.

7.2 Extracting Multiple View Image Descriptors

The initial step of the method (both in training and application phases) is the conversion of each image of a hand into a silhouette contour, and thence into a compact description using shape contexts. Because of the wide variation in scale and orientation of hands in imagery, it is important to incorporate invariance to these transformations within the context. A novel modification for rotation invariance is proposed. Its description is followed by the description of our method for combination of multiple view information.

7.2.1 Shape Contexts

Shape contexts, proposed by Belongie *et al*. [BMP02], are rich shape descriptors that are usualy computed for points on the silhouette contours. They encode local information about each point relative to its neighbours, and they can be made scale and rotation invariant.

Among modifications of shape contexts found in the literature, are (i) that of Ohashi and Shimodaira [OS03a, OS03b, FTR$^+$04], which is simpler than Belongie *et al*.'s method, but the final image descriptor is similar to that obtained after vector quantisation (as done by Agarwal and Triggs); and (ii) that of Thayananthan *et al*. [TSTC03], who used edge orientation and a continuity constraint for shape context matching, so neighbouring pixels in the image have to match neighbouring pixels in the shape contexts space), but the basic image descriptor is the same. The method presented here aims to obtain a robust global image descriptor, rather than to provide a match between sets of object points.

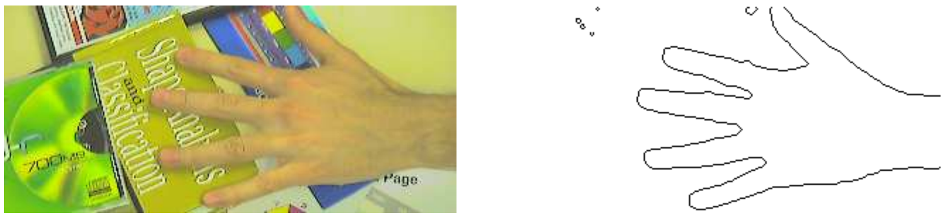
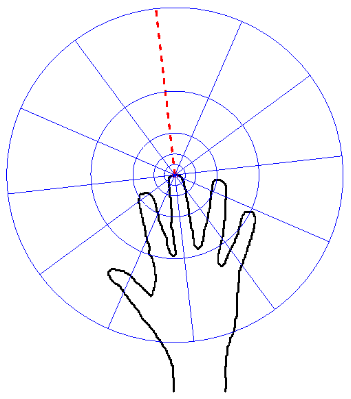
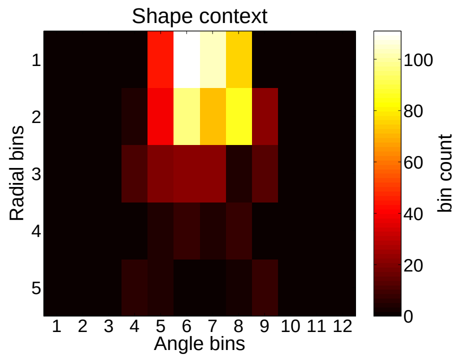

(a) (b)

Figure 7.2: An example of a hand image with a cluttered background (a) and its pixel-wise silhouette countour extracted by skin detection followed by edge detection (b).

Recovery of the silhouette of the hand, assumed un-gloved, is achieved using the histogram-based classifier presented in Chapter 3. This is applied to subsampled to 90×120 pixels to reduce computation cost. In our database, hands occupied about 20% ($\pm 6.2\%$ STD) of the image pixels. The shape contexts are computed only from positions on the silhouette contour, which is easily derived by edge detection in the resulting binary skin/not-skin image. Figure 7.2 shows the extraction of silhouette contour points from a hand image with clutter in the background. Note that only a few points are located outside the contour of the silhouette of the hand.

At any point on the contour, neighbouring contour points are accumulated in 60 bins arranged in


(a)



(b)

Figure 7.3: The shape context of a point at the tip of the index finger: (a) the regions taken into account for computation; (b) the obtained context. The column order in the histogram follows a counter-clockwise scan starting from fiducial "0°" dashed line, and the row order follows from outer to inner sectors.

log-polar fashion, five along the radial direction and twelve around the polar angle, spaced equally in log-distance and angle, respectively. To provide a first layer of scale invariance, the inner radius is set proportional to the mean μ of the distances between all the pairs of points in the silhouette. In our implementation, the inner radius is $\mu/8$, and radius increases in octaves to 2μ, typically covering all of the hand silhouette. Figure 7.3 illustrates the construction of a shape context for a point in the silhouette of the hand shown in Figure 7.2. The resulting 60-bin histogram is normalised, providing again for scale invariance. For image i the complete image description is generated as the set of n_i 60-bin histograms computed at n_i points along the silhouette contour.

Figure 7.4 shows the complete set of shape contexts for one hand silhouette. The n points in 60 dimensions are projected onto the first two principal components. Because the individual shape contexts computed at neighbouring points do not change drastically, and the primary principal components pick out a principal plane, it is possible, even in this feature space, to discern the characteristic four fingers and thumb.

7.2.2 Rotation Invariance

As mentioned before, the orientation of hands in natural actions can largely vary. This can be a challenge for discriminative methods if each different orientation requires a new set of training samples. The solution present in this chapter is to use rotation invariant image descriptors and multiple view. Therefore

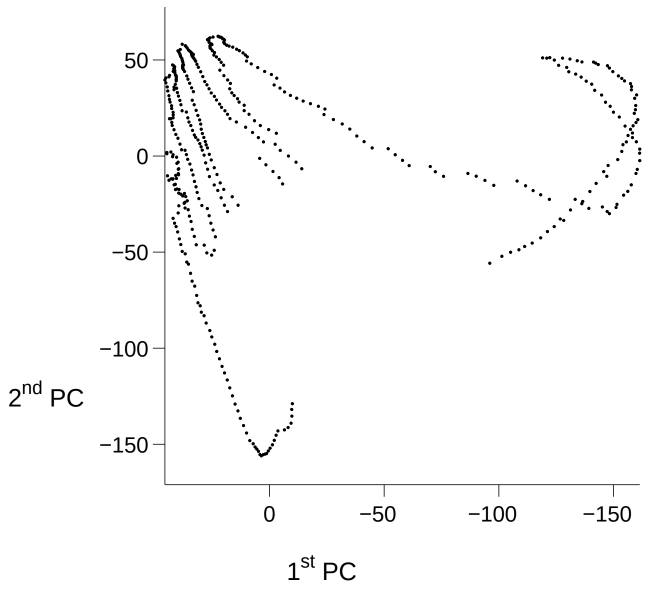


Figure 7.4: The 60-d shape context manifold obtained from a hand silhouette, visualised via a projection onto the first two principal components.

the pose estimation can be focused on internal parameters (joint angles), and the global pose can be estimated with the use of triangulation. Another benefit of rotation invariance is that it aleviates the need for the cameras setup consistence between training and testing phases.

Essentially, rotation invariance is achieved by orienting the fiducial line of the shape contexts according to some local feature of the shape. For instance, Figure 7.5 shows the shape contexts of a point at the tip of the middle finger for different images. In this example, the fiducial line is oriented by the tangent of the local contour. Note that the difference between the shape contexts of that point in a synthetic hand image (panel a), a real image (panel b), and the same image rotated (panel c) is very small in comprarison to the shape context of that point in a hand at a different grasping pose (panel d).

Belongie *et al*. ensured rotational invariance by aligning the fiducial “0°” line with the tangent to the silhouette contour at each point. While this works well if the contour is smooth, which in our experience requires either large images or using proper linked edge detection, the result in low resolution images, and using pixel contour points, was found to be noisy. A more robust alternative was found to be to use the geometric centre of the silhouette and to set the fiducial line to be orthogonal to the line from the centre to the contour point.

The rotation invariance of both tangent-based and centroid-based methods is obtained at the cost of

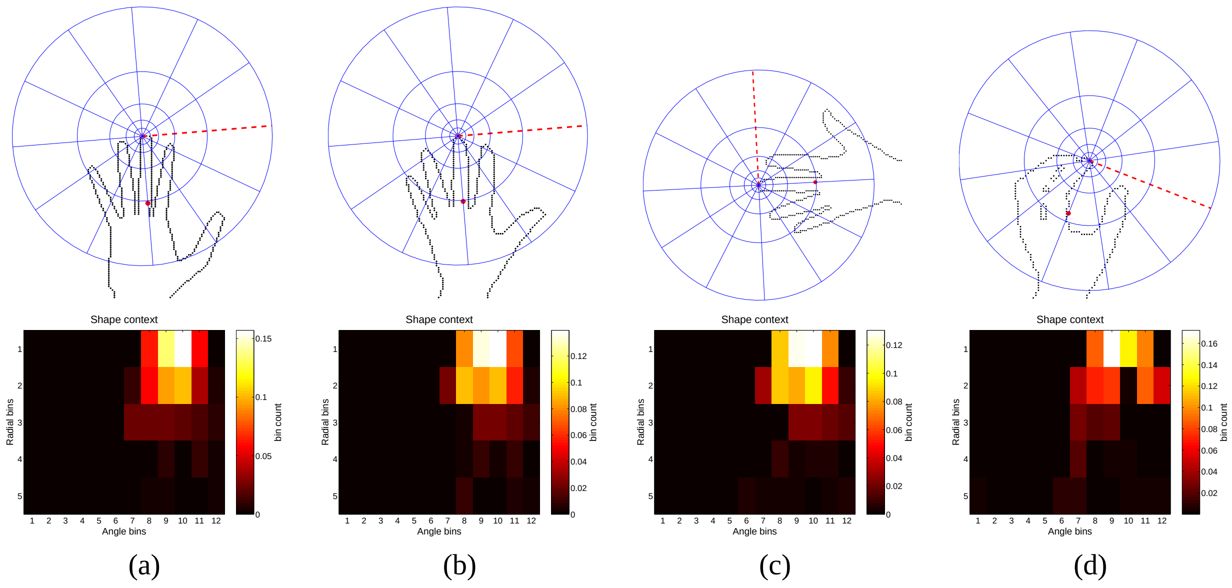


Figure 7.5: The silhouettes, log-polar bins, and the resulting shape context vectors obtained from the tip of the middle finger in four different images using tangent-oriented rotation invariant shape contexts. Context (a) is from a synthetic training image. Its similarity with real images is shown in panel (b), and (c) illustrates rotation invariance. The shape context from a different hand pose is shown in (d).

reducing the amount of global information about the shape of the silhouettes. This can be visualised in the projection of the 60 dimensional shape contexts space shown in Figure 7.6(b, d, and f) – note that these shape context manifolds do not present discerning hand characteristics – and through the nearest neighbour classification results shown in Figs. 7.7 and 7.8[1]. The solution that we adopt in this chapter is to orient the shape context with the axis that links the wrist to the tip of the hand. For simplicity, we assume that two points of the silhouette contour lie on the image borders, and these points are taken to be either side of the forearm. This is more robust than, for instance, using the principal axis obtained through PCA, as it can vary abruptly depending on the hand pose. The illustrations of Figure 7.6(g and h) and the results in Figs. 7.7 and 7.8 show that this maintains the discrimination power of non-rotation invariant shape contexts and adds robustness to planar rotations.

[1]The results in Figs. 7.7 and 7.8 were obtained through classification using global multiple view descriptors for the images, as described later, in Secs. 7.2.3 and 7.2.4.

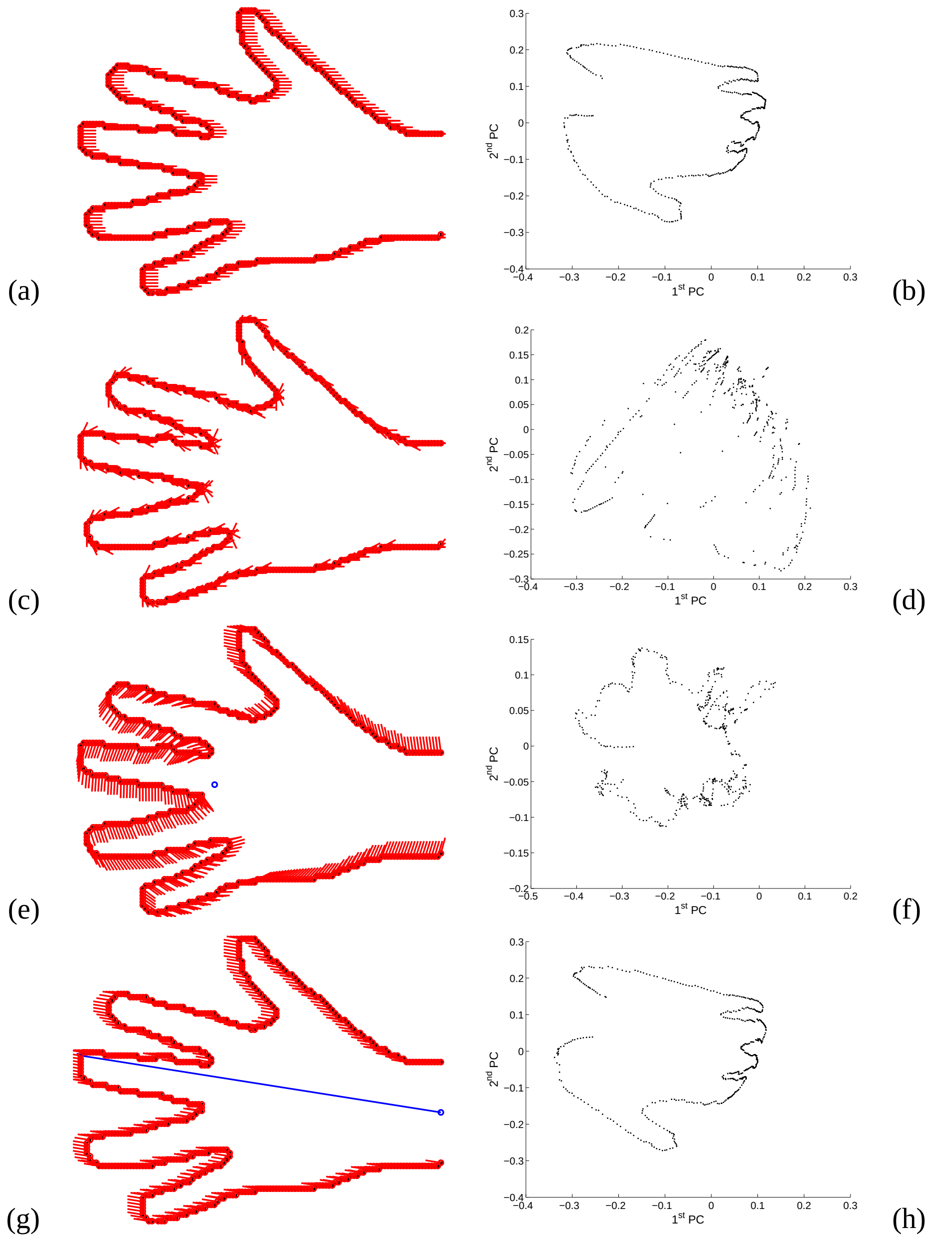


Figure 7.6: Methods to orient the shape contexts for rotation invariance (left) and their respective descriptors in the 60 dimensional space of shape contexts projected in 2 dimensions using PCA: (a and b) without rotation invariance, i.e., using a fixed orientation for the whole image; (c and d) using tangents obtained with a 3×3 window (as suggested in [BMP02]); (e and f) using the orientation orthogonal to the ray from the mass centre (indicated by the blue circle); (g and h) aligning the shape contexts with the hand's axis.

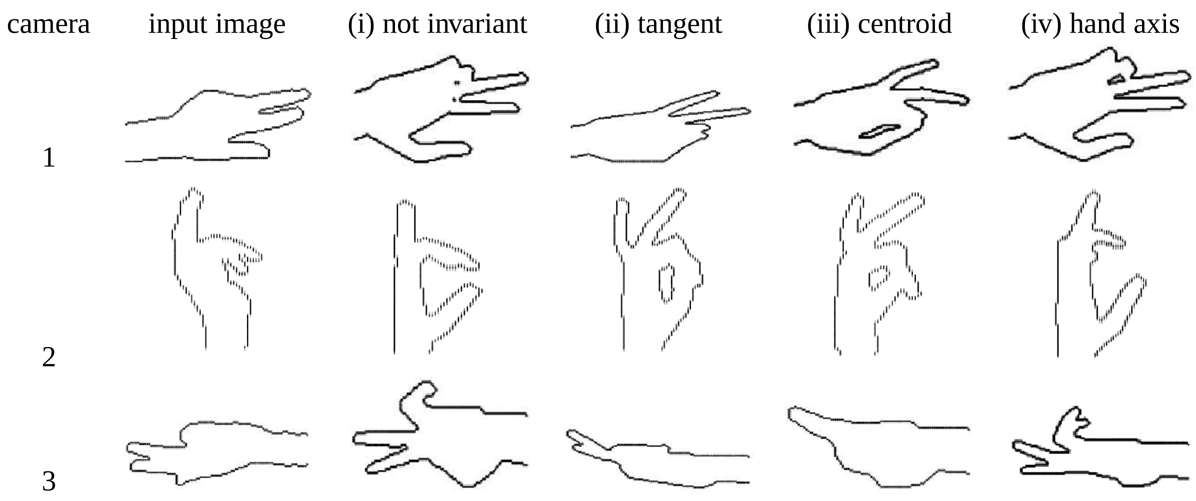


Figure 7.7: Nearest-neighbour classification results using multiple view descriptors obtained from the silhouettes on the first column, using (i) not invariant shape contexts and three methods of rotation invariance: (ii) tangent-based, (iii) centroid-oriented and (iv) hand axis-oriented shape contexts. Note that using the hand axis (iv), the ‘fingers ambiguity’ is avoided.

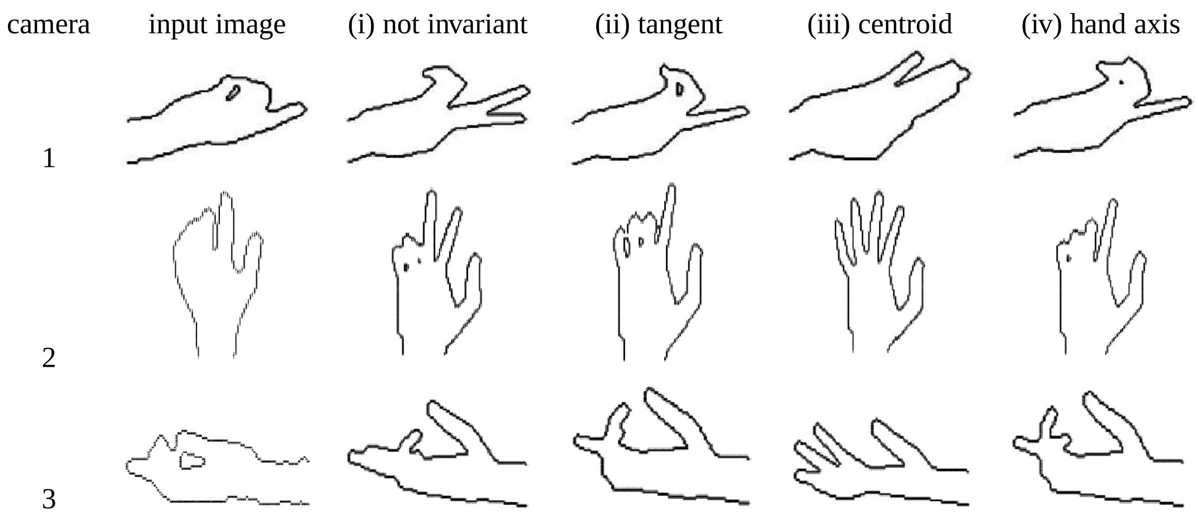


Figure 7.8: Same as Figure 7.7 for another set of testing images. Here, even though the hand is roughly aligned with the training data, tangent-based and hand axis-based rotation invariant shape contexts provided better results than the shape contexts without rotation invariance.

7.2.3 Encoding a Global Image Descriptor

In order to reduce the dimensionality needed to describe an image, a coding method is used. In this chapter, we adopt the same method as Agarwal and Triggs [AT04a]: vector quantisation. In the training phase, a codebook is created in a similar fashon to that of histogram, with bins being calculated using a clustering method. From a training set of I images all ($\sum_{i=1}^{I} n_i$) 60-d shape context vectors are clustered into K centroids using the K-means algorithm [DH73]. Each individual shape context j in image i becomes re-expressed as a K-dimensional vector $\mathbf{x}_{ij}$ with $K-1$ zero elements and a single unit element. In both training and application phases, the complete image descriptor $\mathbf{x}_i$ is generated by summing these and normalising by n_i

$$\mathbf{x}_i = \sum_{j=1}^{n_i} \mathbf{x}_{ij} \,/ n_i \; .$$

To soften the effects of spatial quantisation, the descriptors are built allowing context vectors to vote with Gaussian weights into the few centres nearest to them [AT04a].

Figure 7.9 shows the centroids obtained from 546 training images with a total of 128659 shape context descriptors.

7.2.4 Combining Multiple View Information

Some possibilities have been considered to combine multiple view information. In [UMKR96], the view which is most perpendicular to the hand palm is selected and the other are discarded. However, in many cases it is difficult to estimate the orientation of the hand if each view is analised individually.

The low level approach is to group all the shape contexts from all the images together before performing clustering to build the codebook. The problem of this approach is that the improvement obtained by using multiple views may not be very significant, as one set of measurements can be associated to more than one global orientation.

An alternative is to estimate the pose from each view individually and combine the results at a high level using, for example, a graphical model [Mur02]. If global pose parameters can be estimated using triangulation and if regressors can be trained with a set that is comprehensive in terms of the internal pose parameters and orientations, then the same regressor could be applied for all the cameras, and the setup

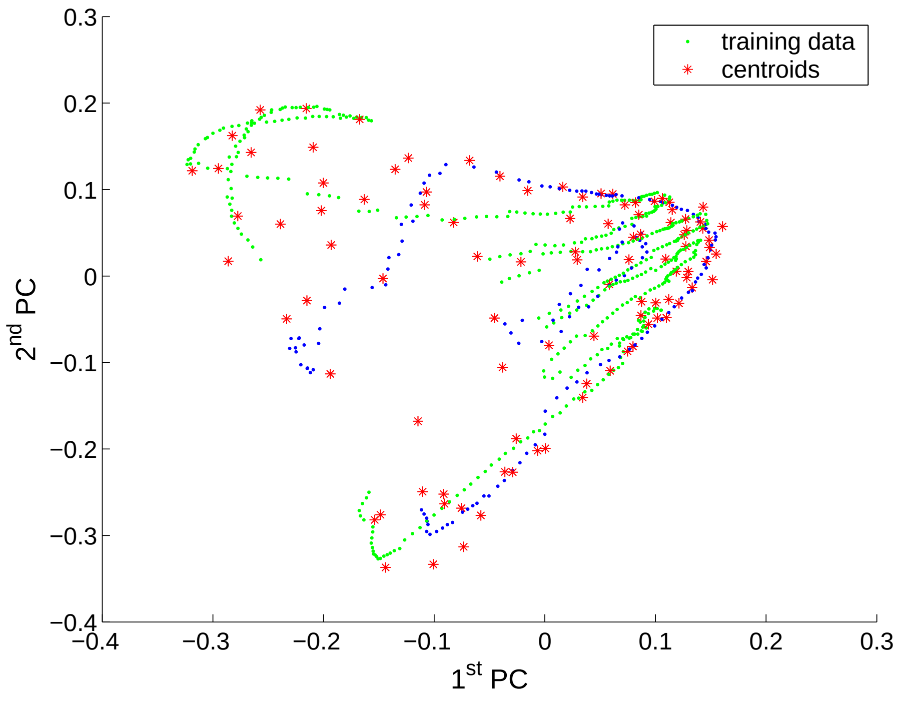


Figure 7.9: K-means of all the shape contexts of a particular training set are shown as stars. The shape contexts of two samples, one with the fingers stretched and another with the hand in a fist pose, are plotted to aid visualisation (differentiated here by green and blue dots).

of cameras would not need to be the same as in training. However, as discussed later, it is not realistic to use very large training sets.

