Abstract

Integrating Multiple Visual Cues for Robust Tracking

Christopher Eric Rasmussen

Yale University

2000

Vision-based tracking is a promising technology for tasks such as human-computer interaction and mobile robot navigation. However, distraction and occlusion are major obstacles to robust performance. Though tracking is often framed as strictly an estimation problem, these phenomena also engender a correspondence problem that must be addressed. Without careful consideration of what image data, if any, to associate with a tracked object from frame to frame, the estimation process can become biased and the object lost.

In this dissertation, we will describe a framework that incorporates and indeed emphasizes explicit reasoning about data association as a means of improving tracking performance in many difficult visual environments. This framework is built around a hierarchy of three tracking strategies that result from ascribing ambiguous or missing data to the following causes: (1) noise-like visual occurrences; (2) persistent, known scene elements (i.e. other tracked objects); or (3) persistent, unknown scene elements.

First, we introduce a randomized tracking algorithm adapted from an existing probabilistic data association filter (PDAF) that is resistant to clutter and follows agile motion. The algorithm is applied to three different tracking modalities—homogeneous regions, textured regions, and snakes—and extensibly defined so that

the inclusion of other methods is straightforward.

Second, we add the capacity to track multiple interacting objects by adapting a joint version of the PDAF to vision. This algorithm oversees correspondence choices between same-modality trackers and image features to ensure that they are feasibly distributed. We then derive a related technique that allows mixed tracker modalities, handles object overlaps, and deduces depth orderings.

Finally, we represent complex objects as conjunctions of cues that are diverse both geometrically (e.g., a person's face, hands, and torso) and qualitatively (e.g., shape, color, and texture). The use of rigid and hinge constraints between part trackers and multiple attributes to describe individual parts renders the whole object more distinctive, reducing susceptibility to mistracking.

Models for tracked targets can be flexibly specified; results are given for a number of objects, including people, cars, airplanes, microscopic cells, and chess pieces.

Integrating Multiple Visual Cues for Robust Tracking

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Christopher Eric Rasmussen

Dissertation Director: Gregory D. Hager

December, 2000

Acknowledgments

My first and foremost thanks go to Greg Hager, who took me in when my original advisor left Yale and I was at a loss for how to proceed. From the moment he fired up that first SSD tracker and I watched a little green box follow my face across the screen as I moved left and right, up and down, I knew that vision was something I wanted to be a part of. Besides being an assiduous teacher of the mathematical foundations of vision and tracking, Greg has also always pushed me to think and write more clearly. If I have managed to reduce my natural prolixity at all over the course of graduate school, the credit is due to him. Working in Greg's lab has led to many interesting experiences and rewarding friendships, and I owe him a great deal for affording me the privilege.

I must also thank the many faculty members whom I have been fortunate enough to know and interact with. In particular, Peter Belhumeur, David Kriegman, Drew McDermott, Zhong Shao, Hemant Tagare, and Jeffrey Westbrook have helped nudge my graduate school career forward at a few crucial steps along the way, been inspiring teachers, and generally made the department and Yale a stimulating and friendly place to be.

Fellow toilers who have brightened slow days (and nights) with conversations both edifying and diverting also deserve recognition. Many thanks to Zachary Dodds, Kentaro Toyama, Shamez Alibhai, Ohad Ben-Shahar, Athos Georghiades, Rich LeGrand, Bret Logan, Tianyun Ma, and Hong Xiao for their camaraderie and many invaluable suggestions.

Technically, I should thank Morrow Long, Rob Hubley, Roger Smith, Ken Hoover, and other members of the facilities group for so readily and frequently dropping their real work when I showed up, desperate to get a balky computer running, an OS installed, my e-mail connection restored, or some other can't-wait problem solved.

Thanks also to Annette Durso, Cathy Esposito, Lori Hammell, Chris Hatchell, Linda Joyce, Judy Smith, and other staff of the Computer Science department for their help over the years with innumerable financial, bureaucratic, and mimeographic crises. So long, and thanks for all the pizza.

Without the funding support of the National Science Foundation, the Department of Defense, and other miscellaneous grant sources, none of this research would have been possible.