In [HSS02], it was demonstrated that the discrimination power is proportional to a measure of the complexity of the curvature of the contour. Thus, for each view, the matching score is weighted by this measure and they are added up to the final score. In the approach proposed here, it is not necessary to employ a shape complexity measure to weight each view: the regressor does that implicitly if linear kernels are used. For that, the information is combined in an intermediate level, by generating vectors $\mathbf{x}$ for each camera individually and concatenating them into a higher dimensional vector that describes the current measurements from all the cameras. The regressor is then trained using these concatenated vectors, as illustrated in figure 7.1. The advantage of this approach is that the images drescriptor encode information from all the views separately, reducing the number of traning data needed to use the additional pose constraints that multiple views offer. The drawback is the need for an agreement of the cameras poses between training and application phases, though the descriptor is robust to rotations on the cameras planes and to variation in scale, i.e., the proximity between the cameras and the hand can vary, as well as the internal camera parameters (e.g. focal length).

In the present implementation, K is set to 30 for each view, so the image descriptors of a three-views data set are represented in a 90-dimensional manifold as shown in Figure 7.10. For comparison, 90-d single view image descriptors were also obtained from the same training set by using $K = 90$.

Note that, for multiple views, the first and second principal components are roughly aligned with the variation in θ_Z, and with the overall degree of flexion of the fingers, respectively, where θ_Z is the rotation around the forearm axis, as detailed in Sec. 7.3. This hints that this dataset of hand appearances can roughly be represented with two degrees of freedom. This effect cannot be observed for single view descriptors $\mathbf{x}$. In that case, the manifold seems to need at least three dimentions to show more separability between hands poses.

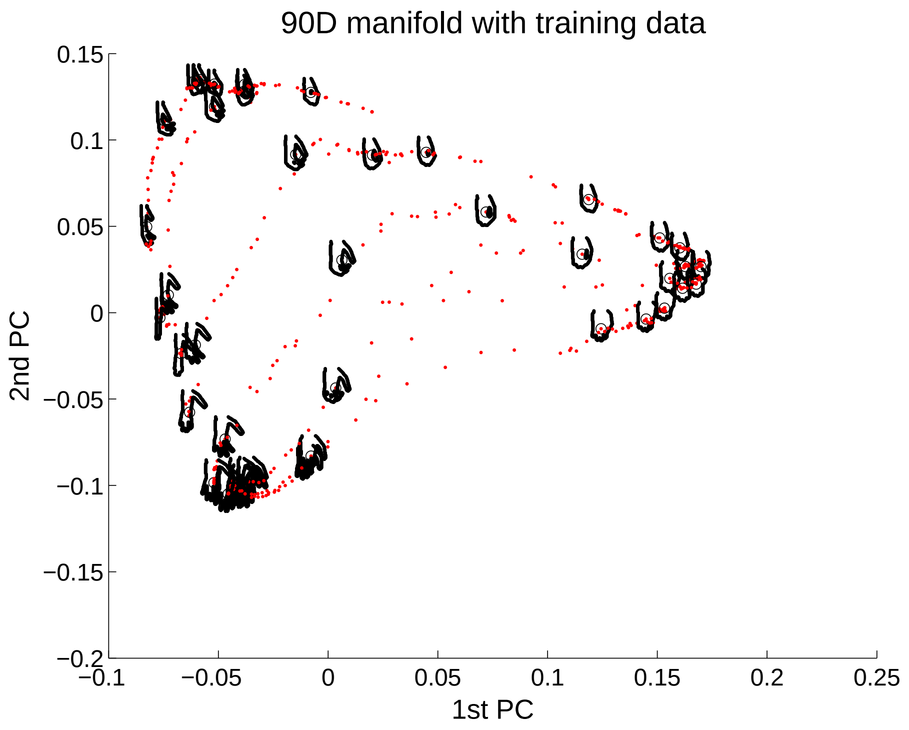
7.3 Obtaining and testing the training data

So far, we have described the generation of a possibly multiview image descriptor $\mathbf{x}$. An essential input to the later regression process is, of course, the association of each $\mathbf{x}$ with a set of known joint angles $\mathbf{y}$.

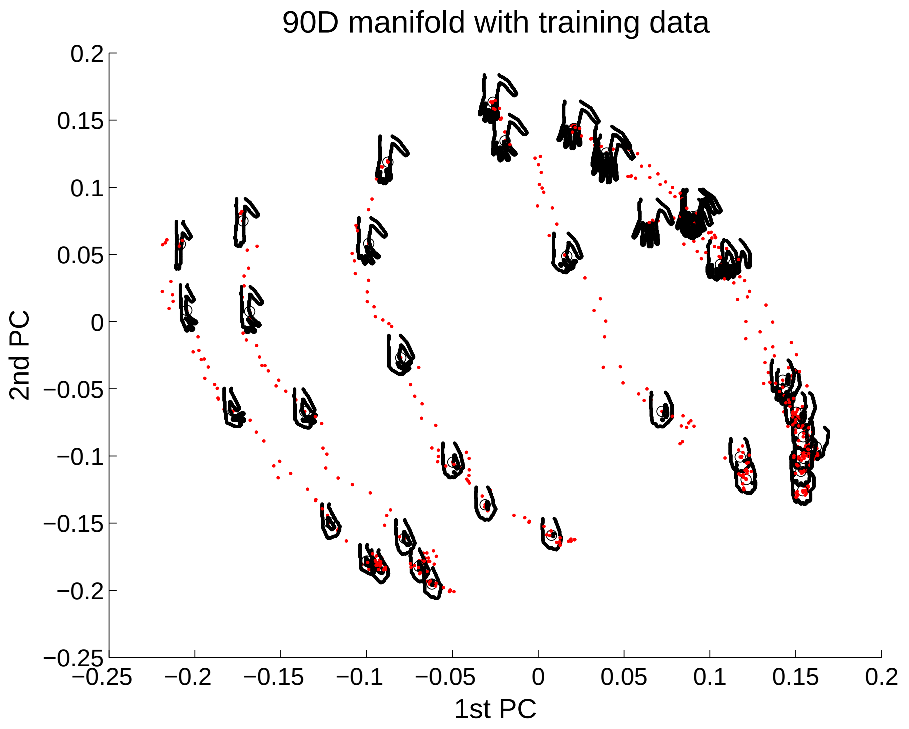
For this chapter we use a subset of the hand trajectory database prepared by Thayananthan and Stenger [STTC03] at Cambridge University's Department of Engineering, using an Immersion Corporation's CyberGlove. The database contains the trajectory of 20 joint angles of the hand of two users. The 28 DOF hand model described in Section 5.5 was used to synthesize images at the poses of this database, so $\mathbf{y} \in \mathbb{R}^{28}$. Since the CyberGlove used does not include sensors between the forearm and the palm, the two degrees of freedom of this joint were set to constant values.

7.3.1 Training Sets

We demonstrate experiments using two training sets. The first, dubbed *open-close*, consists of a trajectory that starts with all the fingers stretched after which a grasping gesture is performed in 78 frames. The glove used to generate this data did not have a global position and orientation sensor, so the trajectory was duplicated seven times for 15° spaced values $0^\circ \leq \theta_Z \leq 90^\circ$, giving a total of 546 poses, some of which are shown in Figure 7.11. For desktop tasks the variation of the other orientation parameters (θ_X and θ_Y) is usually small enough to enable us to rely on the invariance properties of the modified shape contexts.



(a)



(b)

Figure 7.10: 90-d manifolds of $\mathbf{x}$ vectors obtained from a training data visualised using projection onto the first two principal components. The silhouettes of the hand at some key poses (6 poses for each θ_Z angle) are shown in their location in the manifold. Panel (a) shows the manifold for single view $\mathbf{x}$ vectors, and panel (b) shows the same for multiple view $\mathbf{x}$.

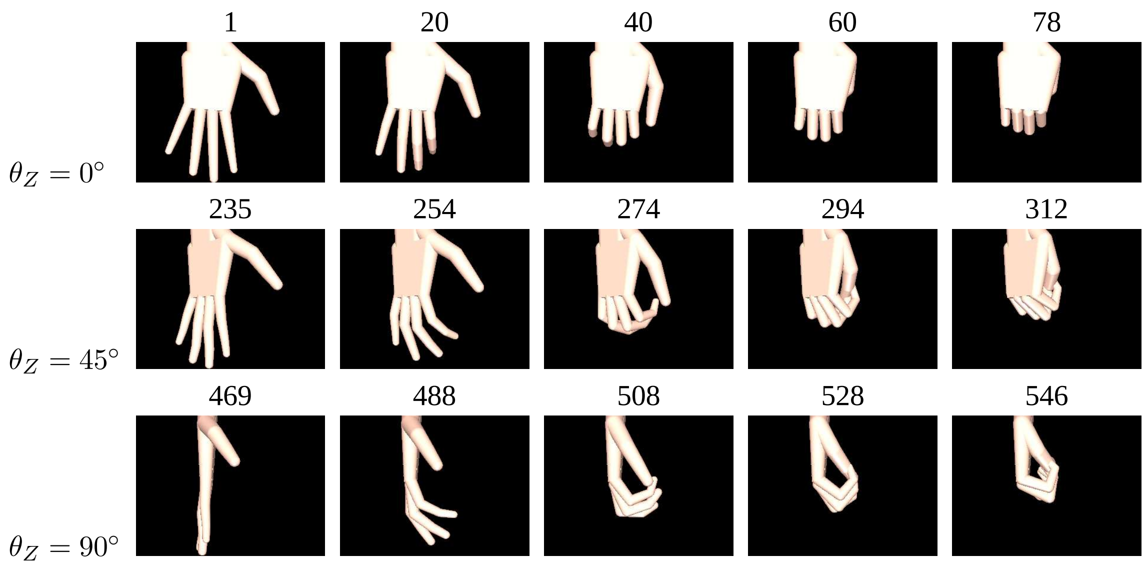


Figure 7.11: Images rendered in camera 2 (top view) for 3 of 7 angles and 5 of 78 grasp strengths from the *open-close* training set. The number above each image is its index in the set.

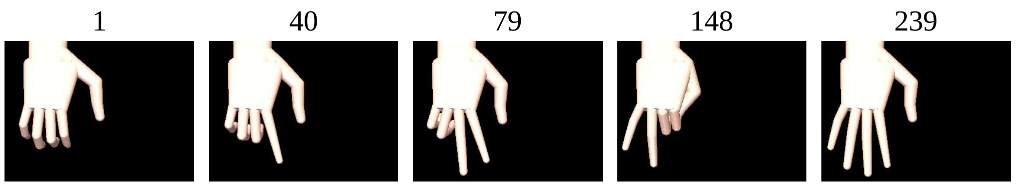
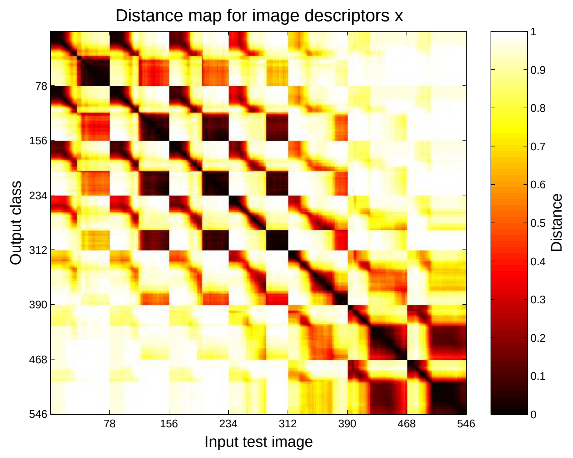


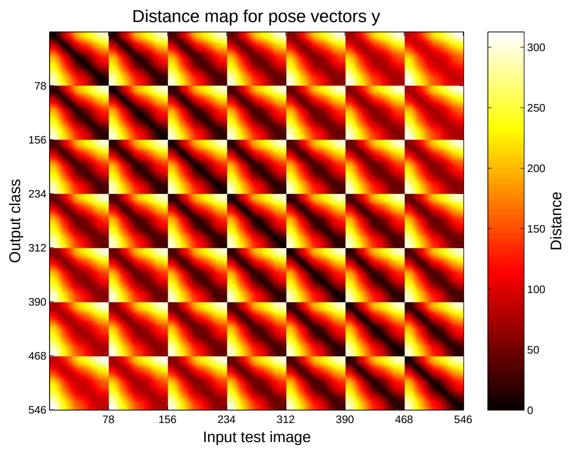
Figure 7.12: Some images from the *complex* training set rendered in camera 2 for $\theta_Z = 0^\circ$.

A more accurate global orientation can be obtained by triangulation when multiple views are used. For a fair comparison between single and multiple view, θ_X and θ_Y are not taken into account. For multi-camera application, the hands were rendered from three different viewpoints. These viewpoints have the same camera calibration parameters as those used on the acquisition of real images for recognition.

The second training set, dubbed *complex*, has fingers moving individually, as shown in Figure 7.12. This sequence has 239 internal poses that, as before, are reproduced for 15° spaced values $0^\circ \leq \theta_Z \leq 90^\circ$, giving a total of 1673 three-dimensional poses.



(a)



(b)

Figure 7.13: The distances between vectors describing the *open-close* training set of 546 images for a single view. (a) Distances of the image descriptors $\mathbf{x}$ and (b) distances of the 28-d poses vectors $\mathbf{y}$. Note the repetition of a pattern in this map, because at each 78 samples the same trajectory is reproduced, with an addition of 15° to θ_Z.

7.3.2 Assessing the training set

Figure 7.13(a) shows the dissimilarity $\mathtt{D}$ between the $\mathbf{x}$ image descriptors in matrix form. The natural dissimilarity measure for histograms is the χ^2 test statistic [BMP02]:

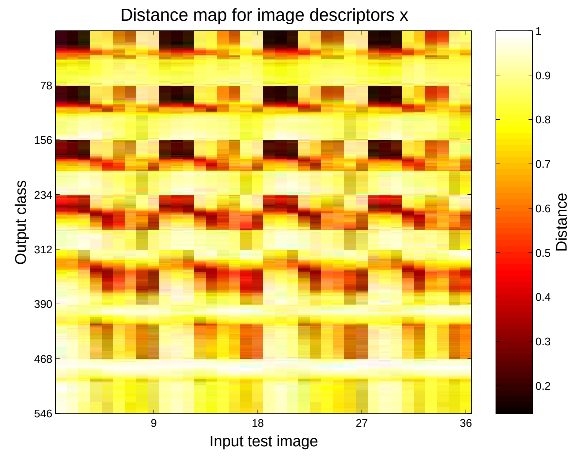
$$\mathtt{D}_{i,j} = \mathtt{D}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2} \sum_{k=1}^{K} \frac{[\mathbf{x}_i(k) - \mathbf{x}_j(k)]^2}{\mathbf{x}_i(k) + \mathbf{x}_j(k)} \,, \tag{7.1}$$

where i and j are sample (i.e., image) indexes and k is the descriptors' dimension index[2]. The dissimilarity between 28 dimensional $\mathbf{y}$ vectors of pose is shown in Figure 7.13(b), obtained using the Euclidean distance. The reduced number of low values in the off-diagonal elements of $\mathtt{D}(\mathbf{x}_i, \mathbf{x}_j)$ shows the discrimination power of the image descriptor. Due to the similarity of the silhouettes, there remains more confusion amongst $\mathbf{x}$ vectors as the fingers are closed up.

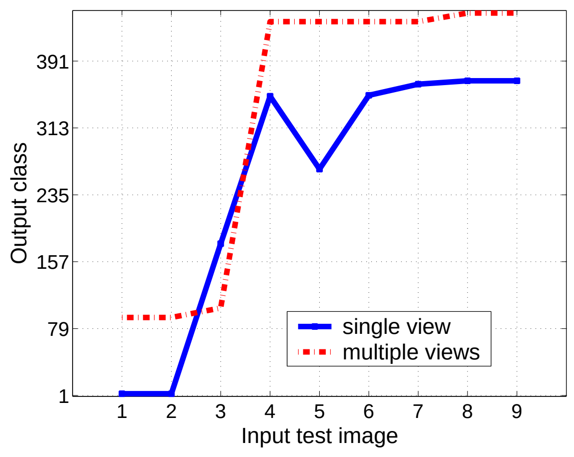
In order to assess the discriminatory power of the image descriptors $\mathbf{x}$, a nearest neighbour classification experiment was performed with 36 hand images – 9 hand poses taken from 4 orientations. The results, shown in Figures 7.14 and 7.15, suggest that the image descriptor is robust enough to provide a good qualitative description of the hand shape from images that are not in the training set, even though the hand model is not accurate. Figure 7.15 also shows that the use of multiple views can improve the

[2]Since shape contexts are histograms, this measure was also used previously in the criterion function of the clustering algorithm for vector quantisation – Section 7.2.3.

nearest neighbour classification result.



(a)



(b)

Figure 7.14: (a) Distance map between the 36 testing samples (9 images with 4 different rotations) and the 546 training samples and a single view. The repetition in the pattern at each 9 samples confirms rotation invariance. (b) Nearest-neighbour classification results for nine samples of the same orientation.

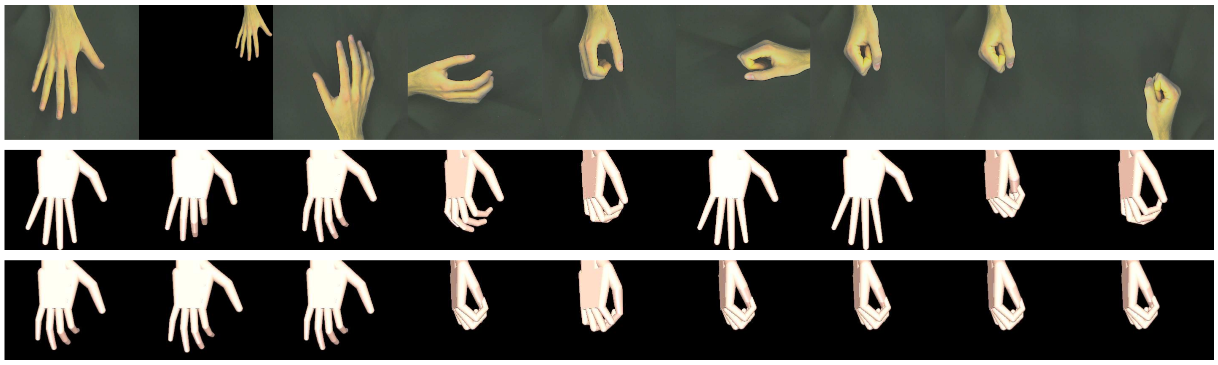


Figure 7.15: 1^{st} row: sample images from camera 2 with modifications in orientation, translation and scale. The nearest-neighbour classification results using single view with scale and rotation invariant descriptors are shown in the 2^{nd} row. The 3^{rd} row shows the same, using multiple views.

An improvement is expected to be achieved with regression because a neighbourhood of training samples is taken into account in the parameters space, whereas nearest-neighbour simply returns the sample with highest score. Another obvious advantage is that the formulation of a regression-based method and its sparsity make it much faster than nearest neighbour classification.

7.4 Learning to Relate Descriptors to 3D Poses

To relate the image descriptors $\mathbf{x}_i$ to the 3D joint angles and pose settings $\mathbf{y}_i$, Agarwal and Triggs [AT04a] proposed the use of a regression method that learns the relation between I pairs of vectors $(\mathbf{x}_i, \mathbf{y}_i)$ by estimating the coefficients or weights of a linear combination of basis functions ϕ_k. The problem is described as:

$$\mathbf{y}_i = \sum_{k=1}^{p} \mathbf{a}_k \phi_k(\mathbf{x}_i) + \boldsymbol{\epsilon} \equiv \mathtt{A}\mathbf{f}(\mathbf{x}_i) + \boldsymbol{\epsilon} \tag{7.2}$$

where $\boldsymbol{\epsilon}$ is a residual error vector, $\mathbf{y}_i \in \mathbb{R}^m$ ($i = 1, 2, \cdots, I$), and $\mathbf{a}_k \in \mathbb{R}^m$ ($k = 1, 2, \cdots, p$). For compactness, the weight vectors can be gathered into an $m \times p$ matrix $\mathtt{A} \equiv (\mathbf{a}_1 \; \mathbf{a}_2 \; \cdots \; \mathbf{a}_p)$ and the basis functions into a $\mathbb{R}^p$-valued function $\mathbf{f}(\mathbf{x}) = (\phi_1(\mathbf{x}) \; \phi_2(\mathbf{x}) \; \cdots \phi_p(\mathbf{x}))^\top$. As discussed later, $p = K$ for linear kernel, and $p = I$ for Gaussian kernel. In order to estimate the bias of the samples in the state space, one can use $\mathbf{f}(\mathbf{x}) = (1 \; \phi_1(\mathbf{x}) \; \phi_2(\mathbf{x}) \; \cdots \phi_p(\mathbf{x}))^\top$ and add a weight parameter to be estimated, but this is unnecessary if the data is standardized to have zero mean and unit standard deviation.

For I training pairs, the estimation problem takes this form: estimate $\mathtt{A}$ such that

$$\mathtt{A} = \arg\min_{\mathtt{A}} \left\{ \sum_{i=1}^{I} ||\mathtt{A}\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i||^2 + R(\mathtt{A}) \right\} \tag{7.3}$$

where $R(\cdot)$ is a regulariser on $\mathtt{A}$. Gathering the training vectors into an $m \times I$ matrix $\mathtt{Y} \equiv (\mathbf{y}_1 \; \mathbf{y}_2 \; \cdots \; \mathbf{y}_I)$ and a $p \times I$ feature matrix $\mathtt{F} \equiv (\mathbf{f}(\mathbf{x}_1) \; \mathbf{f}(\mathbf{x}_2) \; \cdots \; \mathbf{f}(\mathbf{x}_I))$, equation (7.3) can be rewritten as:

$$\mathtt{A} = \arg\min_{\mathtt{A}} \left\{ ||\mathtt{AF} - \mathtt{Y}||^2 + R(\mathtt{A}) \right\} \tag{7.4}$$

7.4.1 Regression with Relevance Vector Machines

For unidimensional signals y, Tipping [Tip01] proposed the use of Relevance Vector Machine (RVM), a method based on sparse Bayesian learning to estimate efficiently an $\mathtt{A}_{(1 \times p)}$ with large sparsity. Each weight parameter is associated with an independent noise model α and there is a prior for α parameters (hyperpriors), which are modelled as Gamma functions, so they have a high probability near zero, enforcing sparsity in the estimate of the weights. Upon minimization, the regularization parameters push the weights a of the less relevant basis functions to zero, thus producing a sparce model. This sparsity can save computational time and space.

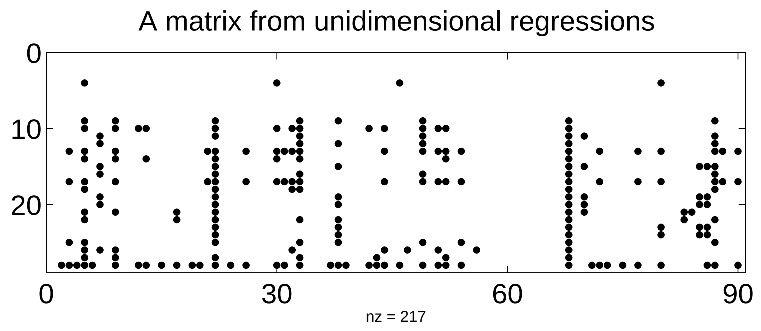


Figure 7.16: Map of non-zero elements of matrix $\mathtt{A}_{(m\times p)}$ resulting from RVM regression of individual parameters separately, using a threshold to select an average of 10 relevance vectors per DOF.

A straightforward adaptation of this method for multidimensional state vectors $\mathbf{Y}$ can be achieved by regressing input vectors $\mathbf{x}$ against each of the m individual elements y_j of $\mathbf{y}$ separately and concatenating the obtained row vectors of weights into matrix $\mathtt{A}_{(m\times p)}$.

An experiment with the *open-close* data set was performed using $K = 90$ (i.e. $K = 30$ for each view) and linear kernel functions ($\mathbf{f}(\mathbf{x}) = \mathbf{x}$). During the optimization, weight values a that were smaller than a threshold $\mathcal{T}_a$ where set to zero. $\mathcal{T}_a$ was set to a value that give an average of 10 non-zero weights for each dimension y_j. The resulting non-zero elements of $\mathtt{A}$ matrix are represented in Figure 7.16. The application of the obtained regressor on samples from the training set resulted on the mean absolute error[3] of $3.1°$, and mean standard deviation of $2.3°$. The worst result was obtained with the interphalangeal[4] joint of the thumb, which is occluded in many of the training images. For that joint angle, the average error was $9.5°$ and the standard deviation was $6.9°$.

A problem with regressing parameters independently is that noisy data potentially provide impossible output poses. For example, a regressor trained to recover 3D pose of walking humans might output poses having both legs to the front.

7.4.2 Agarwal and Triggs' Regression Method

The pose of each DOF of the hand in natural motion without external forces is clearly not independent from the pose of the others. In [AT06b], Agarwal and Triggs describe an adaptation of Tipping's method that estimate the whole matrix $\mathtt{A}$ in a single process, creating a linear combination of relations with

[3]Computed by $\sum_i^I |\mathtt{A}\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i|/I$

[4]For hand joints nomenclature, see Figure 1.1 and [Stu92].

multi-dimensional output. This regressor is estimated by direct optimisation of the weights keeping the hyperprior parameters fixed.

The first step of this algorithm is to initialise $\mathtt{A}$ with *ridge regression*. The regulariser is chosen to be $R(\mathtt{A}) \equiv \lambda||\mathtt{A}||^2$, where λ is a regularisation parameter. The problem can be described as the minimisation of

$$||\mathtt{A}\tilde{\mathtt{F}} - \tilde{\mathtt{Y}}||^2 = ||\mathtt{AF} - \mathtt{Y}||^2 + \lambda||\mathtt{A}||^2 \ , \tag{7.5}$$

where $\tilde{\mathtt{F}} \equiv (\mathtt{F}\ \lambda\mathtt{I})$ and $\tilde{\mathtt{Y}} \equiv (\mathtt{Y}\ \mathtt{0})$. $\mathtt{A}$ can be estimated by solving the linear system $\mathtt{A}\tilde{\mathtt{F}} = \tilde{\mathtt{Y}}$ in least squares, i.e., $\mathtt{A} \leftarrow [(\tilde{\mathtt{F}}\tilde{\mathtt{F}}^\top)^{-1}\tilde{\mathtt{F}}\tilde{\mathtt{Y}}^\top]^\top$. Ridge solutions are not equivariant under scaling of inputs, so both $\mathbf{x}$ and $\mathbf{y}$ vectors are scaled to have zero mean and unit variance before solving. The mean and standard deviation of the components of $\mathbf{x}$ and $\mathbf{y}$ are kept for application on testing data.

The next step is to apply a modification of Tipping's RVM regression method. Instead of modelling $p(\alpha)$ (the hyperpriors of the weights a) as Gamma functions, Agarwal and Triggs use $p(a) \approx ||a||^{-\nu}$, which is a simple case of Gamma function with constant parameters. Their method is a maximum a priori type and directly optimises the weight parameters a, while Tipping's is a type-II maximum likelihood approach that integrates out the parameters and optimizes the hyperparameters.

In order to reduce the risk of premature trapping of weight parameters to zero and overfitting, Agarwal and Triggs proposed to successively approximate the penalty terms with quadratic "bridges". Therefore, with a an element of $\mathtt{A}$, the regularisers $R(a) = \nu \log ||a||$ are approximated by $\frac{\nu}{2}(||a||/a_{scale})^2 + const$, where a_{scale} is a constant that is updated at each iteration[5]. The approximation has the same gradient as the original function at $a = a_{scale}$, and if $const$ is set to $\nu(log||a_{scale}|| - \frac{1}{2})$, the same values at $a = a_{scale}$, as shown in Figure 7.17.

Agarwal and Triggs proposed the use of column-wise set of priors in the regulariser $R(\mathtt{A})$: with $\mathbf{a}$ a column of $\mathtt{A}$, $R(\mathbf{a}) \approx \frac{\nu}{2}(||\mathbf{a}||/a_{scale})^2 + const$, implying that the estimated matrix $\mathtt{A}$ has some columns tending to zero as the method iterates, $\mathbf{a}_k \approx \mathbf{0}$. Depending on the kernel function used, two different aspects of cost reduction for pose estimation can be achieved:

- If *linear basis functions* are used, i.e., $\mathbf{f}(\mathbf{x}) = \mathbf{x}$, the zero vectors $\mathbf{a}_k$ indicate which components

[5] In [AT04a] and [AT04c], the authors have missed the division by two in the approximation of the regularizers $R(a)$, but this has been corrected in [AT04b] and [AT06b].

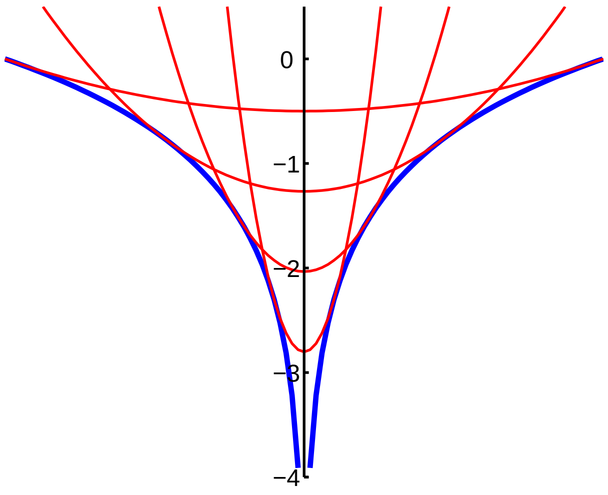


Figure 7.17: Quadratic "bridges" approximations to the $\nu \log ||a||$ regularisers, introduced by Agarwal and Triggs [AT04a] to prevent weight parameters from prematurely becoming trapped at zero in the minimization process, which can cause overfitting.

of vectors $\mathbf{x}$ can be removed without compromising the regression result. Therefore, RVM can be used as a *feature selection* method, resulting in a reduction in the number of shape descriptors needed.

- Alternatively, *kernel basis functions* can be used. They are expressed by $\phi_i(\mathbf{x}) = \mathcal{K}(\mathbf{x}, \mathbf{x}_i)$, making $\mathbf{f}(\mathbf{x}) = [\mathcal{K}(\mathbf{x}, \mathbf{x}_1), \mathcal{K}(\mathbf{x}, \mathbf{x}_2), \cdots, \mathcal{K}(\mathbf{x}, \mathbf{x}_n)]^\top$, where $\mathcal{K}(\mathbf{x}, \mathbf{x}_i)$ is a function that relates $\mathbf{x}$ with the training sample $\mathbf{x}_i$. For example (as used in this chapter), one can use a Gaussian kernel $\mathcal{K}(\mathbf{x}, \mathbf{x}_i) = e^{\beta||\mathbf{x}-\mathbf{x}_i||^2}$, with β estimated from the scatter matrix of the training data. In this case, the column-wise sparsity of $\mathtt{A}$ acts as a method to *select relevant training samples*.

The estimation of $\mathtt{A}$ is then performed in a similar fashion as to Equation 7.5, by iteratively solving the linear system:

$$\mathtt{A}(\mathtt{F}\ \mathtt{R}) = (\mathtt{Y}\ \mathtt{0}) \tag{7.6}$$

where $\mathtt{0}$ is a $m \times p$ matrix of zeros and $\mathtt{R}$ is a $p \times p$ matrix whose rows are defined by ν / a_{scale}, and a_{scale} is the norm $||\mathbf{a}||$ of each column vector of $\mathtt{A}$ from the previous iteration. To reinforce sparsity, the columns of $\mathtt{A}$ whose norms are small are set to zero. This process is repeated until $\mathtt{A}$ converges.

Figure 7.18 shows the $\mathtt{A}$ matrix obtained by this method using linear kernel functions in the *open-close* data set, with multiple view 90-d descriptors $\mathbf{x}$. The threshold $\mathcal{T}_{\mathbf{a}}$ on the norm of the column

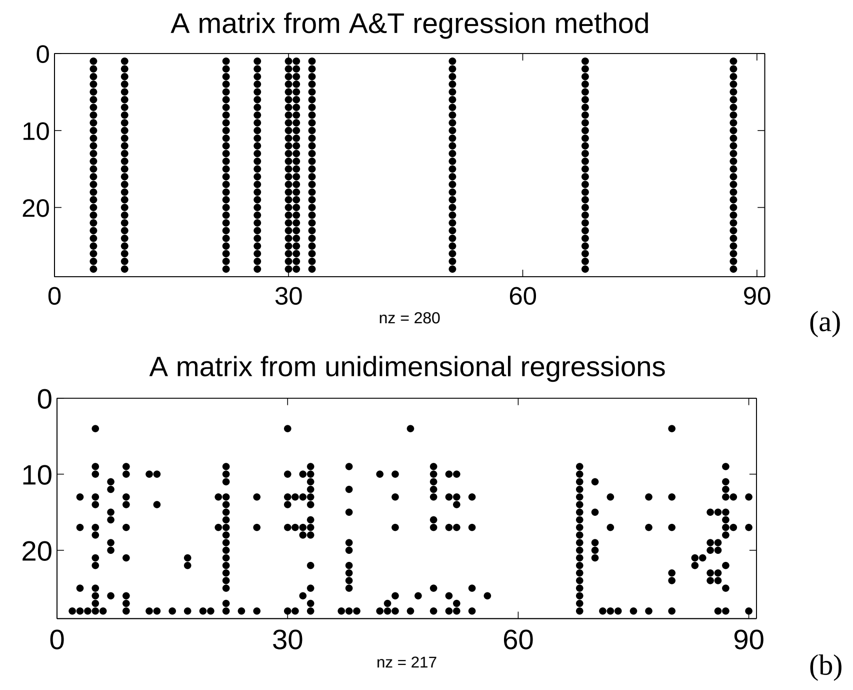


Figure 7.18: Map of non-zero elements of matrix $\mathtt{A}_{(m\times p)}$ resulting from linear regression using Agarwal and Triggs' method, selecting 10 relevance vectors in total (a). The result obtained with RVM of individual parameters shown in Figure 7.16 is reproduced in (b) to show that some agreement between the two methods is obtained on the selection of columns of $\mathtt{A}$.

vectors $||\mathbf{a}||$ was tuned to select 10 relevant features, resulting in the selection of 5 features from camera 1 (side view), 3 features from camera 2 (top view), and 2 features from camera 3 (another side view). For samples in the training set, regression with this matrix resulted in a mean absolute error of $2.7°$, and a mean standard deviation of $2.0°$. The worst average error and standard deviation were $11.8°$ and $8.2°$ respectively, both for the interphalangeal joint of the thumb. This represents an improvement in comparison to the results obtained by regressing the DOFs individually. As discussed later, the column-wise sparsity of matrix $\mathtt{A}$, allows the application of this method for feature selection or sample selection. It is interesting to note that many of the vectors selected using Tipping's method coincide with rows selected by Agarwal and Trigg's method, confirming the common theoretical basis of both.

7.4.3 Applying the Regressor with Feature and Samples Selection

It has been observed that Gaussian kernel functions can provide better results at the expense of being slower than linear kernel functions [AT04a]. Indeed, the results showed later suggest that linear functions are less stable to noise than Gaussian kernel functions. The alternative proposed here is to combine both by first reducing the dimensionality of the image descriptors $\mathbf{x}$ with feature selection and then using regression with Gaussian kernel functions to select the most relevant samples. Since the dimension of the vectors $\mathbf{x}$ is reduced in the first stage, all the distance calculations required to compute $\mathbf{f}(\mathbf{x})$ with Gaussian kernels is sped up.

After the training process has been performed to obtain matrix $\mathtt{A}$ and the sets of selected features and samples, the algoritm shown in Algorithm 3 is applied to estimate the hand pose given a new (set of multiple view) image(s). Note that, although the initial steps are not affected by feature selection and sample selection, these have a large impact on steps 6 to 13 (specially 11). Thus a trade-off between speed and robustness can be achieved.

7.5 Experiments and Results

This section presents experiments on applying regression for feature selection, samples selection and both combined. Aiming to perform a fair comparison, for both single-view and three-views data, 90-dimensional $\mathbf{x}$ descriptors were used, with the difference that, for the former, all the elements of $\mathbf{x}$ were obtained from the same view, and, for three-views, each view was described by a 30-dimensional vector,

Algorithm 3 Pose Estimation with Selected Features and Samples (application phase)

```
Require: lists of selected features, selected samples $\mathbf{x}_i$, shape contexts centroids, means $\overline{\mathbf{x}}$ and $\overline{\mathbf{y}}$,
    $std(\mathbf{x})$, $std(\mathbf{y})$, and matrix $\mathtt{A}$
 1: for each camera $c$ do
 2:   if there are selected centroids from this view then
 3:     Extract the hand silhouette using skin detection and edge detection
 4:     Compute all the shape context vectors from all the silhouette contour points
 5:     Calculate their distances to all the centroids of this view
 6:     Soft histogramming: create the image descriptor $\mathbf{x}^{c\prime}$ taking into account only bins related to the
        selected centroids
 7:   end if
 8: end for
 9: Concatenate the vectors $\mathbf{x}^{c\prime}$ into a single vector $\mathbf{x}'$
10: Standardise it using the mean $\overline{\mathbf{x}}$ and $std(\mathbf{x})$ from the training set
11: Evaluate functions $\phi_i(\mathbf{x}')$ to build $\mathbf{f}(\mathbf{x}')$, where $i$ is the index of kernel functions related only to the
    selected training samples
12: Apply $\mathbf{y} \leftarrow \mathtt{A}\mathbf{f}(\mathbf{x}')$ with the selected columns of $\mathtt{A}$ only
13: ‘De-standardise’ $\mathbf{y}$, using the mean $\overline{\mathbf{y}}$ and $std(\mathbf{y})$ from the training set
```

and they were concatenated to build $\mathbf{x}$. The parameters λ and ν were both set to 0.3, which, in most of the experiments in this chapter, lead to convergence after three iterations for linear kernel functions, and after five iterations for Gaussian kernel functions.

7.5.1 Number of Relevance Vectors

The graphs of Figure 7.19 show the number of selected relevance vectors as a function of the threshold $\mathcal{T}_{\mathbf{a}}$. Note that the same threshold leads to the selection of more relevance vectors for a single view. This hints that even though the same number of training samples (and of the same dimensionality) is used in both cases, fewer relevance vectors are selected for multiple views, indicating that their measurements are more discriminative. Fewers samples and fewer features are needed to achieve the same relevance for multiple views. A similar behaviour has been observed for the *complex* training set.

7.5.2 Synthetic Images

For the experiments with synthetic images, ground truth is available. The data set was evenly split in a training set and a testing set (with no intersection) from the same sequence of movements. Although this practice makes training and testing data very similar, it is enough to distinguish the performance between single and multiple view methods.

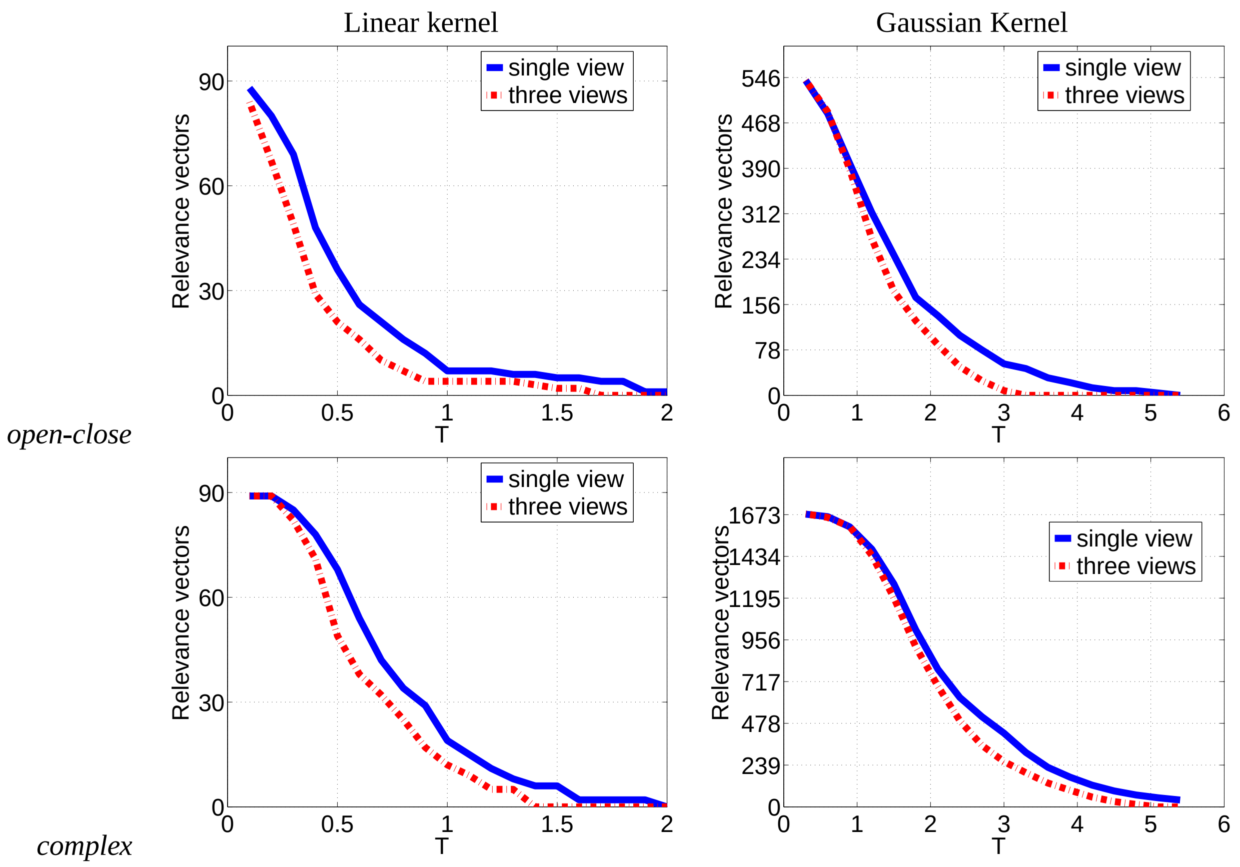


Figure 7.19: Number of selected relevance vectors for linear and Gaussian kernels for single and multiple views as a function of the threshold $\mathcal{T}_\mathbf{a}$ evaluated for both the *open-close* and *complex* training sets.

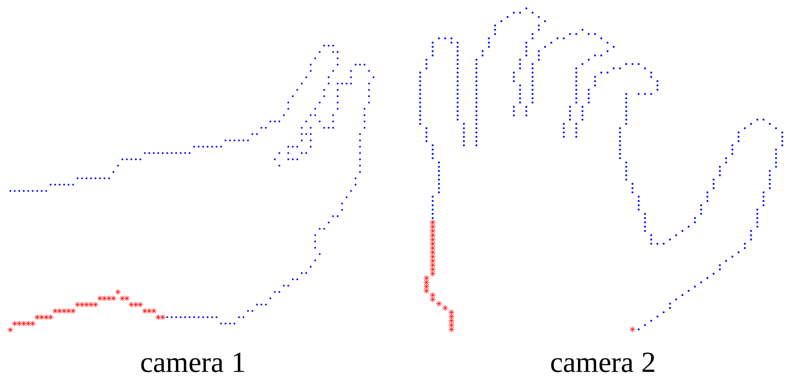


Figure 7.20: Silhouettes obtained from a sample pose in the training set from camera 1 and 2, highlighting (with red '*') the points whose shape context is taken into account after the selection of two relevant features.

Open-Close **Data Set**

The sequence of movements in the *open-close* dataset can roughly be described by two degrees of freedom: flexion of the all joints and twisting movement of the hand about the forearm axis (θ_Z). In order to verify the ability of the regressor to identify this, a feature selection experiment was performed, i.e., a regressor with linear kernel functions was trained with the threshold $\mathcal{T}_{\mathbf{a}}$ on $||\mathbf{a}||$ tuned to select only two relevance vectors. But for a single-view, three features were selected, because any greater threshold resulted on only one feature. For three-views, one vector from the top view and another vector from one of the side views (camera 1) were selected, as shown in Figure 7.20 and 7.21.

The points of the silhouette shown by red '*' in Figure 7.20 are those whose shape contexts (SC) have the selected centroids among their four nearest centroids in SC space (the soft histograming implemeded considers a neighbourhood of four centroids). Note that, for both views the centroids selected are close to the wrist rather than the finger tips. A possible reason for that is that features closer to the finger tips present too much variation between samples and they are not present for some of the samples, like those with the hand in fist pose. This was also observed for single view.

Figure 7.22 illustrates the result by showing the estimated angle of the interphalangeal joint of the index finger and the results for θ_Z. Recall that at each 78 frames the images were generated for a different value of θ_Z (global hand orientation). Note that the pose of the hand was estimated individually for each frame, which explains the jittering motion.

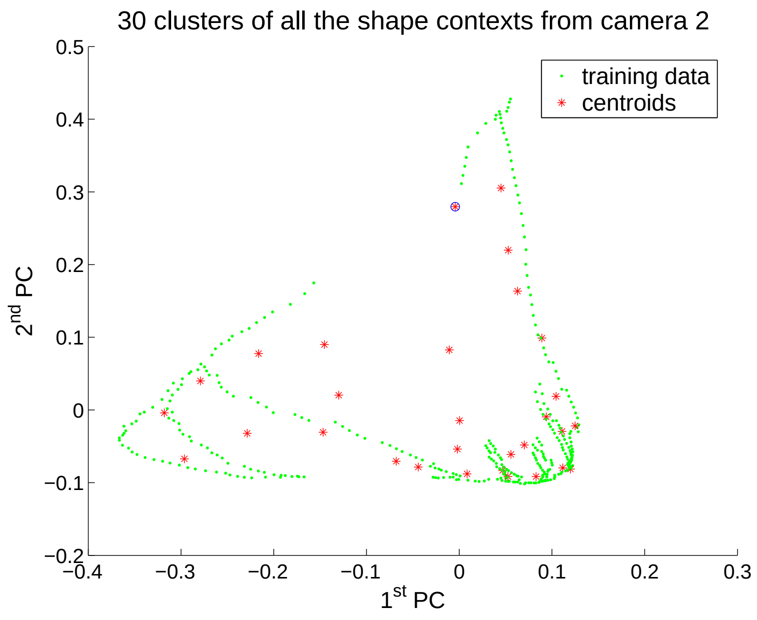


Figure 7.21: Shape contexts manifold with the centroid of the selected cluster from camera 2 indicated by a blue circle.