I'm also indebted to the authors of the following software and pieces of code for making my life considerably easier: `xv` for image manipulation and conversion (John Bradley), `avi2mpg1` for AVI to MPEG conversion (John Schlichther), the Berkeley MPEG library for MPEG encoding and decoding (Greg Ward *et al.*), the JPEG library for JPEG reading and writing (Thomas Lane and the Independent JPEG Group), `mkavi` for AVI encoding (Josh Osborne), Canny edge detection code (Mike Heath), and OpenGL to Postscript conversion code (Mark Kilgard).

John McCormick and Gaudenz Danuser were kind enough to supply some video sequences used in this thesis, so thank you to them.

Thanks very much to Rocco Servedio for offering valuable proofreading and stylistic advice on short notice, and for helping me keep a sense of humor about the whole endeavor.

On a personal note, thanks to Mom and Dad for writing their own dissertations, thereby setting a bad example for an impressionable youngster.

Finally, without Yvonne's support and patience over the course of this often interminable-seeming process, I don't think I could have finished. Thank you for everything.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

If you dare not trust that you see, confess not that you know:

If you will follow me, I will show you enough ...

William Shakespeare

Much Ado About Nothing

Act 3, Scene 2

To follow an object with the eye—to *track* it—is virtually a subconscious activity for a person most of the time. Only under very difficult circumstances is one aware of it taking any effort at all: while watching the ball carom around energetically during a pinball game, perhaps, or trying to pick out a friend's face as they make their way through a crowd. Without attempting to enumerate every reason that tracking is harder in these situations, a few factors stand out. The speed of movement of the ball, its frequently unpredictable changes of direction, the proximity of similar faces to the friend's face, and the constant disappearances and reappearances of their body behind others—all seem to conspire to interrupt what is normally a smooth tracking process.

The history of artificial intelligence amply shows, of course, that even skills which come naturally to people are rarely straightforward to impart to computers. Nonetheless, there is a strong motivation to study visual tracking and its attendant issues, if only because of the many useful tasks that computers and robots could carry out with just a fraction of the human visual system's capabilities. For instance, many researchers have been interested in perception for autonomous vehicles, such as driverless cars that follow lane lines and detect other cars as they negotiate highways [35, 116]. Success in this area would have obvious ramifications for public transportation and shipping. Other have studied mobile robots that visually avoid obstacles and locate landmarks to navigate within buildings, allowing them to make deliveries or give tours [25, 33, 55]. Another focus of investigation has been on gathering information from passive, ground- and air-based cameras for surveillance and activity classification tasks such as security, military reconnaissance, and studying pedestrian and vehicle traffic patterns [79, 113]. The human body has also been the subject of much work on analyzing gestures, facial expressions, and other motions in order to drive character animation, understand sign language, serve as input to games and other software, and recognize actions [32, 40, 44, 104, 112, 122]. Finally, a number of ongoing projects seek to combine disparate tracking skills in an effort to endow "intelligent" houses with an awareness of their occupants and an ability to interact with them [73, 107]. These examples represent just a small sample of the diverse potential applications of visual tracking research.

Traditionally, the emphasis in framing the visual tracking problem has been on *estimation* [86, 94]. Given a set of data that we wish to represent concisely with a parametric model, an *estimator* is a procedure for finding the parameters of the model which in some sense best fits the data. In tracking, the parameters (known collectively as the *state*) are typically time-varying, salient characteristics of an object

Figure 1.1: Multiple distractions complicate tracking

in the field of view. Common choices include location in the image, depth along the camera axis or image scale, orientation in the image plane or three dimensions (3-D), velocity and/or acceleration of any of these quantities, shape (possibly allowing for nonrigid deformation), surface properties like color, and so on. The data available to guide estimation are, theoretically, all of the images observed up to the present time. However, unless the projection of the object occupies the entire image area, the object model is generally only sufficient to explain a fraction of the data. Therefore, a standard procedure is to segment out the portion of the image data which corresponds to the object and use it alone for estimation. This assumes that the object's image projection can be unambiguously discriminated from the rest of the image.