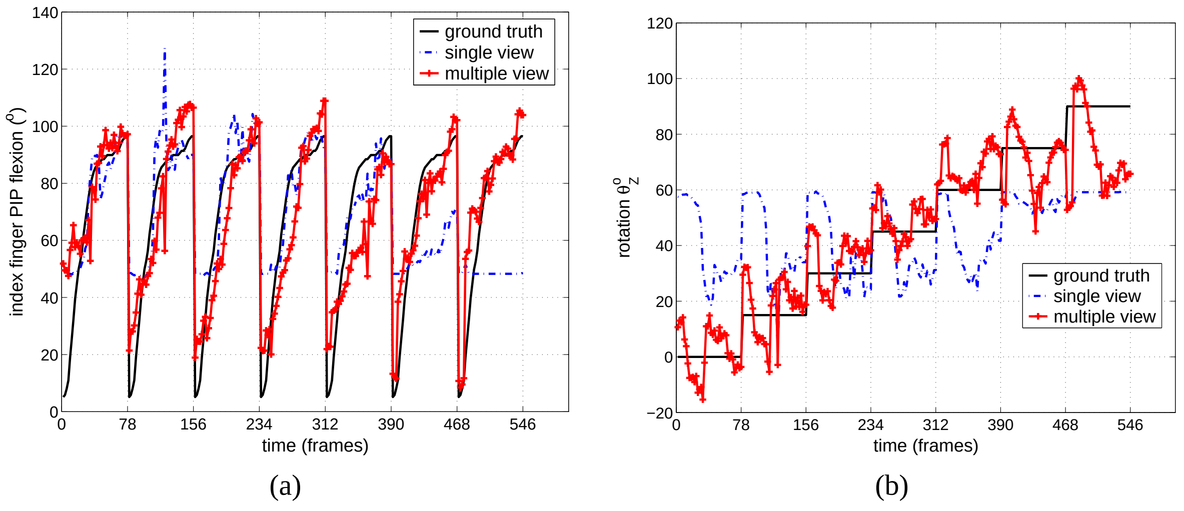
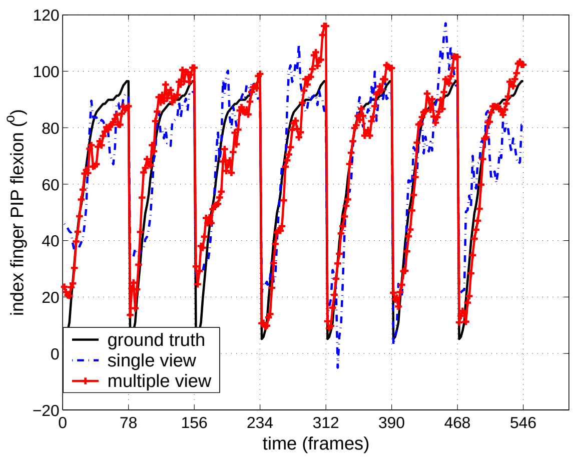
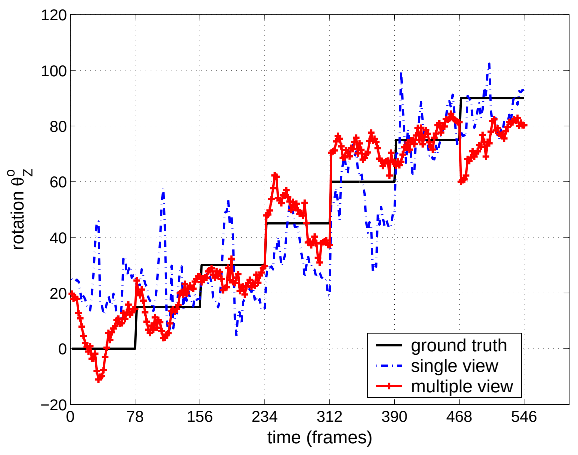


Figure 7.22: Regression results using only three (for single view) and two (for multiple views) relevance vectors (out of 90), with a linear kernel and synthetic images: (a) estimated angle of the interphalangeal joint of the index finger; (b) estimated angle θ_Z of global rotation about the forearm axis.



(a)



(b)

Figure 7.23: Regression results using only 10 samples (out of 273), with Gaussian kernel functions and synthetic images: (a) estimated angle of the interphalangeal joint of the index finger; (b) estimated angle θ_Z of global rotation about the forearm axis.

These results show that the regressor is able to give a rough approximation of the pose using a minimal set of selected vectors (in this case, image features). Even using fewer features for multiple views it is possible to achieve higher accuracy than with a single view. Also, the results for θ_Z with a single view seem to have no correlation with the ground truth. Furthermore, for single view, as θ_Z grows, the estimate for other angles gets poorer because the top view does not offer enough distinct features on its own when the fingers get nearly aligned to the camera axis.

When using Gaussian kernels, it is harder to intuit the minimal set of samples needed to estimate the pose. $\mathcal{T}_{\mathbf{a}}$ was chosen so that 10 relevant samples were selected from the training set. The trajectories obtained from single and multiple views are shown in Figure 7.23 in comparison with the ground truth.

Both for single and multiple views, the selected samples are mostly from “near-fist” hand poses. This may seem odd, but it is not usual in RVM for the the most relevant samples to be distant from the obtained pose estimates, and for them not to be the most comprehensive samples in terms of the variability of state (poses) [Tip01].

Figure 7.24 reports the application of feature selection followed by samples selection to combine speed and performance. Note that the superiority obtained for multiple views is more evident for θ_Z.

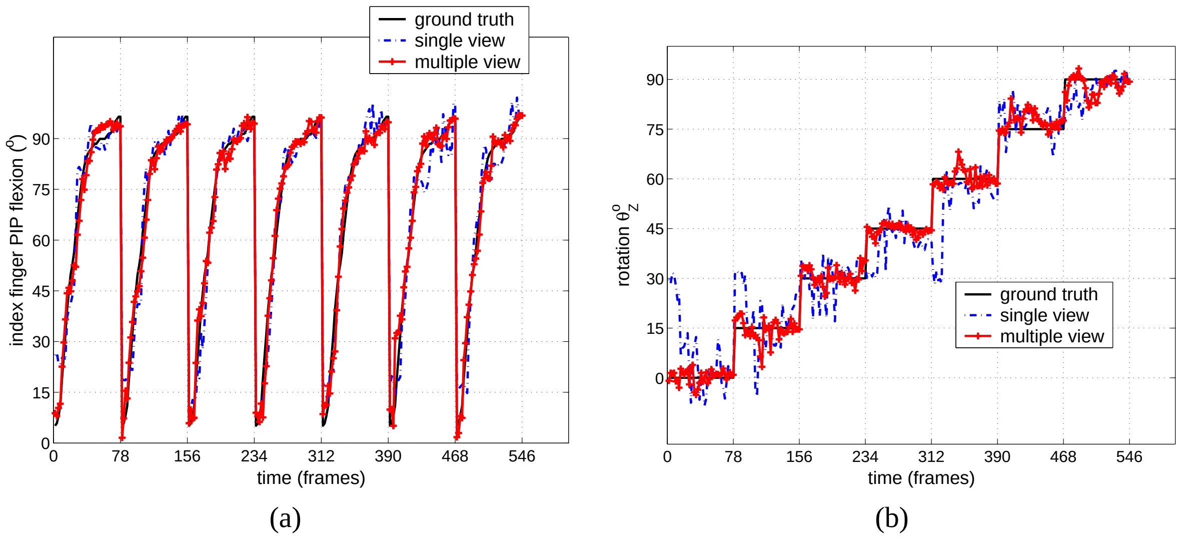


Figure 7.24: Regression results combining both feature selection and samples selection for the interphalangeal joint of the index finger (a) and the angle θ_Z (b). The parameters were tuned to select 13 or 12 features for single and multiple view, respectively; and 29 samples.

Complex Data Set and Quantitative Results

The *complex* data set incorporates a large range of hand poses, so it is more difficult to illustrate the results with graphs as shown for the *open-close* data set. Figure 7.25 shows the mean and standard deviation of the error for each parameter (DOF) of the hand for the *complex* data set, using half the samples for training and half for testing with both feature selection and samples selection. In this case, 36 relevance vectors with 35 dimensions were selected for both single and multiple view. For a comparison, the STD of the training trajectory is also shown. Note that both the greatest variation in angle and the greatest average error occur in the proximal interphalangeal joint of the fingers. In this database, the use of multiple views reduces the error in a roughly uniform manner along the pose parameters.

Table 7.1 shows a quantitative evaluation of the results for both data sets using synthetic images. The columns ‘ftrs.’ and ‘spls.’ indicate how many relevance vectors were selected with linear and Gaussian kernel, respectively. The column ‘worst result’ shows the average error for the parameter (DOF) whose estimate was the worst, indicated in the column ‘which DOF’. The abbreviation T IP refers to thumb’s inter-phalangeal joint, and M DIP to the distal inter-phalangeal joint of the middle finger.

As expected, the worst estimates occurred in two cases: (i) for DOFs related to parts of the hand whose contour was occluded in many of the images, and (ii) for the rotation θ_Z when a single view is

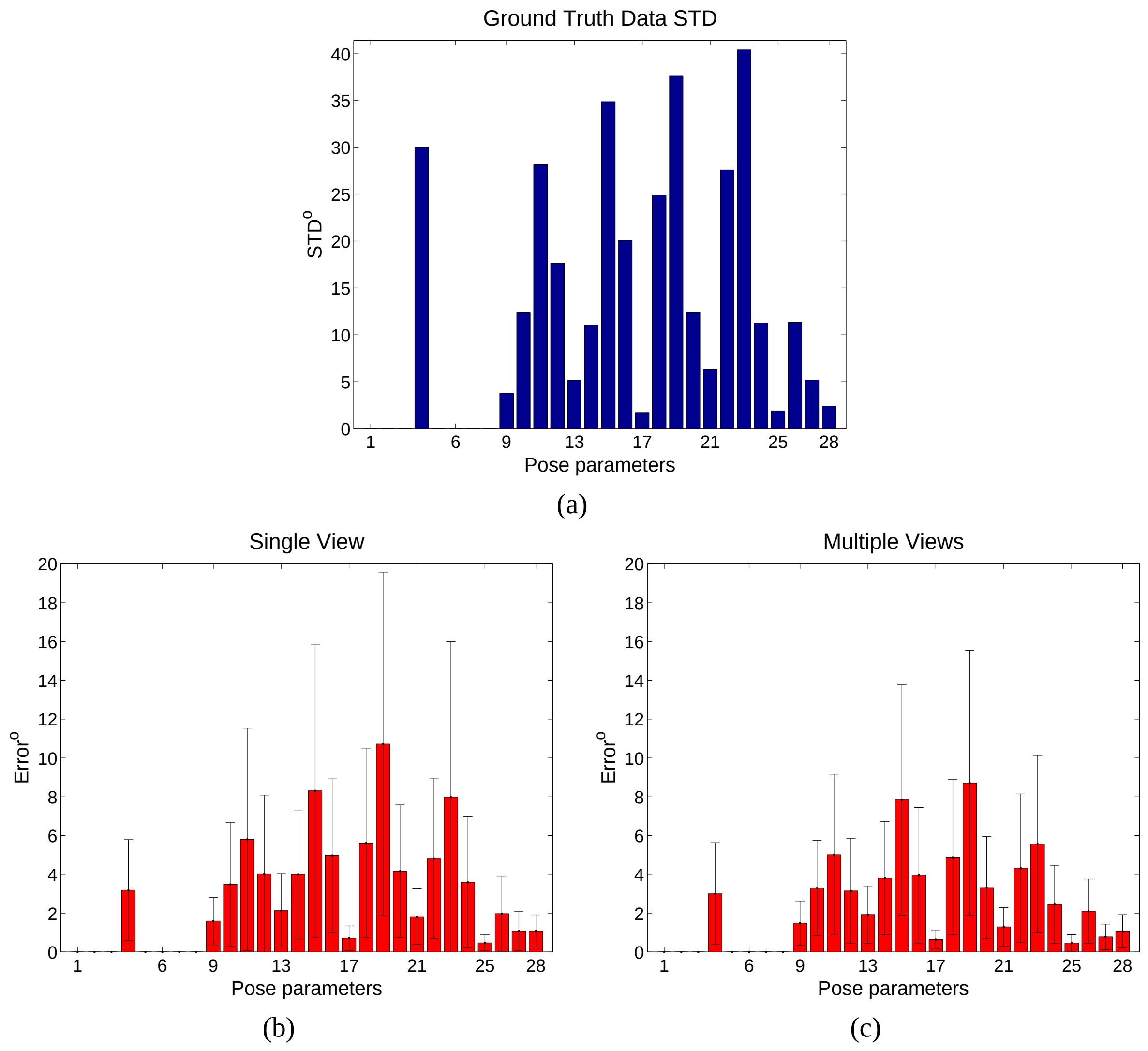


Figure 7.25: Panel (a) shows the standard deviation (STD) of the value of each hand parameter along the trajectory of the *complex* data set. Parameter 1-6 are absolute pose, 7 and 8 are wrist angles, 9 is abduction of the little finger and 10-12 are flexion angles. The same pattern repeats for each of the other fingers and thumb. Panel (b) shows the mean error and STD for each parameter using a single top view. Panel (c) shows the same for multiple views.

# Views	Data Set	Kernel	Ftrs.	Spls.	Avg. Error	STD	Worst Result	Which DOF
1	*Open-Close*	Linear	3	273	8.6°	6.9°	23.2°	θ_Z
		Gaussian	90	10	5.6°	4.3°	14.5°	T IP
		Both	13	29	2.3°	2.0°	6.0°	θ_Z
	Complex	Linear	31	839	3.0°	2.7°	11.4°	M DIP
		Gaussian	90	42	2.9°	2.5°	9.9°	M DIP
		Both	35	36	2.9°	2.6°	10.7°	M DIP
3	*Open-Close*	Linear	2	273	5.4°	4.4°	17.0°	T IP
		Gaussian	90	10	3.6°	2.7°	14.9°	T IP
		both	12	29	1.6°	1.2°	7.0°	T IP
	Complex	Linear	31	839	2.5°	2.1°	8.9°	M DIP
		Gaussian	90	41	2.4°	2.0°	8.3°	M DIP
		Both	34	36	2.4°	2.0°	9.0°	M DIP

Table 7.1: Results with synthetic data obtained using 273 and 839 training samples for *open-close* and *complex* data sets, respectively. The same amount of samples was used for testing, though there is no intersection between the sets. For both data sets, the total number of features used is 90.

used, as this is not a rotation parallel to the top view image plane.

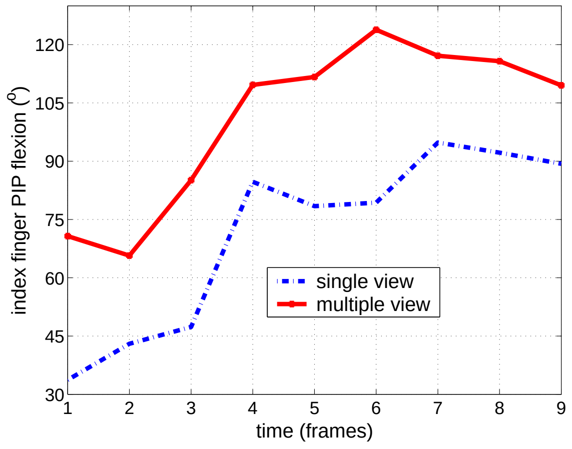
In general, the improvement obtained by using multiple views is evident, particularly when the number of features used is small. However the improvement is view-dependent, and if a single view captures the most meaningful silhouette the improvement is diminished. A further reduction in improvement arises because the synthetic images used so far are noise free. As shown in next section, improvement is restored when using real images.

7.5.3 Real Images

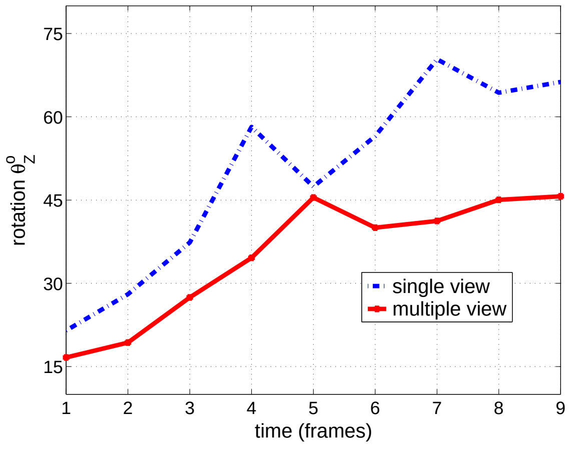
For real images, whole training sets were used, giving 546 training pairs for the *open-close* data set and 1679 for the *complex* data set. For testing, images of the right hand of a single subject were used. Since there is no ground truth available for the real images, only qualitative results are shown.

Figure 7.26, shows the estimated index PIP joint and θ_Z angles obtained by training the regressor with the *open-close* data set and applying it to the nine images shown in Figure 7.15. It is intuitive to visualise the correctness of these results, as both the estimated index PIP joint and θ_Z angles are expected to grow with time. In this case, the use of multiple views does not seem to show an improvement in relation to single view.

However, Figure 7.27 shows that multiple views provide a significant improvement for images with



(a)



(b)

Figure 7.26: Regression results obtained using the *open-close* data set, combining linear kernel functions to select 30 features (out of 90) and Gaussian kernel functions to select 46 and 47 samples (out of 546) for single and multiple view data, respectively. (a) shows results for the interphalangeal joint of the index finger, and (b) for θ_Z.

more complex movements, using the regressor trained with the *complex* data set. This improvement becomes more evident when a small selection of features and samples is used, as shown in Figure 7.28. Note that, for a single camera, the regressor seems to be unable to recover some of the poses, probably because the measurements generate poses that extrapolate the space of trained poses.

7.5.4 Computational Cost

As the aim is to use this system for (re-)initialisation of a video-rate hand tracker, the computational cost is evaluated in this section. Although most components of this system were implemented using an interpreted language (MatLab 6.5, except where indicated), the time measurements presented here give a good clue of the computational complexity of each part of the algorithm. These experiments were performed using a computer with two 2.4GHz Pentium 4 CPUs and 750MB of RAM running Red Hat 9 Linux (though the algorithm was not parallelised).

Feature Extraction

This is the first step to obtain image descriptors, both for training and testing samples. For 5037 images of 120×90 pixels, the average time for skin colour detection was 2.8ms using a compiled C++ implementation. To extract subsampled silhouette contour points and calculate their shape contexts it takes

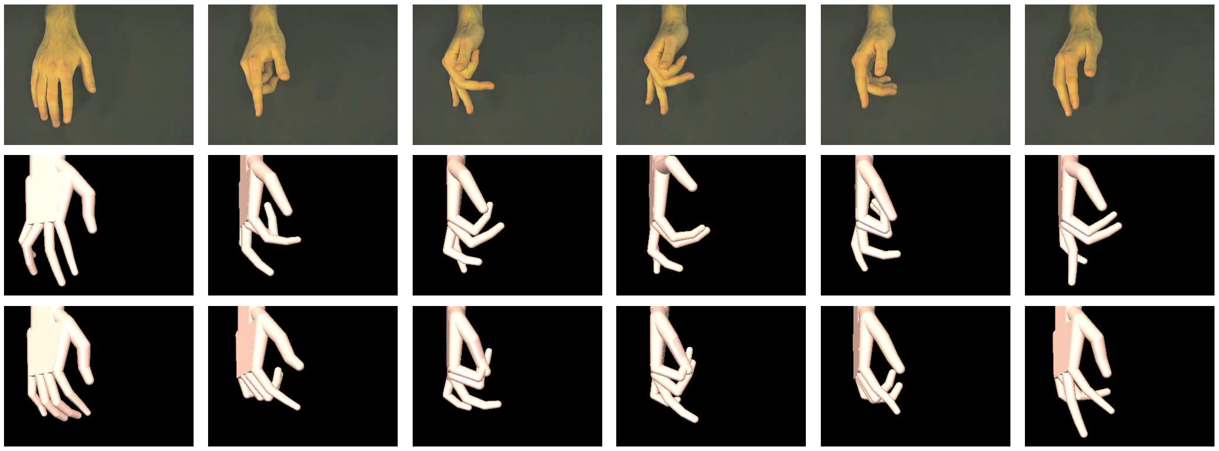


Figure 7.27: Results obtained from real images (top row) for single view (middle row) and multiple views (bottom row), using Gaussian kernel with all the samples and all the features.

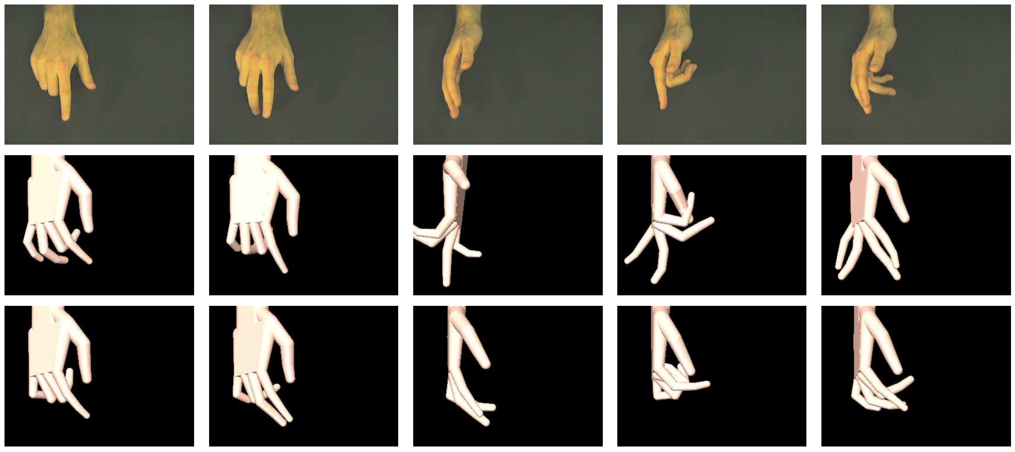


Figure 7.28: Results obtained from real images (top row) for single view (middle row) and multiple views (bottom row), using combined linear kernel to select 32 features (out of 90) and Gaussian kernel to select 38 samples (out of 1679)).

Kernel	Input Dim.	Selected RVs	Iters.	Total Time (s)
Linear	90	32	4	6.7
Gaussian	90	38	4	327.7
Both	32	38	5	305.4

Table 7.2: Training time for the *complex* data set (1679 samples) for linear, Gaussian and the combined kernels to select 32 features and 38 samples.

further 141.7ms per image in MatLab. Next, to calculate the quantised vectors $\mathbf{h}_c$ it takes 26.4ms per image. Therefore, the average time for this pre-processing stage is 170.9ms and this is the only stage where the use of multiple view can represent a linear ($\mathcal{O}(C)$, where C is the number of cameras) increase in the amount of time required by the algorithm.

Training Phase

The most demanding step of the training phase is the clustering to obtain the centres for vector quantisation. With the *complex* data set the algorithm did not converge until the maximum number of iterations (30) was reached. For this, K-means can take between one and ten days, depending on how many iterations of the second phase were performed, and this is data dependant [Seb84]. For the *open-close* data set, a result was obtained between 30 mins and 2 days, again, depending on the view and the number of second phase iterations performed. No convergence was reached within 30 iterations, but experiments have shown that the quality of the centroids for vector quantisation does not affect the discrimination power of the obtained vectors [JT05].