This basic approach has led to much fruitful research on tracking topics such as inferring 3-D structure from 2-D images [106] and dynamics [18]—how to keep up with a moving target by predicting where it will be next. These kinds of problems are challenging in their own right. But as visual tracking moves out of the laboratory and into the real world, controlled conditions disappear and it becomes considerably more difficult to accurately identify an object's image projection. Unsurprisingly, the same visual phenomena noted above which are problematic for people also interfere with estimation: agile motion, distractions, and occlusions. We define *agile motion* as a sustained object movement or acceleration that exceeds a tracker's dynamic prediction abilities. Its occurrence can undermine the estimation process because it renders the putative location of the object's image projection uncertain, complicating efficient segmentation. A further obstacle to clear-cut segmentation is a *distraction*, or another scene element which has a similar image appearance to the object being tracked. For example, the numerous penguins in the rookery shown in Figure 1.1 hinder any effort to track just one bird. A naive penguin segmenter may return multiple penguin-containing image regions or it may isolate a unique-but-wrong one. Either outcome will cause the estimator to work from an incorrect data set. Finally, *occlusion* results when another scene element is interposed between the camera and the tracked object, blocking a portion of the object's image projection. This results in incomplete data or no data being supplied to the estimation algorithm.

These phenomena are worrisome because of their potential to bias a tracker's estimates by polluting or decimating the data. Moreover, if a visual disturbance lasts too long or is too severe, the estimator/tracker can effectively become confused and lose track altogether of the object. By *mistracking*, we mean that some criterion for the quality of the state estimate over the duration of the tracking task is not satisfied. Such a criterion might be that the difference between the state estimate and the ground truth state (insofar as it is knowable or humanly assessable) is never greater than a certain threshold, or at least for no longer than a preset length of time. Since visual disturbances are ubiquitous in everyday situations, methods to overcome them are critically important if visual tracking is to be robust.

The aim of this dissertation is to analyze some of the essential causes of disturbances in tracking and outline a comprehensive computational framework for dealing with them. At the most fundamental level, it seems clear from our exposition that these problems require an approach that combines both estimation and *correspondence*. Correspondence is the question of how to determine what image data to properly associate with the object being tracked and therefore to base the estimation process on. We contend that by combating the various forms of the correspondence problem that they engender, we will be able to counteract many of the negative effects of agile motions, distractions, and occlusions on accurate estimation.

We will tackle these problems with two broad approaches. First, we will describe several *data association* [5] versions of the Kalman filter [69], an estimation technique commonly used for visual tracking, that are specially constructed to handle certain classes of these occurrences. Two of the data association filters that we present are existing algorithms from the radar and sonar literature that we have made nontrivial modifications to in order to adapt to vision. We will also introduce two other, more sophisticated filters, that are novel. The standard Kalman filter requires tinkering because in order to assure an optimal estimate it assumes a Gaussian distribution on target observations. Thus, it may fail when this assumption is violated, as is the case when a distraction or occlusion induces a multimodal distribution on target-like observations.

The second part of our strategy is a method of defining a tracked object more *distinctively* so that visual disruptions happen less frequently and with less severity. Distractions are defined as image features that are similar to the target, so a more specific target description (e.g., "red triangle" instead of just "red") tends to reduce the number of such features. A *de facto* description of an object is given by the set of *modalities*—appearance, color, shape, etc.—used to track it, leading us to employ conjunctions of modalities for greater robustness. Geometric relationships such as "triangle-below-circle" instead of just "triangle" can also be added to an object description to boost distinctiveness. The overarching aim of these measures is to reduce or eliminate the incidence of non-target-originated observations and thus the degree to which the assumption of a Gaussian distribution is not met.

Exceptions to the Kalman filter's Gaussian assumption do not necessarily cause mistracking. They simply remove the guarantee of optimality, which in any case is rarely achievable for real-world images. Thus, the algorithms we will present do not seek to completely eliminate problems, only to mitigate them. Suboptimal performance can, of course, result in mistracking. Our claim is that these techniques will help many trackers perform at a satisfactory level in a wider range of visual situations.