Given the training samples represented by the (concatenated) quantised vectors $\mathbf{x}$, to train Agarwal and Trigg's regressor is a much faster process, as shown in table 7.2. Note an improvement in speed using the linear kernel functions followed by Gaussian kernel functions (*both*) in relation to using Gaussian kernel functions only.

Application

As shown in table 7.3, the actual pose estimation process is extremely fast (note that the scalars are in microseconds). For both for training and application, the difference between using both kernels and Gaussian kernel functions only is not very significative in comparison with the difference between these and the use of linear kernel functions only. However, combining both methods give the robustness of

Kernel	Features	Training Samples	Time (μs)
Linear	32	1679	7.2
Gaussian	90	38	35.7
Both	32	38	25.4

Table 7.3: Average time over 1679 trials for the application of the regressor to the *complex* data set using linear, Gaussian and the combined kernels. Note that the time scale is in microseconds (μs).

Gaussian kernel functions and the additional reduction on the risk of overfitting if all the features are used.

Summing up all the steps give, in the worst case (three cameras and Gaussian kernel functions) 651.69ms per frame, which is a very good result for a global detector with no prior information of the hand pose, implemented mostly in an interpreted language. Better results are expected in a compiled version, but the use of this method for detection only to (re-)initialise a generative tracker is still the best option for better results at lower computational cost.

Memory Usage

For application, this algorithm is not very memory demanding. Below is the list of required data:

- list of selected features: $\mathcal{O}(K)$, K is the dimension of $\mathbf{x}$;
- selected samples $\mathbf{x}_i$: $\mathcal{O}(K \times I)$, I is the number of training samples;
- shape context centroids: $\mathcal{O}(K)$;
- mean and standard deviation for vectors $\mathbf{x}$ and $\mathbf{y}$: $\mathcal{O}(K)$ and $\mathcal{O}(m)$, where m is the dimension of the state vectors $\mathbf{y}$;
- matrix $\mathtt{A}$: $\mathcal{O}(m \times K)$ for linear kernel functions or $\mathcal{O}(m \times I)$ for Gaussian kernel functions.

For a new image, in order to get $\mathbf{x}'$ (steps 4 to 6 of Algorithm 3), first it is necessary to obtain all the shape contexts from the silhouette contour. These are $\mathcal{O}(r \times a \times n)$, where r and a is the number of radial and angular bins of the shape contexts. In this chapter, $r = 5$ and $a = 12$; n is the number of points in the silhouette contour. For the *complex* data set the average number of points per image is 2549.1 per image in the experiments of this chapter, which is not a high value for today's computers.

However, the training phase can be particularly demanding, specially in the clustering step, where it is convenient to keep all the shape context of all the training images in the memory: $\mathcal{O}(r \times a \times n \times I)$. With the *complex* data set, 750MB of memory was enough for all the experiments (including training and clustering), in a MatLab implementation. But, for training, it was necessary process views individually and store data from the other views in the HD. Obviously, a significant amount of memory could be saved in a C or C++ implementation, in which numbers would not need to be represented with double precision for the calculations.

7.6 Conclusions

This chapter presented a regression-based method for estimation of hand pose in 3D from global image descriptors, advancing the single-view method of Agarwal and Triggs [AT04a] proposed for human pose estimation.

Skin silhouettes were extracted from colour imagery, and their contour points described using the shape contexts of Belongie *et al*. [BMP02]. The considerable variation in hand pose typically observed in imagery requires care to be taken to ensure scale and rotational invariance in the contexts. The use of contexts aligned with the axis of the forearm was found to be the best. By ensuring rotational and scale invariance, the number of training samples needed was reduced, provided triangulation was first used to recover the global pose parameters.

A global image descriptor for each view was obtained by coding the manifold of shape contexts using vector quantisation, and the descriptors combined at an intermetiate level into multiview descriptors by concatenation. The mapping between multiview descriptors and 3D poses was learned using Agarwal and Triggs' [AT04a] extension of Tipping's Relevance Vector Machine [Tip01].

Our experiments have, inter alia, examined the effects of feature selection (linear kernel functions) and sample selection (Gaussian kernel functions) both on the quality of pose determination and on the computational time, using both synthetic and real imagery. We have found that linear kernel functions have the advantage of computational cost independent on the amount of training data used. However, we have found Gaussian kernel functions to be more robust, so we have performed experiments combining both linear and Gaussian kernels for speed and robustness. Our experiments have also shown that, for general views, fewer relevance vectors are needed in the multiple view case. Their measurements are more discriminative, allowing correct pose estimates to be recovered in cases where a single view all but fails.

An obvious modification to the current image descriptor would be the use of a more sophisticated coding method, like Gaussian mixtures or Jurie and Triggs's method [JT05]. Another possibility is to explore the extension of RVM for multidimensional target spaces of Thayananthan *et al*. [TNS+06] which, like the original RVM, optimises the hyperparameters. But the main thrust of future work should be to evaluate how relevant is the use of multiple hypotheses if multiple views are employed. A more application-oriented direction of this work is the integration with a generative tracker for real-time results, as done in [AT05], [EZ05], and [RS06].

8

Conclusions

This thesis concludes with a brief overview of the topics that have been discussed. The main contributions are listed and some suggestions are given for interesting areas of future research.

8.1 Summary of this thesis and conclusions

After an introduction to the motivation for the work in the area of visual human-computer interaction, this thesis presented a review of the literature focused on hand tracking and human motion capture. Two main approaches for 3D articulated object pose estimation have been identified: generative and discriminative. Generative methods are the more traditional approach to tracking in which a model of the object is rendered at a predicted pose and image measurements evaluate the agreement between this model and the observed image. These are then used to compute a new prediction of the pose and the tracking cycle repeats.

The discriminative approach is a natural way of estimating pose from a single image, but it has also been extended to "tracking-by-detection" frameworks. This method relies on extensive prior knowledge of appearances and poses of the object in order to build a direct map between image observation and 3D pose.

This thesis has developed methods in both categories. It started by revising a generative method first proposed in the early 1990's by Harris [Har92a] for tracking rigid objects in real-time. RAPiD tracker uses sparse edge measurements to compute the pose update by solving a linear system. As a test of the combined use of calibration, hand detection using colour, and tracking at video-rate using edge following, an application was developed in which a pointing hand controlled the direction of gaze of a wearable active camera.

By analysing the motion of kinematic chains, it was shown that the formalism developed in RAPiD could be transferred to articulated objects. The articulated RAPiD Tracker (ART) was tested with synthetic and real images, including a video sequence of the hand grasping a box, where both hand and box are tracked.

Acquiring information about the interaction between objects was one of the original motivations of this thesis. For this reason, an alternative method to represent articulated objects was studied. This method, proposed by Drummond and Cipolla [DC02], is based on estimating an initial motion screw for each object part individually and then imposing constraints afterwards. Although this is clearly different from ART's approach, which encode all the constraints in a single system, it was shown that these methods provide equivalent results. However, they differ in terms of simplicity, robustness and asymptotic speed in relation to the number of object parts. It was shown that ART is more suited to articulated objects with highly constrained parts (e.g. hands), and Drummond and Cipolla's tracker (DCT) is more appropriate for problems in which the constraints between parts are low and where the ability to switch on and off the constraints is desired.

Although the bone linkage of hands can clearly be modelled as a kinematic tree, skin tissue challenges generative methods if its complex dynamics is not modelled, but modelling such tissues can be computationally expensive. Furthermore, the reported success of generative methods for hand tracking is usually restricted to predictable movements, but sudden motions are very usual for hands. For these reasons, a discriminative method was implemented.

This method is based on building a map between a global shape descriptor and 3D poses. Hand images are described using shape contexts measured at the contour of the silhouette of skin coloured blobs. This description is encoded using a quantisation of the shape context space which gives a high dimensional vector for each image. A multiple view descriptor is obtained by concatenation of these measurements. To create an efficient map between image measurements and 3D poses, a multivariate regression method based on Relevance Vector Machines was implemented. Experiments have shown accurate pose estimation results on synthetic data and satisfactory results on real images. Comparisons between single and multiple view versions of this method showed that the extension to multiple views proposed in this thesis improves the results and reduces the number of relevance vectors required.

8.2 Contributions

Although a vast literature was found in this field, the problem of 3D hand locating and tracking in real-time remains very challenging and it is still open for contributions. This thesis has achieved significant contributions in the following areas:

- **Tracking articulated objects**

  A novel articulated object tracker described with the same formalism as RAPiD was proposed in Chapter 5. This method can track articulated objects with any type of joints and topology. A qualitative evaluation of this method was presented for a range of objects, including a hand interacting with a box.

  An alternative method (DCT) was investigated in Chapter 6. In this method, constraints are post-imposed after an initial estimate of the motion of each object part is computed. A novel comparative study between this approach and the pre-imposed constraints approach of ART was presented. This study took into account equivalences in their formulation, tracking results, simplicity, robustness and speed of the methods varying a number of parameters. This comparison was published in [dTM06] and [dTM05].

- **Estimating 3D hand pose from multiple view**

  A new regression-based method for 3D hand pose estimate from multiple view images was proposed in Chapter 7. The experiments showed that this method achieves better performance in comparison to using a single privileged view. Not only accuracy is improved, but complexity can be reduced. Results of this work have been published in [dM06].

8.2.1 Secondary contributions

- **Literature survey**

Despite not presenting new technical advances in the field, Chapter 2 tries to attend to the current demand for a survey of the literature. A lack of comprehensive surveys focused on hand tracking has been identified, since the last comprehensive review in this field was published ten years ago [PSH97].

- **Skin detection**

  One of the apparatus building methods presented in Chapter 3 is a skin colour detector that is based on a histogram-based classifier applied to the YCbCr colour space. Its novelty lies on the particular combination of colour space and classification method. This method is an important component of the systems published in [TMdM02b], [dMM06], [MTdC$^+$03] and [dM06].

- **Guiding a wearable active camera using pointing gestures**

  Chapter 4 presented a new system that combines a fast shape detector with the RAPiD tracker in order to detect and track pointing gestures. A cost function diagnoses the result to evaluate if the detector should be fired to re-start tracking. This system was applied to command the gaze direction of a wearable active camera mounted on the user's shoulder. Parts of this work have been published in [dMM06] and [MTdC$^+$03].

8.3 Future directions

Some of the work developed contemporaneously to this thesis give clues of future directions. Below, key points are listed.

In the field of image features for 3D tracking, results can be enhanced if other image features are combined with the edge features used in this thesis. For instance edges can be combined with optical flow and shading information in [LMSO03]. A combination framework as that of Tordoff *et al.* [TMdM02b] can also be evaluated for this application.

To reduce the complexity of ART and improve robustness, learnt hand motion data can be used. A method such as Lawrence's SGPLM [Law04] can provide an efficient data-driven dimensionality reduction of the state space incorporating learnt constraints.

This thesis concentrated on pose update assuming that the frame-rate is high enough. To improve the results for fast (but smooth) motions, a motion filter can be used. Alternatively, a particle filter-based method such as that of [BKMV04], can be applied to improve robustness and avoid local minima, but this may increase the computational cost.

Although shape contexts are strong descriptors, they rely on good silhouette or edge segmentation results, which is not always possible for cluttered backgrounds. An alternative is the use of SIFT features, as done in [AT06a]. In that work, the appearance of human-like parts is learnt in order to suppress background features, so no segmentation is required. This method is likely to be very successful for hands, because bare hands texture is much simpler than the texture of people wearing different clothes.

For the global image descriptor, it can be relevant to evaluate coding methods that are better than the simple vector quantisation employed in Chapter 7. An example is the method of Jurie and Triggs [JT05].

Even though the use of multiple views virtually eliminates ambiguities in pose estimation, the amount of self-occlusion in hands still make some different hand poses have similar appearances (e.g. a fist). A study of the importance of multiple regressors (e.g. [TNS$^+$06]) for pose estimation should be considered as future work.

This thesis advocates that hands should be located and tracked using the combination of a discriminative and generative methods. This has been done in Chapter 4, limited to pointing hand detection and tracking. The combination of the approaches of Chapters 7 and 5 will extend the range of possible poses, but this has not been attempted in this thesis.

The methods developed in this thesis are yet to be integrated with other systems that watch humans to perform action and intention recognition. Methods for combination of temporal information shall be applied, and the research on behaviour understanding shall lead to the ultimate goal of developing smart human–machine interfaces. Current methods demand training and some effort that, despite being subtle, in the long term can lead to chronic damages of RSI. In the ideal interface, the training and effort should be transferred to machines: they should observe humans and be able to adapt themselves for the optimal communication — but the poor researchers that develop them will all suffer from RSI before such interfaces become available.

A

Colour spaces

This appendix describes the colour spaces that are most commonly used for brightness normalisation in order to detect skin colour. A large set of skin and background samples is used to show their spread in the different colour spaces.

A.1 The RGB colour space

Extensive experiments in the human visual system have showed that the cones — sensors in the eye responsible for colour vision — can be divided into three principal sensing categories, corresponding roughly to red (R), green (G) and blue (B) [WS00]. Therefore, colours are seen as combinations of these so-called primary colours [GW00]. For this reason, most of the cameras and emissive colour displays represent pixels as a triple of intensities of the primary colours in the RGB colour space: $[R, G, B] \in \mathbb{R}^3$. This is also the reason why the RGB space is very commonly used by the computer graphics and image processing community.

A disadvantage of the RGB representation is that the channels are very correlated, as all of them include a representation of brightness. This is illustrated in Figure A.1 and A.2, in which the brightness information can be recognised from R, G and B channels shown separately.

True colour 24 bits RGB images have the triple $[R, G, B]$ represented by 256 discrete values (ranging from 0 to 255) [Jac01], thus the range of RGB colour values forms the cube of $(2^8)^3$ possible values as shown in Figure A.3. The high correlation between lightness and RGB channels can be noted by the line of the grey values, where $R = G = B$. In fact, if the corresponding elements in two points, $[R_1, G_1, B_1]$

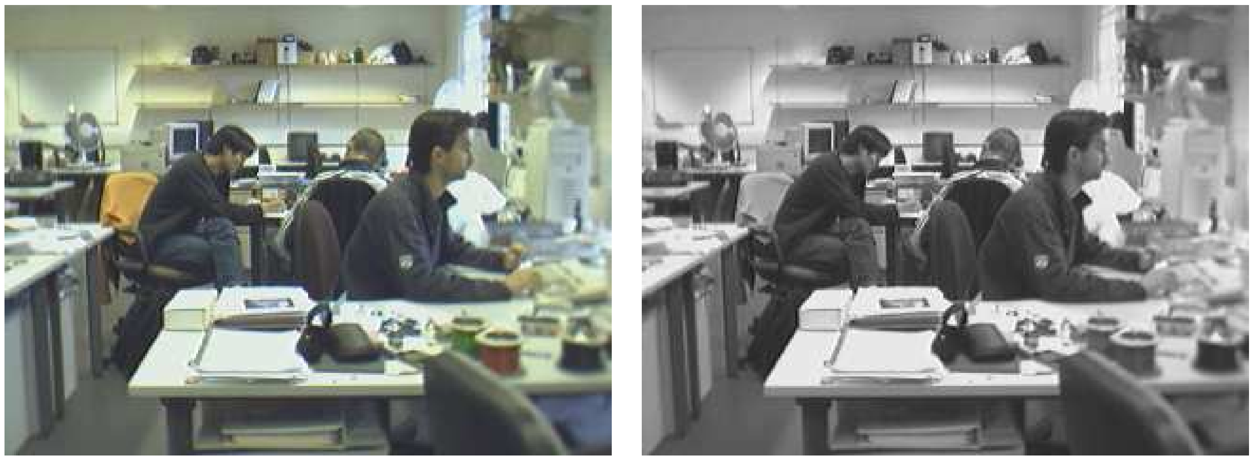

(a) (b)

Figure A.1: Sample colour image (a); and its grey level version (b).

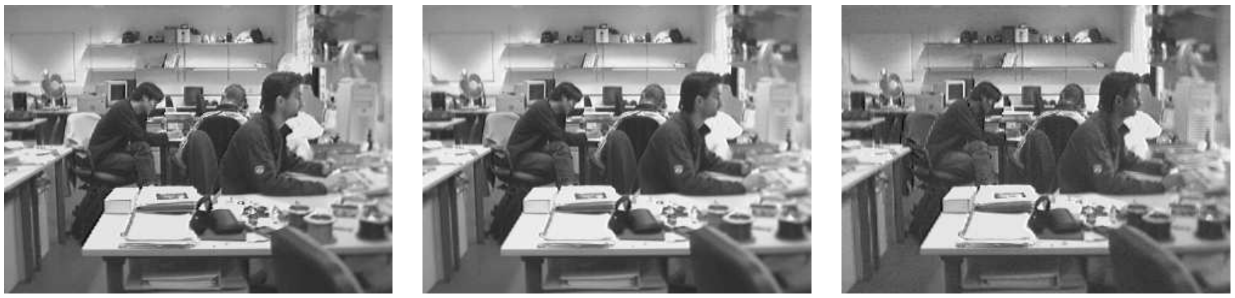

R G B

Figure A.2: RGB channels of image in Figure A.1(a) shown separately.

and $[R_2, G_2, B_2]$, are proportional, i.e.,

$$\frac{R_1}{R_2} = \frac{G_1}{G_2} = \frac{B_1}{B_2}, \tag{A.1}$$

they have the same colour, but different brightness [YW96]. Differences in brightness are often disregarded by humans, as our visual system is capable of adapting to different brightness and various illumination sources such that the perception of a colour constancy is maintained within a range of environmental lighting conditions [WS00].

A.2 The CIE chromatic space

The CIE chromatic space is a standard proposed in 1931 by the Commission Internationale de l'Eclairage – the International Commission on Illumination. Some modifications have been proposed later, but this

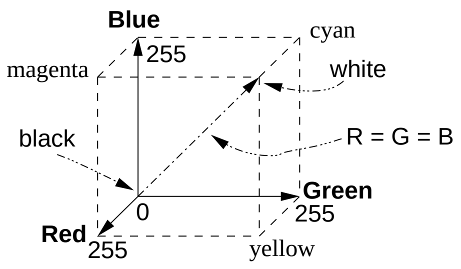


Figure A.3: The RGB colour cube.

section is restricted to the 1931 standard. It has been used in several colour processing tasks [GW00] and it is used to define the colour gamut, i.e., the range of possible colour values that a device can represent.

This two dimensional space has the x and y axes respectively defined by the pure chromatic colours red and green (r, g), defined by this normalisation process:

$$\begin{aligned} r &= \frac{R}{R+G+B} \\ g &= \frac{G}{R+G+B} \end{aligned} \tag{A.2}$$

which is, in fact, a $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ map. Pure blue (b) is redundant after the normalisation because $r+g+b = 1$ [WS00].

The use of this colour space for skin detection has became popular specially after the work on face tracking developed at SCS, Carnegie Mellon University [YW96, YLW98b].

A.3 The perceptual colour space

The perceptual colour spaces were designed by Smith in [Smi78] in order to provide a more "intuitive" way of describing colours and lightness. Three quantities are used to define them: hue, saturation and brightness. Brightness embodies the achromatic notion of intensity. Hue is an attribute associated with the dominant wavelength in a mixture of light waves. It represents colour as perceived by an observer. Thus, when we call an object blue, yellow or red, we are specifying its hue. Saturation refers to the relative purity or the amount of white light (or grey of equal intensity) mixed with a hue. Primary colours (pure red, green and blue) are fully saturated, whereas colours such as pink (red and white) and lavender (violet and white) are less saturated. The degree of saturation is inversely proportional to the amount of white light added [GW00].

Basically, there are two distinct perceptual colour spaces: HSL (hue, saturation, lightness); and

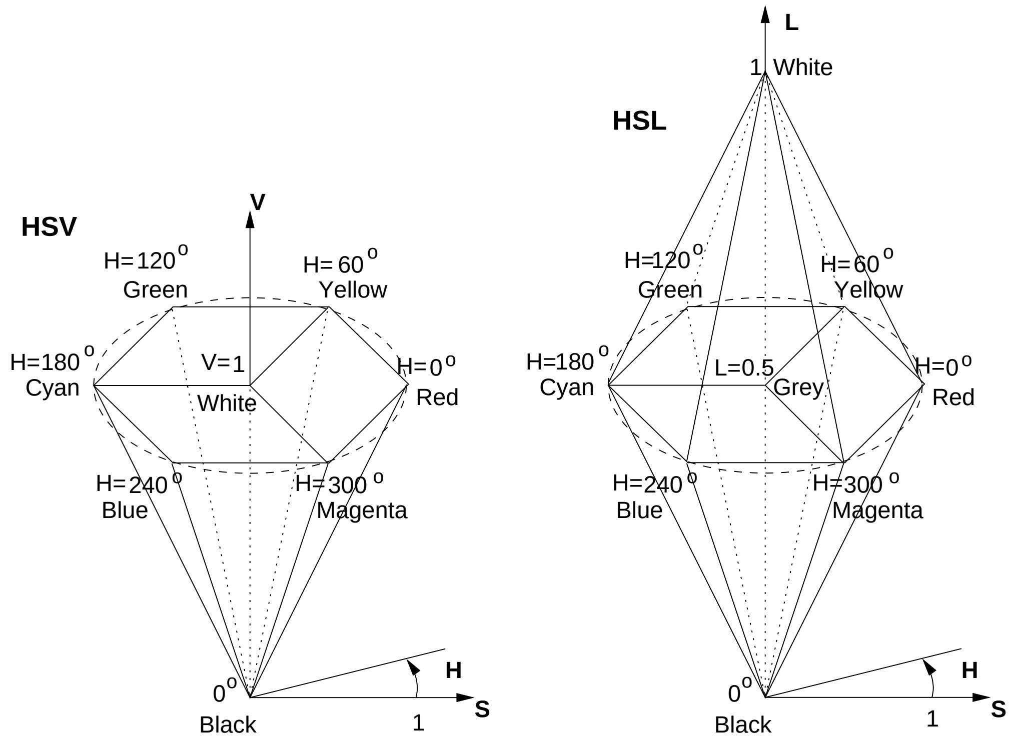


Figure A.4: HSV and HSL colour spaces.

HSV (hue, saturation, value). Both are defined with polar coordinate systems. HSV is represented by a hexcone where Hue is the angle around the vertical axis, S is the distance from the central axis and V is the distance along the vertical axis. Primary and secondary pure colours are fully saturated ($S = 1$). As illustrated in Figure A.4, starting from $H = 0^o$ (which represents pure red), a secondary or primary colour is located at each 60^o of hue. Complementary colours are 180^o opposite one another measured by H. Colours along the vertical axis have zero saturation, i.e., grey scale values. Note that when $S = 0$, the value of H is irrelevant [Jac01], [Smi78].

HSL colour space is a double hexcone and can be thought of as a deformation of the HSV space. The distinction between HSV and HSL lays in the representation of brightness information, which determines the distribution and dynamic range of both the brightness (L or V) and saturation (S). In practice, the HSL colour space is best for grey level image processing and also for representing objects in such a way that colour images can be distinguished even in monochrome images (e.g. to show colour cartoons on black-and-white TV receivers), whereas the HSV image space is a better representation for colour processing [Jac01].

As described in [RMG98], [AP96], and [ZYW00], on performing skin detection, the brightness channel is discarded and the HS space is used instead. Therefore, there is no significant difference between HSV and HSL in this application [Bow99].

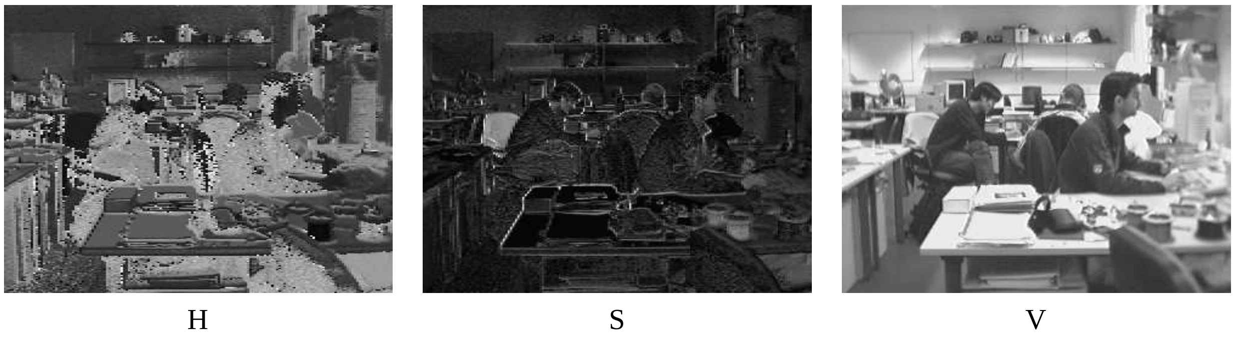


Figure A.5: HSV channels from Figure A.1(a) shown separately.

Figure A.5 shows the H, S and V channels obtained from image from Figure A.1(a).

A.4 The YUV and YCbCr colour spaces

The YUV image space was created in order to make colour television broadcasts backwards-compatible with black and white TV receivers. The colour signal also needed to conserve bandwidth because three channels of RGB data would not fit into the limited broadcast signal bandwidth. The Y channel describes Luma, the range of value between dark and light, which is the signal shown in black and white televisions. The U and V chrominance channels subtract the Luminance values from Red (U) and Blue (V) to represent the colour only information (without brightness) [Mal02]. Thence, the basic conversion equation from RGB to YUV is:

$$\begin{aligned} Y &= 0.3R + 0.6G + 0.1B \\ U &= B - Y \\ V &= R - Y \end{aligned} \tag{A.3}$$

The coefficients used to obtain luma are the same as those used for the NTSC standard conversion from RGB to grey level images [Poy96]. These coefficients are based on psychovisual experiments that estimated the proportion of red, green and blue that we perceive. It is shown that approximately 65% of all the cones in the human eye are sensitive to green light, 33% are sensitive to red light and only about 2% are sensitive to blue, but the blue cones are the most sensitive [WS00].

The YCbCr colour space was developed as part of ITU-R BT.301 during the development of a world-wide digital component video standard. This colour space was extensively used in the development of the JPEG standard, and was used for skin colour detection by several research projects, including the Pfinder [WADP97].

As shown in equation A.4, YCbCr is a scaled and zero-shifted version of the YUV, so that the chrominance values are always positive [PM93]:

$$\begin{aligned} Cb &= \frac{U}{2} + 0.5 \\ Cr &= \frac{V}{1.6} + 0.5, \end{aligned} \tag{A.4}$$

for U ranging between $[-0.9, 0.9]$ and V ranging between $[-0.7, 0.7]$, which are the ranges obtained from the conversion from RGB $\in [0, 1]$. So the range of Cb and Cr are $(0.05, 0.95)$ $(0.06, 0.94)$, respectively. For digital 8-bits values of U and V, a 128 shift is employed, rather than 0.5.

Figure A.6 shows the RGB colour cube in the YCbCr colour space. It shows that not all the possible values in the triple $[Y, Cb, Cr]$ represent possible RGB colours. Therefore, special care must be taken to about overflow or underflow in RGB, when converted from YCbCr. Brightness normalisation is done by discarding the Y channel.

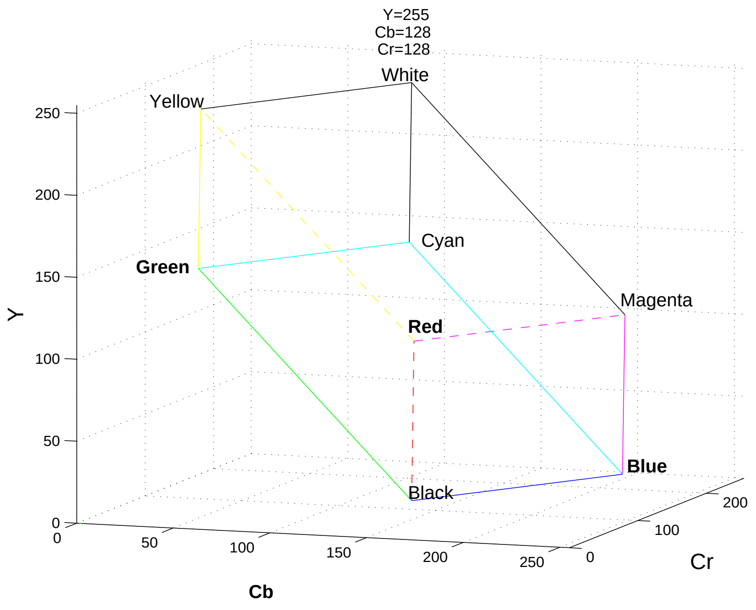
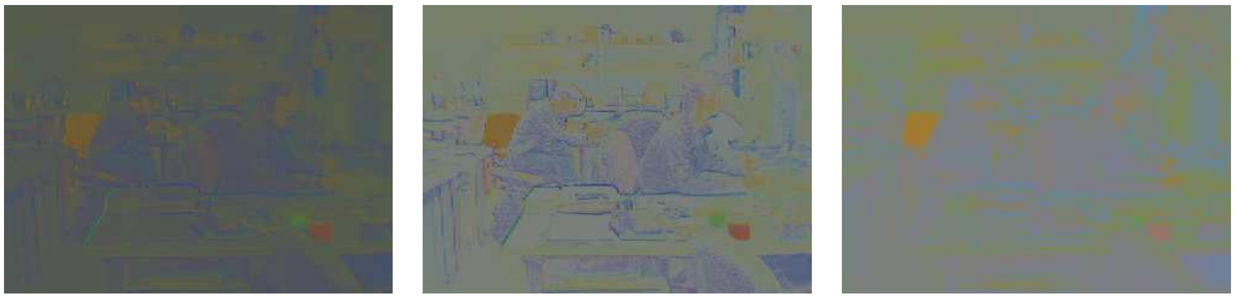


Figure A.6: The RGB colour cube in the YCbCr colour space, using 8 bit representation of values.

A.5 Visualising the colour spaces

To illustrate the effect of brightness normalisation using the above colour spaces, each method has been applied to the image of Figure A.1(a). An intermediate grey level (127) was chosen and the resulting images are shown in Figure A.7. Note that skin areas appear uniform and that shading information is lost for all the three methods.



(a) CIE Pure Colours (b) HSV (c) CbCr

Figure A.7: Resulting images after brightness normalisation of the image in Figure A.1(a) using the CIE colour space (a), the HSV without the V channel (b), and the YCbCr without the Y channel (c).

In order to illustrate the distribution of skin samples in the colour spaces, a database with images from 141 different people was used. This database is composed by hand images grabbed from seven volunteers, and the AR face detection database of the University of Purdue [MB98]. The Purdue database contains 134 people (men and women) from several ethnic groups. Background samples were obtained from the background regions in the images (e.g. people's clothing and other objects) and other images grabbed in the laboratory, as shown by some samples in Figure A.8. The camera used in the acquisition had the automatic colour and brightness balance.

Skin and background regions of the images in this database were manually selected in order to obtain the data set. After the training process, more than 0.5 million samples of skin and more than 1.2 million samples of background were obtained. Figure A.9 shows the skin and background samples in the RGB colour space.

Figure A.10(a) shows the plot of only skin samples in the RGB colour space. Note that the samples are more spread in the direction of the brightness variation. The directions of global variation of the sample data can be evaluated by performing Principal Component Analysis in this space [dCJ01], [Mar02].

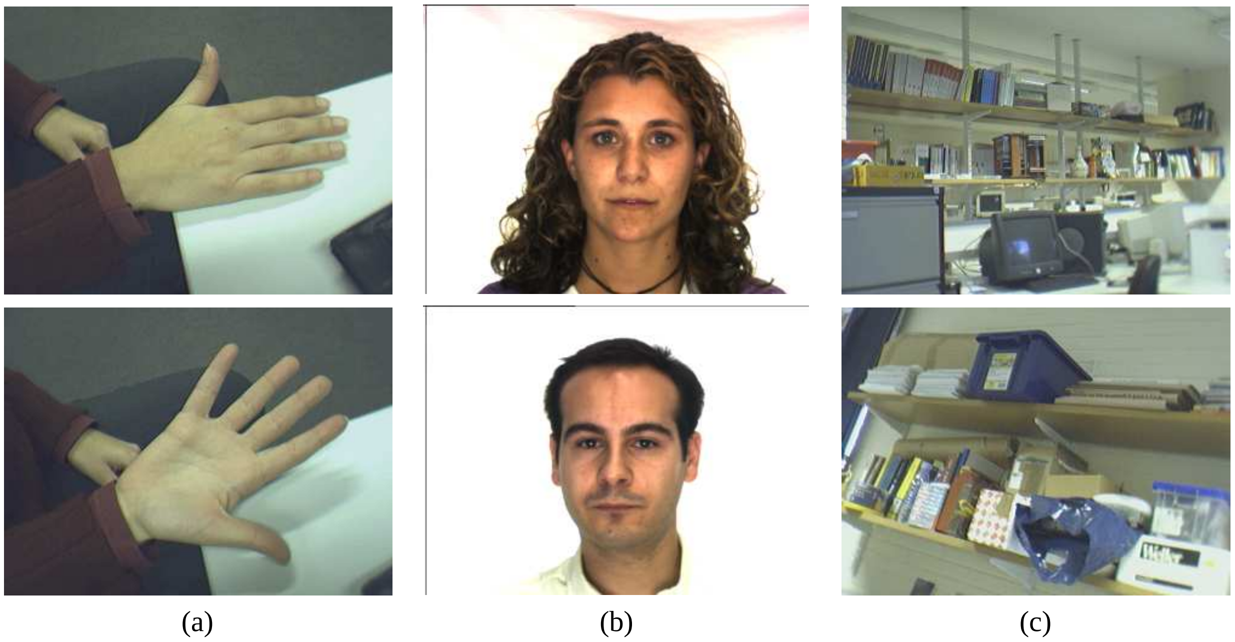


Figure A.8: Database samples: (a) hand images from local volunteers; (b) faces from the AR database; (c) background.

The eigenvector of the covariance matrix of the samples which is associated to the largest eigenvalue is oriented according to the largest variation of the data set. The second eigenvectors points to the direction that is perpendicular to the first, and has the second largest variation of the data, and so on. The eigenvectors of the skin colour database are shown in Figure A.10(b). The angle between the first eigenvector and the vector that points to the direction of the brightness variation is only 3.35 degrees. This confirms that it is necessary to use a normalised colour space or remove brightness information in order to get a more compact cluster of skin samples.

To illustrate the compression of the skin colour cluster in normalised colour spaces, Figures A.11, A.12, A.13 show the skin and background samples in the CIE, HSV and YUV chromatic spaces, respectively.

In comparison to the plots of the skin and background samples in the full colour spaces, the plots in normalised planes illustrate that such projections lead to lower dimensional spaces with more compact skin colour samples, improving the separability between them and background samples.

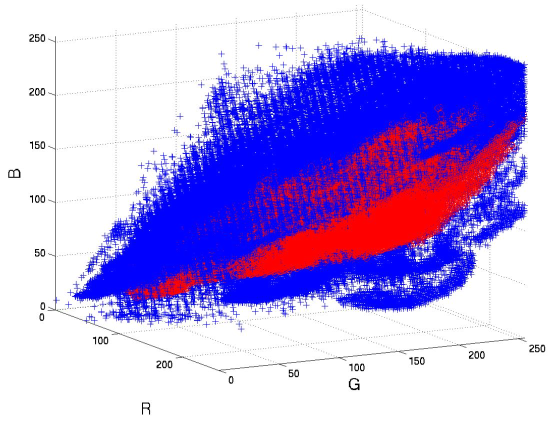
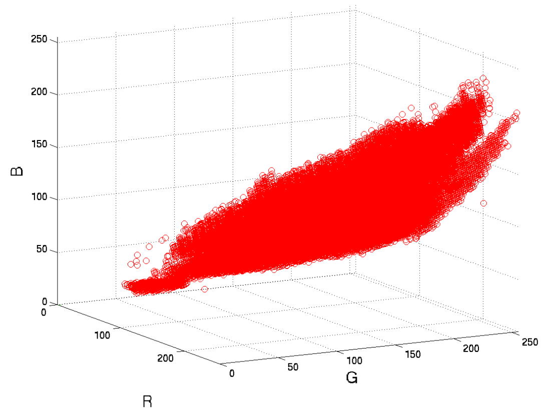
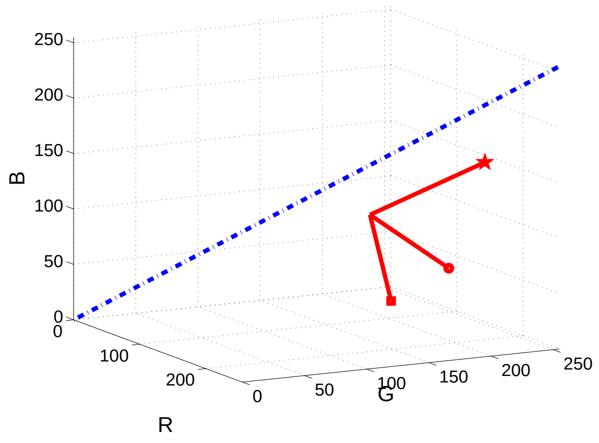


Figure A.9: Colour samples in the RGB space: skin (red circles) and background (blue crosses) samples plot together.



(a)



(b)

Figure A.10: Variation of skin samples: (a) Skin colour samples in the RGB space; (b) Eigenvectors of the skin samples in the RGB space in their respective mean position. The first, the second and the third eigenvectors are indicated by a star, a square and a circle in its end, respectively. The dashed line indicates the grey level (brightness) direction.

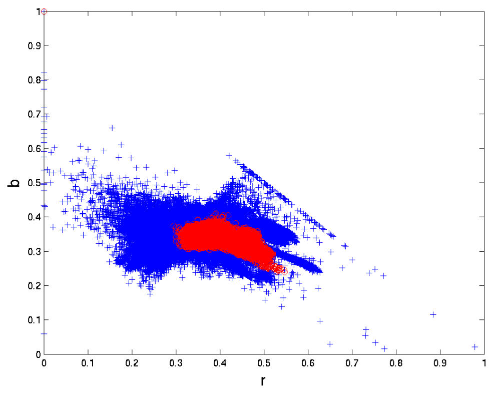
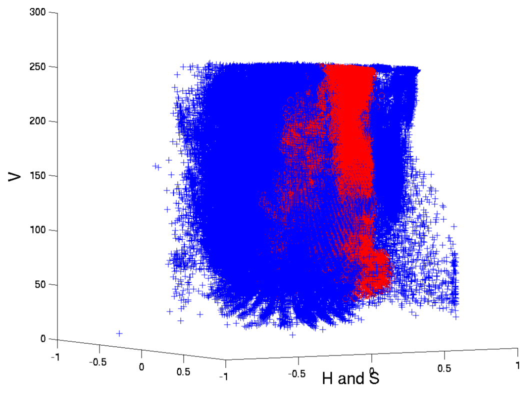
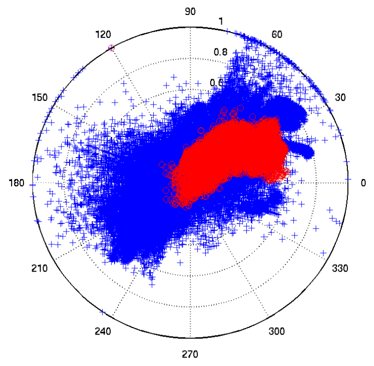


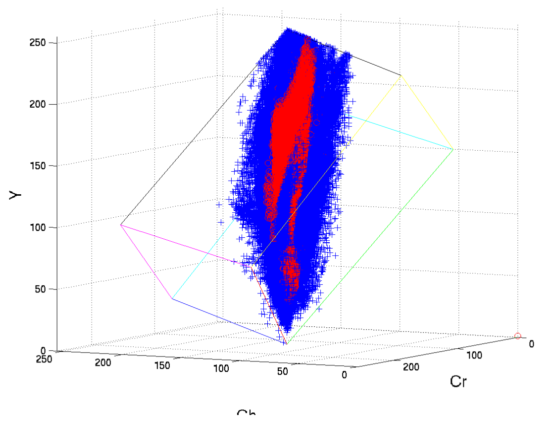
Figure A.11: Skin (red circles) and background (blue crosses) in the CIE chromatic space.



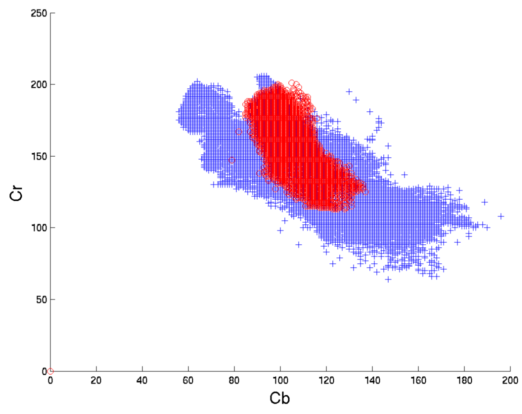


(a) (b)

Figure A.12: Skin (red circles) and background (blue crosses) shown in the HSV colour space (a); and their projection into the HS plane (b).



(a)



(b)

Figure A.13: Skin (red circles) and background (blue crosses) shown in the YCbCr colour space (a); and their projection into the CbCr plane (b).

B

Adjoint transformation in DCT

This appendix complements information of Chapter 6 about the adjoint transformation used by Drummond and Cipolla.

Consider two frames a and b where points are related by the homogeneous transformation

$$\mathbf{X}^b = \mathtt{T}_a^b \mathbf{X}^a = \begin{pmatrix} \mathtt{R}_a^b & \mathbf{t}_{ab} \\ \mathbf{0}^\top & 1 \end{pmatrix} \mathbf{X}^a \,. \tag{B.1}$$

To derive the effect of changing frame on the screw vector, $\boldsymbol{\alpha}$, consider writing the scene velocity in frame b in two different ways

$$\begin{pmatrix} \dot{\mathbf{X}}^b \\ 0 \end{pmatrix} = \sum_i \alpha_i^b \mathtt{G}_i \mathtt{T}_a^b \begin{pmatrix} \mathbf{X}^a \\ 1 \end{pmatrix} = \mathtt{T}_a^b \sum_i \alpha_i^a \mathtt{G}_i \begin{pmatrix} \mathbf{X}^a \\ 1 \end{pmatrix} \tag{B.2}$$

indicating that

$$\sum_i \alpha_i^b \mathtt{G}_i = \mathtt{T}_a^b \sum_i \alpha_i^a \mathtt{G}_i \mathtt{T}_b^a \,. \tag{B.3}$$

Recalling that $\boldsymbol{\alpha} = (\boldsymbol{\omega}^\top \boldsymbol{v}^\top)^\top$, and the earlier expressions for $\mathtt{G}_i$, Eq. (B.3) is just

$$\begin{pmatrix} [\boldsymbol{\omega}^b]_\times & \boldsymbol{v}^b \\ \mathbf{0}^\top & 0 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_a^b & \mathbf{t}_{ab} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} [\boldsymbol{\omega}^a]_\times & \boldsymbol{v}^a \\ \mathbf{0}^\top & 0 \end{pmatrix} \begin{pmatrix} \mathtt{R}_b^a & -\mathtt{R}_b^a \mathbf{t}_{ab} \\ \mathbf{0}^\top & 1 \end{pmatrix} , \tag{B.4}$$

where $[\boldsymbol{\omega}]_\times$ is the antisymmetric matrix such that $[\boldsymbol{\omega}]_\times \mathbf{r} = \boldsymbol{\omega} \times \mathbf{r}$. Hence

$$[\boldsymbol{\omega}^b]_\times = \mathtt{R}_a^b [\boldsymbol{\omega}^a]_\times \mathtt{R}_b^a \tag{B.5}$$

which is equivalent to

$$\boldsymbol{\omega}^b = \mathtt{R}_a^b \boldsymbol{\omega}^a \,. \tag{B.6}$$

Also

$$\boldsymbol{v}^b = -\mathtt{R}_a^b[\boldsymbol{\omega}^a]_\times \mathtt{R}_b^a \mathbf{t}_{ab} + \mathtt{R}_a^b \boldsymbol{v}^a = [\mathbf{t}_{ab}]_\times \mathtt{R}_a^b \boldsymbol{\omega}^a + \mathtt{R}_a^b \boldsymbol{v}^a \ . \tag{B.7}$$

The relationship between the screws is defined as

$$\boldsymbol{\alpha}^b = \mathrm{Ad}(\mathtt{T}_a^b)\boldsymbol{\alpha}^a \tag{B.8}$$

and hence using Eqs. (B.5,B.7) the adjoint is given by

$$\mathrm{Ad}(\mathtt{T}_a^b) = \begin{pmatrix} \mathtt{R}_a^b & \mathtt{0} \\ [\mathbf{t}_{ab}]_\times \mathtt{R}_a^b & \mathtt{R}_a^b \end{pmatrix} \tag{B.9}$$

Eq. (B.9) agrees with Drummond and Cipolla, given that they recover $\boldsymbol{\alpha} = (\boldsymbol{v}^\top \ \boldsymbol{\omega}^\top)^\top$. Though a minus sign appears missing from the definition of their antisymmetric matrix $[\mathbf{t}]_\wedge$.

As the measurement vector $\mathbf{d}$ is an invariant, $\mathtt{F}^b\boldsymbol{\alpha}^b = \mathtt{F}^a\boldsymbol{\alpha}^a$ and so

$$\mathtt{F}^b = \mathtt{F}^a \ \mathrm{Ad}(\mathtt{T}_a^b)^{-1} \tag{B.10}$$

which gives

$$\mathtt{C}^b = \mathtt{F}^{b\top}\mathtt{F}^b = \mathrm{Ad}(\mathtt{T}_a^b)^{-\top}\mathtt{C}^a \ \mathrm{Ad}(\mathtt{T}_a^b)^{-1} \ . \tag{B.11}$$

This differs from the equivalent in Drummond and Cipolla, a difference which may arise because equation (29) in ref [DC02] states $\mathtt{T}_a^b\mathtt{G}_i(\mathtt{T}_a^b)^{-1} = \sum_j \mathrm{Ad}(\mathtt{T}_a^b)_{ij}\mathtt{G}_j$, in contradiction with the later (agreed) statement in equation (32) in ref [DC02] that $\boldsymbol{\alpha}^b = \mathrm{Ad}(\mathtt{T}_a^b)\boldsymbol{\alpha}^a$.

Bibliography

[AASK04] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. Boostmap: A method for efficient approximate similarity rankings. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Washington DC, June 17 - July 2*, 2004.

[AC97] J.K. Aggarwal and Q. Cai. Human motion analysis: A review. In *Proc IEEE Nonrigid and Articulated Motion Workshop*, pages 90–102, San Juan, Puerto Rico, June, 16 1997.

[Ahm95] S. Ahmad. A usable real-time 3d hand tracker. In *Proceedings 28th Asilomar Conference on Signals, Systems and Computers*, pages 1257–1261. IEEE Computer Society Press, 1995.

[And01] Andrew Blake, Thomas Leung, Jim Rehg and Kentaro Toyama. *Workshop on Models versus Exemplars in Computer Vision (in conjunction with CVPR)*, Kauai, Hawaii, USA, December 2001. IEEE Computer Society.

[AP96] A. Azarbayejani and A. Pentland. Real-time self-calibrating stereo person tracking using 3d shape estimation from blob features. In *13th Int Conf on Pattern Recognition*, pages 627–632, volume C, Vienna, Austria, August 1996. IEEE Computer Society Press.

[AS02] V. Athitsos and S. Sclaroff. An appearence-based framework for 3d hand shape classification and camera viewpoint estimation. In *5th IEEE Int Conf on Face and Gesture Recognition*, May 2002.

[AS03] V. Athitsos and S. Sclaroff. Estimating 3d hand pose from a cluttered image. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Madison WI, June 18-20*, 2003.

[AT04a] A. Agarwal and W. Triggs. 3d human pose from silhouettes by relevance vector regression. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Washington DC, June 17 - July 2*. IEEE Computer Society Press, 2004.

[AT04b] A. Agarwal and W. Triggs. Learning methods for recovering 3d human pose from monocular images. Technical Report 5333, INRIA - Institut National de Recherche en Informatique et en Automatique, Grenoble, France, October 2004.

[AT04c] A. Agarwal and W. Triggs. Learning to track 3d human motion from silhouettes. In *Proc 21^{st} Int Conf on Machine Learning*, Banff, Canada, 2004.

[AT05] A. Agarwal and W. Triggs. Monocular human motion capture with a mixture of regressors. In *Workshop on Vision for Human Computer Interaction (V4HCI), in conjunction with CVPR*, San Diego, CA, June 2005. IEEE Computer Society Press.

[AT06a] A. Agarwal and W. Triggs. A local basis representation for estimating human pose from cluttered images. In *Proc Asian Conf on Computer Vision*, Lecture Notes in Computer Science, pages 50–59, Hyderabad, India, January 13–16 2006. Springer-Verlag.

[AT06b] A. Agarwal and W. Triggs. Recovering 3d human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):44–58, January 2006.

[AZ95] M. Armstrong and A. Zisserman. Robust object tracking. In *Proc Asian Conf on Computer Vision*, 1995.

[BA02] C. Breazeal and L. Aryananda. Recognizing affective intent in robot directed speech. *Autonomous Robots*, 12(1):83–104, 2002.

[Bay76] Bryce E. Bayer. Color imaging array. US Patent number US3971065, July 20 1976.

[BF95] U. Brockl-Fox. Real-time 3d interaction with up to 16 degrees of freedom from monocular image flows. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June, 26–28 1995.

[BGP96] E. Bernardo, L. Goncalves, and P. Perona. Monocular tracking of the human arm in 3d: Real time implamentation and experiments. In *Int Conf on Pattern Recognition*, 1996.

[BH95] A. Baumberg and D. Hogg. An adaptive eigenshape model. In D. Pycock, editor, *6th British Machine Vision Conference*, volume 1, pages 87–96, Birmingham, July 1995. BMVA.

[BI98] A. Blake and M. Isard. *Active Contours*. Springer Verlag, April 1998.

[BKMM$^+$04] M. Bray, E. Koller-Meier, P. Müller, L. Van Gool, and N. N. Schraudolph. 3d tracking by rapid stochastic gradient descent using a skinning model. In 1st *European Conference on Visual Media Production*, pages 59–68, London, UK, March 2004. Institution of Electrical Engineers, IEE.

[BKMV04] M. Bray, E. Koller-Meier, and L. Van Gool. Smart particle filtering for 3d hand tracking. In *Proc Sixth IEEE Int Conf on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004.

[BL98] L. Bretzner and T. Lindeberg. Use your hand as a 3d mouse, or, relative orientation from extended sequences of sparse point and line correspondence using the affine trifocal tensor. In *Proc 5th European Conf on Computer Vision, Freiburg, Germany*, volume 1406 of *LNCS*, pages 141–157. Springer, 1998.

[BM98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Santa Barbara, June 23-25*, pages 8–15. IEEE Computer Society Press, 1998.

[BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, April 2002.

[BMP04] C. Bregler, J. Malik, and K. Pullen. Twist based acquisition and tracking of animal and human kinematics. *Int Journal of Computer Vision*, 56(3):179–194, 2004. Kluwer Academic Publishers, The Netherlands.

[BOC$^+$98] C. Bregler, S. M. Omohundro, M. Covell, M. Slaney, S. Ahmad, D. A. Forsyth, and J. A. Feldman. Probabilistic models of verbal and body gestures. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*, pages 267–290. Cambridge University Press, 1998.

[Bow99] R. Bowden. *Learning Non-linear Models of Shape Motion*. PhD thesis, Department of Sustems Engineering, Brunel University, October 1999.

[BPH98] G. A. Berry, V. Pavlovic, and T. S. Huang. Battleview: A multimodal hci research application. In *Workshop on Perceptual User Interfaces*, San Francisco, November 1998.

[Bra98] G. R. Bradski. Real-time face and object tracking as a component of a perceptual user interface. In *Proc Workshop on Applications of Computer Vision*, pages 214–219, 1998.

[Bra99] M. Brand. Shadow puppetry. In *Proc 7th Int Conf on Computer Vision, Corfu, Greece*, pages 1237–1244. IEEE Computer Society Press, 1999.

[Bro83] Brown. *Introduction to Random Signal Analysis and Kalman Filtering*. Wiley, 1983.

[BS02] R. Bowden and M. Sarhadi. A non-linear model of shape and motion for tracking finger spelt american sign language. *Image and Vision Computing*, 20(9-10):597–607, August 2002.

[Car] Carnegie Mellon University. Graphics lab motion capture database. Created with funding from NSF EIA-0196217.

[Cas98] J. Cassel. A framework for gesture generation and interpretation. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*, pages 191–215. Cambridge University Press, 1998.

[CC03] S. C. Chan and J. Cooperstock. Hand tracking and gesture recognition. HTML, Last update: 22^{nd} September 2003. *http://www.cim.mcgill.ca/˜jer/research/gesture/*.

[CGH02] C-S Chua, H. Guan, and Y-K Ho. Model-based 3d hand posture estimation from a single 2d image. *Image and Vision Computing*, 20:191–202, 2002.

[CH88] M. J. Stephens C. Harris. A combined corner and edge detector. In 4^{th} *Alvey Vision Conference*, pages 147–151, Manchester, UK, August 31 - September 2 1988.

[CH98] R. Cipolla and N.J. Hollinghurst. A human-robot interface using pointing with uncalibrated stereo vision. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*, pages 97–110. Cambridge University Press, 1998.

[CMC04] A. I. Comport, E. Marchand, and F. Chaumette. Object-based visual 3D tracking of articulated objects via kinematic sets. In *Proc IEEE Workshop on Articulated and Non-Rigid Motion (in conjunction with CVPR)*, 2004.

[CMC06] A. I. Comport, E. Marchand, and F. Chaumette. Kinematic sets for real-time robust articulated object tracking. *Image and Vision Computing*, 2006.

[CnN98] D. Chai and K. n. Ngan. Locating facial region of a head-and-shouders colour image. In *3rd IEEE Int Conf on Automatic Face and Gesture Recognition*, pages 124–129, Nara, Japan, 1998.

[Coh] C. Cohen. The gesture recognition home page. HTML. *http://www.cybernet.com/˜ccohen/*.

[COK93] R. Cipolla, Y. Okamoto, and Y. Kuno. Robust structure from motion using parallax. In *Proc Fourth Int Conf on Computer Vision*, pages 374–382, Berlin, Germany, May 1993. IEEE Computer Society and ACM/SIGGRAPH, IEEE Computer Society Press.

[Cra89] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison Wesley Longman, Inc., January 1989.

[DBR00] J. Deutscher, A. Blake, and I. D. Reid. Articulated body motion capture by annealed particle filtering. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Hilton Head SC, June 13-15*, pages 2126–2133. IEEE Computer Society Press, 2000.

[DC99] T. Drummond and R. Cipolla. Visual tracking and control using lie algebras. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Fort Collins CO, June*, volume 2, pages 652–657. IEEE Computer Society Press, 1999.

[DC00] T. W. Drummond and R. Cipolla. Real-time tracking of multiple articulated structures in multiple views. *Lecture Notes in Computer Science*, II(1843):20–36, June/July 2000.

[DC01] T. Drummond and R. Cipolla. Real-time tracking of highly articulated structures in the presence of noisy measurements. In *Proc Int Conf in Computer Vision (ICCV)*, Vancouver, Canada, July 2001. IEEE Computer Society Press.

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

[dCJ01] L. da Fontoura Costa and R. M. Cesar-Jr. *Shape Analysis and Classification - Theory and Practice*. Image Processing. CRC Press, 2001.

[DDR01a] A. J. Davison, J. Deutscher, and I. Reid. Markless motion capture of complex full-body movement for character animation. In *Eurographics Workshop on Animation and Simulation*, 2001.

[DDR01b] J. Deutscher, A. Davison, and I. Reid. Automatic partitioning of high dimensional search spaces associated with articulated body motion capture. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, 2001.

[DEA99] D. C. Dryer, C. Eisbach, and W. S. Ark. At what cost pervasive? a social computing view of mobile computing systems. *IBM Systems Journal*, 38(4):652–676, May 1999.

[DGH$^+$02] Z. Duric, W. D. Gray, R. Heishman, F. Li, A. Rosenfeld, M. J. Schoelles, C. Schunn, and H. Wechsler. Integrating perceptual and cognitive modeling for adaptive and intelligent human computer interaction. *Proc IEEE*, 90(7):1272–1289, July 2002.

[DH73] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, USA, 1st edition, 1973.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, New York, USA, second edition, 2000.

[dM06] T. E. de Campos and D. W. Murray. Regression-based hand pose estimation from multiple cameras. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, New York NY, June 17-22*, 2006.

[dMM06] T. E. de Campos, W. W. Mayol Cuevas, and D. W. Murray. Directing the attention of a wearable camera by pointing gestures. In *Proc Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, Manaus, Brazil, 8–11 October 2006. IEEE Computer Society Press.

[DNBB99] J. Deutscher, B. North, B. Bascle, and A. Blake. Tracking through singularities and discontinuities by random sampling. In *Proc 7th Int Conf on Computer Vision, Corfu, Greece*, pages 1144–1149. IEEE Computer Society Press, 1999.

[DP95] T. Darrell and A.P. Pentland. Attention-driven expression and gesture analysis in an interactive environment. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June, 26–28 1995.

[dTM05] T. E. de Campos, B. J. Tordoff, and D. W. Murray. Linear recovery of articulated pose change: Comparing pre- and post-imposed constraints. Technical Report OUEL 2279/05,, Department of Engineering Science, Oxford University, 2005.

[dTM06] T. E. de Campos, B. J. Tordoff, and D. W. Murray. Recovering articulated pose: A comparison of two pre and postimposed constraint methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1), January 2006.

[EBN$^+$05] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. A review on vision-based full dof hand motion estimation. In *Workshop on Vision for Human Computer Interaction (V4HCI), in conjunction with CVPR*, San Diego, CA, June 2005. IEEE Computer Society Press.

[EZ05] M. Everingham and A. Zisserman. Identifying individuals in video by combining generative and discriminative head models. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, San Diego CA, June 20-25*, 2005.

[FAB$^+$98] W. T. Freeman, D. B. Anderson, P. A. Beardsley, C. N. Dodge, M. Roth, C. D. Wissman, W. S. Yerazunis, H. Kage, K. Kyuma, Y. Miyake, and K. I. Tanaka. Computer vision for interactive computer graphics. *IEEE Computer Graphics and Applications*, 18(3):42–53, May-June 1998.

[FAK03] H. Fillbrandt, S. Akyol, and K. F. Kraiss. Extraction of 3D hand shape and posture from image sequeces for sign language recognition. In *Int Workshop on Analysis and Modelling of Faces and Gestures (In conjunction with ICCV)*, pages 181–186, Nice, France, October, 17 2003. IEEE.

[Fau93] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, Cambridge, MA, USA, 1993.

[FB81] M. A. Fischler and R. C. Bolles. Random sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 26:381–395, 1981.

[FdC00] R. S. Feris, T. E. de Campos, and R. M. Cesar Jr. Detection and tracking of facial features in video sequences. In *Lecture Notes in Artificial Intelligence*, volume 1973, pages 129–137, Acapulco, Mexico, April 2000. Springer-Verlag Press.

[FGTK02] R. S. Feris, J. Gemmell, K. Toyama, and V. Kruger. Hierarchical wavelet networks for facial feature localization. In *Proc 5th IEEE Int Conf on Automatic Face and Gesture Recognition*, Washington, DC, USA, May 2002.

[FH00] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient matching of pictorial structures. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Hilton Head SC, June 13-15*, 2000.

[FH04] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int Journal of Computer Vision*, 2004.

[FP03] D. A. Forsyth and Jean Ponce. *Computer Vision: a modern approach*. Prentice Hall, 2003.

[FR04] H. Fei and I. Reid. Joint bayes filter: A hybrid tracker of non-rigid hand motion recognition. In T. Pajdla and J. Matas, editors, *Proc European Conference on Computer Vision*, number 3023 in Lecture Notes in Computer Science, pages 497–508, Prague, Czech Rep., May 2004. Springer-Verlag Berlin Heidelberg.

[Fri99] R. Frishholz. Face detection home page. HTML, February 1999. *http://home.t-online.de/home/Robert.Frischholz/face.htm*.

[FTR$^+$04] R. Feris, M. Turk, R. Raskar, K. Tan, and G. Ohashi. Exploiting depth discontinuities for vision-based fingerspelling recognition. In *Workshop on Real-Time Vision for Human-Computer Interaction*, Washington DC, USA, June 2004. IEEE. Associated with CVPR.

[GA90] C. Gosselin and J. Angeles. Singularity analysis of closed-loop kinematic chains. *IEEE Trans. Robotics and Automation*, 6(3):281–290, June 1990.

[Gav99] D. M. Gavrila. The visual analysis of human movement: a survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

[GD96] D. M. Gavrila and L. S. Davis. 3D model-based tracking of humans in action: a multiview approach. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, San Francisco CA, June 18-20*, pages 73–80. IEEE Computer Society Press, 1996.

[GdUP95] L. Goncalves, E. diBernardo, E. Ursella, and P. Perona. Monocular tracking of the human arm in 3d. In *5th Int Conf on Computer Vision*, pages 764–770, Los Alamitos, CA, USA, 1995. IEEE Computer Society Press.

[Gen92] D. B. Gennery. Visual tracking of known three-dimensional objects. *Int Journal of Computer Vision*, 7(3):243–270, 1992.

[GM83] C. M. Ginsberg and D. Maxwell. Graphical marionette. In *Proc Siggraph/Sigart Workshop on Motion*, pages 172–179, New York, USA, April 1983. ACM, ACM Press.

[GMHP04] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. In *Proc SIGGRAPH*, pages 522–531. ACM, 2004.

[GP99] D. M. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proc IEEE Int Conf on Computer Vision*, pages 87–93, 1999.

[GW00] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, Pearson Education Int., New Jersey, 2nd - international edition, 2000.

[Har92a] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*, pages 59–73, Cambridge, MA, USA, 1992. MIT Press.

[Har92b] C.H. Harris. Camera calibration. Technical report, Roke MAnor Research, Siemens, UK, 1992.

[HASW06] J. Han, G. M. Awad, A. Sutherland, and H. Wu. Automatic skin segmentation for gesture recognition combining region and support vector machine active learning. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[HCT95] A. Hill, T. F. Cootes, and C. J. Taylor. Active shape models and the shape approximation problem. In D. Pycock, editor, *6th British Machine Vision Conference*, volume 1, pages 157–166, Birmingham, July 1995. BMVA.

[HGT06] R. S. Feris H. Guan and M. Turk. The isometric self-organizing map for 3d hand pose estimation. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[HH96] T. Heap and D. Hogg. Towards 3d hand tracking using a deformable model. In *Proc 2nd IEEE Int Conf on Automatic Face and Gesture Recognition, Killington VT, 1996*, Killington, Vermont, USA, October 1996.

[HHD98] I. Haritaoglu, D. Harwood, and L. S. Davis. W4s: A real-time system for detecting and tracking people in $2\frac{1}{2}$d. In H. Burkhard and B. Neumann, editors, *5th European Conference on Computer Vision, Vol. I*, number 1406 in Lecture Notes in Computer Science, pages 877–892, Freiberg, Germany, June 1998. Springer.

[HKL06] B. W. Hwang, S. Kim, and S. W. Lee. A full-body gesture database for automatic gesture recognition. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[HLF99] N. Howe, M. Leventon, and W. Freeman. Bayesian reconstruction of 3d human motion from single-camera viceo. In *Neural Information Processing Systems (NIPS)*, 1999.

[HM99] J. J. Heuring and D. W. Murray. Modelling and copying human head movements. *IEEE Transactions on Robotics and Automation*, 15(6):1095–1108, 1999.

[Hog83] D. C. Hogg. Model-based vision: a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

[HOW96] Y. Hel-Or and M. Werman. Constraint fusion for recognition and localization of articulated objects. *Int Journal of Computer Vision*, 19(1):15–28, July 1996.

[HS90] C. Harris and C. Stennett. RAPiD – a video rate object tracker. In *Proc 1st British Machine Vision Conf, Oxford*, pages 73–78, 1990.

[HSS02] Y. Hamada, N. Shimada, and Y. Shirai. Hand shape estimation using sequence of multi-ocular images based on transition network. In 15^{th} *Int Conf on Vision Interface*, pages 362–368, Calgary, Alberta, Canada, May 27-29 2002. CIPPRS and IAPR.

[HTH00] P. Hong, M. Turk, and T.S. Huang. Gesture modeling and recognition using finite state machines. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition*, Grenoble, France, March 28–30 2000.

[Hu62] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. Information Theory, IT*, 8, 1962.

[HZ01] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2001. second printing.

[IB96] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proc 4th European Conf on Computer Vision, Cambridge, UK*, pages 343–356. Springer, 1996.

[IB98] M. Isard and A. Blake. Condensation: Conditional density propagation for visual tracking. *Int Journal of Computer Vision*, 29(1):5–28, 1998.

[Jac01] K. Jack. *Video Demystified*. LLH Technology Publishing, third edition, 2001.

[JBY96a] S. Ju, M. Black, and Y. Yacoob. Cardboard people: a parameterized model of articulated motion. In *Proc IEEE Int Conf Automatic Face And Gesture Recognition*, pages 38–44, Killington, 1996.

[JBY96b] S. X. Ju, M. J. Black, and Y. Yacoob. Cardboard people: a parametrized model of articulated motion. In *Proc 2nd IEEE Int Conf on Automatic Face and Gesture Recognition, Killington VT, 1996*, pages 38–44, 1996.

[JDM00] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, January 2000.

[JP97] T. S. Jebara and A. Pentland. Parametrized structure from motion from 3d adaptive feedback tracking of faces. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, pages 144–150, San Juan, Puerto Rico, June 1997.

[JR98] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. Technical report, Compaq Computer Corporation, Cambridge Research Laboratory, Massachusetts, USA, December 1998.

[JR02] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *Int Journal of Computer Vision*, 46(1):81–96, January 2002.

[JRM06] A. Just, Y. Rodriguez, and S. Marcel. Hand posture classification and recognition using the modified census transform. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[JS05] A. Jaimes and N. Sebe. Multimodal human computer interaction: A survey. In *IEEE Int Workshop on Human Computer Interaction (in Conjunction with ICCV)*, Beijing, China, October 15–21 2005. Invited paper.

[JT05] F. Jurie and W. Triggs. Creating efficient codebooks for visual recognition. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, San Diego CA, June 20-25*, 2005.

[JU97] S. J. Julier and J. K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *SPIE AeroSense Symposium*, Orlando, FL, USA, 1997.

[KCX06] M. Kato, Y.W. Chen, and G. Xu. Articulated hand tracking by PCA-ICA approach. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[KD05] C. Kemp and T. Drummond. Dynamic measurement clustering to aid real time tracking. In *Proc 10th Int Conf on Computer Vision, Beijing, China, Oct 15-21*. IEEE, 2005.

[KH95] C. Kervrann and F. Heitz. Learning structure and deformation modes of nonrigid objects in long image sequences. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June, 26–28 1995.

[KM96] I. A. Kakadiaris and D. Metaxas. 3D human body model acquisition from multiple views. In *Proc 5th Int Conf on Computer Vision, Boston, USA*, pages 618–623. IEEE Computer Society Press, 1996.

[KOKS01] T. Kurata, E. Okuma, M. Kourogi, and K. Sakaue. The hand mouse: Gmm hand-color classification and mean shift tracking. In *Second Int Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time (in conjunction with ICCV)*, pages 119–124, Vancouiver, Canada, July 2001.

[KT04a] M. Kölsch and M. Turk. Robust hand detection. In *Proc IEEE Int Conf on Automatic Face and Gesture Recognition*, May 2004.

[KT04b] M. Kölsh and M. Turk. Fast 2d hand tracking with flocks of features and multi-cue integration. In *IEEE Workshop on Real-Time Vision for Human-Computer Interaction, in association with CVPR*, Washington DC, USA, July 2004.

[KTZ04] M. Pawan Kumar, Phil H. S. Torr, and Andrew Zisserman. Extending pictorial structures for object recognition. In *Proc British Machine Vision Conference*, Kingston University, London, UK, September 2004. BMVA.

[Law04] N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In L. Saul S. Thrun and B. Schlkopf, editors, 16^{th} *Neural Information Processing Systems (NIPS)*. MIT Press, Cambridge, MA, 2004.

[LCO$^+$05] M. A. Lebedev, J. M. Carmena, J. E. O'Doherty, M. Zacksenhouse, C. S. Henriquez, J. C. Principe, and M. A. Nicolelis. Cortical ensemble adaptation to represent velocity of an artificial actuator controlled by a brain-machine interface. *Journal of Neuroscience*, 25(19):4681–4693, May 2005.

[Lev44] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[LF02] R. Lockton and A. Fitzgibbon. Real-time gesture recognition using deterministic boosting. In *Proc British Machine Vision Conference*, Cardiff, UK, September 2002. BMVA.

[LF05] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, September 2005.

[LH98] C. C. Lien and C. L. Huang. Model-based articulated hand motion tracking for gesture recognition. *Image and Vision Computing*, 16:121–134, 1998.

[LH00] F. Lathuilière and J. Y. Hervé. Visual tracking of hand posture with occlusion handling. In *Proc Int Conf on Pattern Recognition*, Barcelona, Spain, September 2000. IAPR, IEEE.

[Lie04] C. C. Lien. The scalable model-based hand posture analysis system. *Submitted to IEE Vision, Image and Signal Processing*, 2004.

[LK81] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc Imaging Understanding Workshop*, pages 121–130, 1981.

[LMSO03] S. Lu, D. Metaxas, D. Samaras, and J. Oliensis. Using multiple cues for hand tracking and model refinement. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, Madison, Wisconsin, USA, June 2003. IEEE Computer Society.

[Low92] D. G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int Journal of Computer Vision*, 8(2):113–122, August 1992.

[Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int Journal of Computer Vision*, 60(2):91–110, 2004.

[LPV06] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. A layered deformable model for gait analysis. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[LS99] D. D. Lee and H. S. Seung. Learning the parts of objects with nonnegative matrix factorization. *Nature*, 401:788–791, October 1999.

[LWH01] J. Lin, Y. Wu, and T. Huang. Modeling the constraints of human hand motion. In 5^{th} *Annual Federated Laboratory Symposium*, Maryland, USA, March 2001.

[Lyo02] D. E. Lyons. A qualitative approach to computer sign language recognition. Master's thesis, Department of Engineering Science, University of Oxford, Oxford, UK, August 2002. Supervised by M. Brady and I. Reid.

[Mal02] J. Maller. *Joe's Filters 3.0 for Final Cut Pro, FXScript Reference on RGB and YUV Color*. *http://www.joemaller.com/fcp/fxscript_yuv_color.shtml*, February 2002.

[Mar63] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Applied Math.*, 11:431–441, 1963.

[Mar02] B. Martinkauppi. *Face colour under varying illumination - analysis and applications*. PhD thesis, Department of Electrical and Information Engineering, University of Oulu, Oulu, Finland, August 2002. *http://herkules.oulu.fi/isbn9514267885/*.

[May04] W. W. Mayol Cuevas. *Wearable Visual Robots*. PhD thesis, University of Oxford, Department of Engineering Science, Michaelmas 2004.

[MB98] A. M. Martinez and R. Benavente. The AR face database. Technical Report CVC 24, Purdue University, June 1998. Available at *http://rvl1.ecn.purdue.edu/˜aleix/ar.html*.

[MDTM04] W. W. Mayol Cuevas, A. J. Davison, B. J. Tordoff, and D. W. Murray. Interaction between hand and wearable camera in 2d and 3d environments. In *Proc 15th British Machine Vision Conf, Kingston University, London*. British Machine Vision Association, September 2004. http://bmvc.king.ac.uk/.

[MG01] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81:231–267, 2001.

[MI00] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In D. Vernon, editor, *6th European Conference on Computer Vision*, number 1843 in Lecture Notes in Computer Science, pages 3–19, Dublin, Ireland, June/July 2000. Springer. Part II.

[MM01] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *Workshop on Models versus Examplars in Computer Vision, associated with CVPR*. IEEE Computer Society Press, 2001.

[MM02] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In A. Heyden et al., editor, *European Conference on Computer Vision*, number 2352 in Lecture Notes in Computer Science, page 666 ff., Copenhagen, Denmark, May 2002. Springer-Verlag Berlin Heidelberg.

[MN78] D. Marr and H.K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc Roy Soc Lond B*, 200:269–294, 1978.

[Moo01] M. Moore. Human facotors issues in direct brain-computer interfaces. In *Workshop on Perceptive User Interfaces*, Orlando, Forida, USA, November 2001. ACM. Invited Talk.

[MR98] D. Morris and J. Rehg. Singularity analysis for articulated object tracking. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, pages 289–297, 1998.

[MREM04] G. Mori, X. Ren, A. Efros, and J. Malik. Recovering human body configurations: Combining segmentation and recognition. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, Washington DC, USA, July 2004.

[MRT98] D. W. Murray, I. D. Reid, and R. L. Thompson. Real-time visual recovery of pose using line tracking in multiple cameras. In *Proc 9th British Machine Vision Conf, Southampton*, 1998.

[MTdC$^+$03] W. W. Mayol Cuevas, B. J. Tordoff, T. E. de Campos, A. J. Davison, and D. W. Murray. Active vision for wearables. In *Proc Eurowearable 2003*, Birmingham, September 2003. IEE.

[MTM00] W.W. Mayol Cuevas, B.J. Tordoff, and D.W. Murray. Wearable visual robots. In *Int Symposium on Wearable Computing*, Atlanta, GA, USA, 2000.

[MTM02] W.W. Mayol Cuevas, B.J. Tordoff, and D.W. Murray. Designing a miniature wearable visual robot. In *IEEE Int Conf on Robotics and Automation*, Washington DC, May 2002. IEEE Computer Society Press.

[Mur02] K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, Department of Computer Science, University of California, Berkeley, Fall 2002. He has papers with J Rehg on articulated figure tracking and with W Freeman on boosting. He let many MatLab toolkboxes available on line.

[NH01] K. Nickels and S. Hutchinson. Model-based tracking of complex articulated objects. *IEEE Trans. Robotics and Automation*, 17(1):28–36, 2001.

[NM65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[NR99] C. Nolker and H. Ritter. GREFIT: Visual recognition of hand postures. In *Proc 3rd Gesture Workshop (GW)*, volume 1739 of *Lecture Notes on Artificial Ingelligence (LNAI)*, pages 61–72, Gif-sur-Yvette, France, March, 17–19 1999.

[NSMO96] K. Nirei, H. Saito, M. Mochimaru, and S. Ozawa. Human hand tracking from binocular image sequences. In *Proc 22nd Intl. Conf. on Industrial Electronics, Control, and Instrumentation (IECON)*, pages 297–302, Taiwan, August 5–9 1996.

[O'R98] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, 1998.

[OS03a] G. Ohashi and Y. Shimodaira. Edge-based feature extraction method and its application to image retrieval. In *7th World Multi-conference on Systemics, Cybernetics and Informatics*, Florida, USA, 2003.

[OS03b] G. Ohashi and Y. Shimodaira. Edge-based feature extraction method and its application to image retrieval. *Journal of Systemics, Cybernetics and Informatics*, 2(6), 2003.

[OZ00] R. O'Hagan and A. Zelinsky. Visual gesture interfaces for virtual environments. In *1st Australasian User Interface Conference*, Canberra, 31 January – 3 February 2000.

[PB98] C. Pinhanez and A. Bobick. Human action detection using PNF propagation of temporal constraints. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Santa Barbara, June 23-25*, 1998.

[PCI03] S. Perrin, A. Cassinelli, and M. Ishikawa. Laser-based finger tracking system suitable for MOEMS integration. In *Proc Image and Vision Computing (IVCNZ)*, Massey University, Palmerston North, New Zealand, November 2003. New Zealand Computer Society.

[Pen00] A. Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):107–119, January 2000.

[PF02] R. Plänkers and P. Fua. Model-based silhouette extraction for accurate people tracking. In *Proc 7th European Conf on Computer Vision, Copenhagen*, pages 325–339, 2002.

[PM93] W.B. Pennebaker and J. L. Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1993.

[Por02] M. Porta. Vision-based user interfaces: Methods and applications. *Int J. Human-Computer Studies*, 57:27–73, 2002.

[Poy96] C. A. Poynton. *A Technical Introduction to Digital Video*. John Wiley & Sons, 1996.

[PSH97] V. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

[PTVF88] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1988.

[Que95] F. Quek. Eyes in the interface. *Image and Vision Computing*, 13(6):511–525, 1995.

[RASS01] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff. 3d hand pose reconstruction using specialized mappings. In *Proc 8th Int Conf on Computer Vision, Vancouver, Canada, July*, 2001.

[Reh95] J. Rehg. *Visual Analysis of Articulated Objects with Applications to Hand Tracking*. PhD thesis, School of Computer Science, Carnegie Mellon University, April 1995.

[RF03] D. Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Madison WI, June 18-20*, 2003.

[RK94] J. Rehg and T. Kanade. Digiteyes: Vision-based human hand tracking. In Jan-Olof Eklundh, editor, *3rd European Conference on Computer Vision*, volume 800 of *Lecture Notes in Computer Science*, Stockholm, Sweden, May 1994. Springer.

[RK95a] J. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *5th Int Conf on Computer Vision*, pages 612–617, Cambridge, MA, USA, June 1995.

[RK95b] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc 5th Int Conf on Computer Vision, Boston, USA*, pages 612–617. IEEE Computer Society Press, 1995.

[RL87] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection*. Wiley, New York, 1987.

[RL00] M. Ringer and J. Lasenby. Modelling and tracking articulated motion from multiple camera views. In *Proc British Machine Vision Conference*, University of Bristol, UK, September 2000. BMVA.

[RMG98] Y. Raja, S. McKenna, and S. Gong. Segmentation and tracking using colour mixture models. *Lecture Notes in Computer Science*, I(1351):607–614, January 1998.

[RMK03] J. M. Rehg, D. D. Morris, and T. Kanade. Ambiguities in visual tracking of articulated objects using two- and three-dimensional models. *Int Journal of Robotics Research*, 22(6):393–418, 2003.

[RN98] B. Reeves and C. Nass. *The Media Equation — How People Treat Computers, Television, and New Media Like Real People and Places*. Cambridge University Press and CSLI Publications, 1998.

[RO00] G. Rätsch and T. Onoda. Boosting research site. HTML, available from October 2000. *http://www.boosting.org/*.

[RS06] R. Rosales and S. Sclaroff. Combining generative and discriminative models in a framework for articulated pose estimation. *Int Journal of Computer Vision*, 67(3):251–276, March 2006.

[RST02] R. Ronfard, C. Schmid, and W. Triggs. Learning to parse pictures of people. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *7th European Conference on Computer Vision, Vol. IV*, number 2353 in Lecture Notes in Computer Science, pages 700–714, Copenhagen, Denmark, May 2002. Springer.

[RTF$^+$04] R. Raskar, K. H. Tan, R. S. Feris, J. Yu, and M. Turk. Non-photorealistic camera: Depth edge detection and stylized rendering using multi-flash imaging. In *Proc SIGGRAPH*. ACM, August 2004.

[SBF00] H. Sidenbladh, M J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In David Venon, editor, *European Conference on Computer Vision*, number 1843 in Lecture Notes in Computer Science, pages 702–718, Dublin, Ireland, June/July 2000. Springer-Verlag.

[SBR$^+$04] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose-limbed people. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Washington DC, June 17 - July 2*. IEEE Computer Society Press, 2004.

[Seb84] G.A.F Seber. *Multivariate Observations*. Wiley, New York, 1984.

[SF96] D. Saxe and R. Foulds. Toward robutst skin identification in video images. In *2nd IEEE Int Conf on Automatic Face and Gesture Recognition*, pages 379–384, Killington, Vermont, USA, 1996.

[SISB03] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *Neural Information Processing Systems (NIPS)*, 2003. Available at *http://books.nips.cc/*.

[SK98] J. Segen and S. Kumar. Gesturevr: Vision-based 3d hand interface for spatial interaction. In *Proc 6th ACM Int Conf on Multimedia*, pages 455–464, Bristol, UK, 1998.

[SK99] J. Segen and S. Kumar. Shadow gestures: 3d hand pose estimation using a single camera. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Fort Collins CO, June*, 1999.

[SK00] J. Segen and S. Kumar. Look ma, no mouse! *Communications of the ACM*, 43(7):102–109, July 2000.

[SKK00] Y. Sato, Y. Kobayashi, and H. Koike. Fast tracking of hands and fingertips in infrared images for augmented desk interface. In *4th IEEE Int Conf Automatic Face and Gesture Recognition*, Grenoble, France, March, 28–30 2000.

[SKLM05] C. Sminchisescu, A. Kanaujia, Z. Lio, and D. Metaxas. Discriminative density propagation for 3d human motion estimation. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, San Diego CA, June 20-25*, 2005.

[SKS01] N. Shimada, K. Kimura, and Y. Shirai. Real-time 3-D hand pose estimation based on 2-D appearance retrieval using monocular camera. In *Recognition, Analysis and Tracking of Faces and Gestures in Realtime Systems*, pages 23–30, 2001.

[SLM$^+$02] T. Starner, B. Leibe, D. Minnen, T. Westeyn, A. Hurst, and J. Weeks. Computer vision-based gesture tracking, object tracking, and 3d reconstruction for augmented desks. *Int Journal of Machine Vision and Applications*, 2002.

[SMC01] B. Stenger, P. R. S. Mendonça, and R. Cipolla. Model-based hand tracking using an unscented kalman filter. In Tim Cootes and Chris Taylor, editors, *British Machine Vision Conference*, volume 1, pages 63–72, The University of Manchester, UK, September 2001. BMVA.

[Smi78] A. R. Smith. Color gamut transform pairs. In *5th Int Conf on Computer Graphics and Interactive Techniques*, pages 12–19, New York, USA, 1978. Siggraph - ACM Special Interest Group on Computer Graphics and Interactive Techniques, ACM Press.

[SS96] N. Shimada and Y. Shirai. 3-d hand pose estimation and shape model refinement from a monocular image sequence. In *Proc Int Conf on Visual Systems and Multimedia*, Gifu, Japan, September, 18–20 1996.

[SSK01] Y. Sato, M. Saito, and H. Koike. Real-time input of 3d pose and gestures of a user's hand and its applications for hci. In *IEEE Virtual Reality Conference (VR)*, Yokohama, Japan, March 13–17 2001.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, Seattle, USA, June 1994.

[ST01a] C. Sminchisescu and W. Triggs. Covariance scaled sampling for monocular 3D body tracking. In *Proc IEEE Conf on Computer Vision and Pattern Recognition, Kauai HI, Dec 8-14*, pages I:447–454. IEEE Computer Society Press, 2001.

[ST01b] C. Sminchisescu and W. Triggs. A robust multiple hypothesis approach to monocular human motion tracking. Technical Report 4208, Institut National de Recherche en Informatique et en Automatique, France, June 2001.

[ST02] C. Sminchisescu and W. Triggs. Building roadmaps of local minima of visual models. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *7th European Conference on Computer Vision, Vol. I*, number 2350 in Lecture Notes in Computer Science, pages 566–582, Copenhagen, Denmark, May 2002. Springer.

[Ste04] B. D. R. Stenger. *Model-Based Hand Tracking Using a Hierarchical Bayesian Filter*. PhD thesis, University of Cambridge, March 2004.

[Str88] G. Strang. *Linear Algebra and its Applications*. Saunders College Publishing / Harcourt College Publishers, USA, 3rd edition, 1988.

[STTC03] B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Filtering using a tree-based estimator. In *Proc 9th Int Conf on Computer Vision, Nice, France, Oct 13-16*. IEEE Computer Society Press, 2003.

[STTC04] B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Hand pose estimation using hierarchical detection. In N. Sebe et al., editor, *Proc Workshop on Human-Computer Interaction – ECCV*, Lecture Notes in Computer Science, pages 105–116, Prague, Czech Republic, 2004. Springer-Verlag Berlin Heidelberg.

[Stu92] D. J. Sturman. *Whole-Hand Input*. PhD thesis, Media Arts and Science Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA, February 1992. *http://xenia.media.mit.edu/ djs/thesis.ftp.html*.

[Sul92] G. D. Sullivan. Visual interpretation of known objects in constrained scenes. *Phil Trans Roy Soc Lond*, 337:361–370, 1992.

[SVD03] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proc 9th Int Conf on Computer Vision, Nice, France, Oct 13-16*, 2003.

[SWP98a] T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk and computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375, 1998.

[SWP98b] T. Starner, J. Weaver, and A. Pentland. A wearable computer based american sign language recognizer. *Lecture Notes in Artificial Intelligence, Springer-Verlag*, 1458:84–97, 1998.

[SZ94] D. J. Sturman and D. Zeltzer. A survey of glove-based input. *IEEE Computer Graphics and Applications*, 14(1):30–39, January 1994.

[TB02] K. Toyama and A. Blake. Probabilistic tracking with examplars in a metric space. *Int Journal of Computer Vision*, 48(1):9–19, 2002.

[The03] The TargetJr Consortium. VXL: The Vision-something-Libraries. Available online from http://vxl.sourceforge.net, 2000-2003.

[Tip01] M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, June 2001. http://jmlr.csail.mit.edu/.

[TK92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *Int Journal of Computer Vision*, 9(2):137–154, 1992.

[TLC$^+$98] C. J. Taylor, A. Lanitis, T. F. Cootes, G. Edwards, and T. Ahmad. Model-based interpretation of faces and hand gestures. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*, pages 217–233. Cambridge University Press, 1998.

[TM02] B. J. Tordoff and D. W. Murray. Guided sampling and consensus for motion estimation. In *Proc 7th European Conf on Computer Vision, Copenhagen*, pages 82–96, 2002.

[TM05] B. J. Tordoff and D. W. Murray. Guided-MLESAC: Faster image transform estimation by using mathcing priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10), October 2005.

[TMdM02a] B. J. Tordoff, W. W. Mayol Cuevas, T. E. de Campos, and D. W. Murray. Head pose estimation for wearable robot control. In *Proc 13th British Machine Vision Conf, Cardiff*, pages II:807–816, 2002.

[TMdM02b] B. J. Tordoff, W. W. Mayol Cuevas, T. E. de Campos, and D. W. Murray. Head pose estimation for wearable robot control. In *Proc 13th British Machine Vision Conference*, Cardiff, Wales, September 2002. British Machine Vision Association.

[TNS$^+$06] A. Thayananthan, R. Navaratnam, B. Stenger, P.H.S. Torr, and R. Cipolla. Multivariate relevance vector machines for tracking. In *Proc 9th European Conf on Computer Vision, Graz, Austria*, Lecture Notes in Computer Science. Springer-Verlag, 2006.

[Tor02] B. J. Tordoff. *Active Control of Zoom for Computer Vision*. PhD thesis, Robotics Research Group, Department of Engineering Science, University of Oxford, Oxford, UK, December 2002.

[TRMM01] R. L. Thompson, I. D. Reid, L. A. Munoz, and D. W. Murray. Providing synthetic views for teleoperation using visual pose tracking in multiple cameras. *IEEE Transactions on Systems, Man, and Cybernetics*, 31(1):43–54, 2001.

[TSTC03] A. Thayananthan, B. Stenger, P.H.S. Torr, and R. Cipolla. Learning a kinematic prior for tree-based filtering. In *Proc British Machine Vision Conf*, Norwich, UK, September 2003. BMVA.

[TT97] M. Turk and Y. Takebayashi, editors. *1st Workshop on Perceptual User Interfaces*, Banff, Alberta, Canada, October 1997. ACM.

[Tur04] M. Turk. Computer vision in the interface. *Communications of the ACM*, 47(1):61–67, January 2004.

[Tv01] J. Triesch and C. von der Malsburg. A system for person-independent hand posture recognition against complex backgrounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1149–1453, December 2001.

[UMKR96] A. Utsumi, T. Miyasato, F. Kishino, and R.Nakatsu. Hand gesture recognition system using multiple cameras. In *Proc 13th Int Conf on Pattern Recognition, Vienna, Austria*, pages 667–671, 1996.

[vHB01] C. von Hardenberg and F. Bérard. Bare-hand human-computer interaction. In *Workshop on Perceptive User Interfaces*, Orlando, FL, USA, November 2001. ACM.

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc of the IEEE Conf on Computer Vision and Pattern Recognition*, 2001.

[VM96] C. Vogler and D. Metaxas. ASL recognition based on a coupling between HMMs and 3D motion analysis. In *Proc 6th Int Conf on Computer Vision, Bombay, India, January 4-7*, pages 363–369. IEEE Computer Society Press, 1996.

[WADP97] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[WAP06] S. Wagner, B. Alefs, and C. Picus. Framework for a portable gesture interface. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[Wat93] R. Watson. A survey of gesture recognition techniques. Technical Report TCD-CS-93-11, Trinity College, Dublin, Ireland, July, 17 1993.

[WBAS03] T. Westeyn, H. Brashear, A. Atrash, and T. Starner. Georgia tech gesture toolkit: Supporting experiments in gesture recognition. In Sharon L. Oviatt, Trevor Darrell, Mark T. Maybury, and Wolfgang Wahlster, editors, *Proc 5th Int Conf on Multimodal Interfaces (ICMI)*, Vancouver, BC, Canada, November 5-7 2003. ACM.

[WGH$^+$03] K. Warwick, M. N. Gasson, B. D. Hutt, I. Goodhew, P. J. Kyberd, B. J. Andrews, P. Teddy, and A. Shad. The application of implant technology for cybernetic systems. *Archives of Neurology*, 60(10):1369–1373, October 2003.

[WH99a] Y. Wu and T. S. Huang. Capturing articulated human hand motion: A divide-and-conquer approach. In *Proc 7th Int Conf on Computer Vision, Corfu, Greece*, pages 606–611. IEEE Computer Society Press, 1999.

[WH99b] Y. Wu and T. S. Huang. Human hand modelling, analysis and animation in the context of HCI. In *Proc 6th Int Conf on Image Processing, Kobe, Japan, October*. IEEE Computer Society Press, 1999.

[WH99c] Y. Wu and T.S. Huang. Vision-based gesture recognition: a review. *Lecture Notes in Computer Science*, 1739:103+, 1999.

[WH01] Y. Wu and T.S. Huang. Hand modeling, analysis, and recognition for vision-based human computer interaction. *IEEE Signal Processing Magazine*, 18(3), 2001.

[WHY03] Y. Wu, G. Hua, and T. Yu. Tracking articulated body by dynamic Markov network. In *Proc 9th Int Conf on Computer Vision, Nice, France, Oct 13-16*, pages 1094–1101. IEEE Computer Society Press, 2003.

[WL88] C. W. Wampler and L. J. Leifer. Applications of damped least- squares methods to resolved-rate and resolved-acceleration control of manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 110:31–38, 1988.

[WLH00] Y. Wu, Q. Liu, and T.S. Huang. An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization. In *Proc Asian Conf on Computer Vision*, Taiwan, 2000.

[WLH01] Y. Wu, J. Y. Lin, and T. S. Huang. Capturing natural hand articulation. In *Proc Int Conf in Computer Vision (ICCV)*, Vancouver, Canada, 2001. IEEE.

[WN99] S. Wachter and H. H. Nagel. Tracking persons in monocular image sequences. *Computer Vision and Image Understanding*, 74(3):174–192, 1999.

[WS00] G. Wyszecki and W. S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley Classics Library. Wiley Inter-Science, 2nd edition, August 2000. First edition published in 1967.

[YD98] Y. Yacoob and L. S. Davis. Learned temporal models of image motion. In *Proc 6th Int Conf on Computer Vision, Bombay, India, January 4-7*, pages 446–453. IEEE Computer Society Press, 1998.

[YI98] M. Yachida and Y. Iwai. Looking at human gestures. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*, pages 291–311. Cambridge University Press, 1998.

[YLW98a] J. Yang, W. Lu, and A. Waibel. Skin-color modeling and adaptation. In *Proc Asian Conf on Computer Vision*, volume II, pages 687–694, Hong Kong, 1998.

[YLW98b] J. Yang, W. Lu, and A. Waibel. Skin-color modeling and adaptation. In *Proc Asian Conf on Computer Vision*, volume II, pages 687–694, 1998.

[YPL06] H. D. Yang, A. Y. Park, and S. W. Lee. Robust spotting of key gestures from whole body motion sequence. In *Proc of the IEEE Conf on Automatic Face and Gesture Recognition, Southampton, UK*, 2006.

[YW96] J. Yang and A. Waibel. A real-time face tracker. In *Third IEEE Workshop on Applications of Computer Vision*, pages 142–147, 1996.

[ZH03] H. Zhou and T. Huang. Tracking articulated hand motion with eigen dynamics analysis. In *Proc 9th Int Conf on Computer Vision, Nice, France, Oct 13-16*. IEEE Computer Society Press, 2003.

[ZYW00] X. Zhu, J. Yang, and A. Waibel. Segmenting hands of arbitrary color. In *Proc Fourth IEEE Int Conf on Automatic Face and Gesture Recognition*, Grenoble, France, March 2000.