This work is primarily organized around an analysis of three interpretations of distractions and occlusions in the context of estimation, and the description of a series of filters tailored to deal with them. First, we treat visual disturbances as random, transitory events. This prompts the simplest of the four data association filters that we will present: the Probabilistic Data Association Filter (PDAF) [5]. Second, we examine the case where disruptions are caused by other tracked objects. The Joint Probabilistic Data Association Filter (JPDAF) [5] and Joint Likelihood Filter (JLF) are responses to this interpretation. Finally, we allow that disruptions may be persistent and of unknown origin. Our strategy in this difficult situation is to define the object more distinctively: the Constrained Joint Likelihood Filter (CJLF) is a method for conjoining trackers of different modalities that are geometrically linked. Each of these filters (in the order just listed) incorporates the core of the one preceding it while adding more functionality. Thus, for example, the PDAF can withstand noisy visual phenomena that may be distracting, but the JPDAF compensates for these *and* distractions caused by other targets.

As the PDAF and JPDAF are based on the Kalman filter, they work with point-

Figure 1.2: Noise-like distractions due to falling snow

like *measurements* rather than directly on images. The other major component of this dissertation is therefore a process for segmenting a discrete set of image areas that resemble the target (where resemblance is a metric that depends on the modality used for tracking) and supplying summaries of these as measurements to the data association filters, including the JLF and CJLF. The term "measurement" thus serves as a convenient shorthand for coherent subsets of the image data that may be used for state estimation. It is the data association filters that address the correspondence portion of the visual tracking problem defined above by selecting or weighting the influence of these alternatives on estimation.

We will now examine the three interpretations of visual disruptions and the data association filters that they motivate in somewhat more detail.

Noise An often-plausible interpretation of multiple or missing target-like observations results from ascribing them to random visual events. The image is typically corrupted by noise generated by the camera CCD or video capture device, but such effects are usually only significant for targets projecting to very small image areas.

Another kind of transient event that can hamper object tracking is due to scene processes such as dazzling sun reflections on a choppy body of water, or bad weather like the falling snow in Figure 1.2, which results in a speckled foreground. These phenomena can distract estimation by causing false target-like observations if they and the object have similar colorations, or if the segmentation procedure is sensitive to strong motion or intensity cues. Alternatively, the noisy image patches may periodically obscure the target, suppressing the expected target-originated observation. If the correspondence mechanism is too narrow in the data it allows to be used for estimation (picking only one of several possible observations, for example), it may exclude the correct data and eventually be pulled away from the target. By considering more data, our adaptation of the PDAF algorithm to vision lessens the chance of a misstep and thus improves robustness. Assuming that their distribution is roughly uniform or Poisson and that their density is not too great, the effects of the transient, non-target-originated observations cancel each other out and the more persistent target-originated observation dominates the estimation process.

The segmentation procedure that we introduce for finding measurements uses a nonlocal search in order to consistently identify multiple possible sources of ambiguity. A byproduct of this nonlocality is improved performance when the target makes agile movements. The sudden jump of a target away from its expected state can confound trackers that are not prepared for clutter, but by casting a wide net for distractors our algorithm catches up with such motions and resumes normal tracking.

Other objects When trying to track multiple similar or identical objects, by definition there is more than one image feature that matches the object description well, such as with the penguins in Figure 1.1. This means that the segmentation procedure of a single tracker running in isolation will consistently return multiple measurements when the objects are close to one another. Because these measurements are persistent, they invalidate the assumption of the PDAF that only one measurement is due to the target and that the rest stem from noise. The inclusive use of data by the PDAF backfires in this case, yielding a state estimate that is a compromise between the estimates that would be obtained if a separate estimator were correctly associated with each persistent measurement. The JPDAF effectively implements this approach by extending the PDAF to share information between trackers. The maintenance of a logic of associations prevents different trackers from claiming the same measurement and thus substantially improves the quality of the data they use for state estimation. To ensure that the trackers are all considering the same pool of measurements, a joint segmentation procedure is necessary to merge the results of the individual segmenters (which only look at the image in the neighborhood of where they expect their object to project) and eliminate duplicates.

The JPDAF only works for tracking a set of identical objects and does not consider the possibility of targets occluding one another, a fairly frequent occurrence. An occlusion modifies the observed image projection of an object, causing a mismatch with the expectations of the standard segmentation procedure. The JLF expands the sharing of information between trackers in order to deduce occlusions during segmentation, resulting in better measurements. Moreover, the JLF evenly assorts trackers among image features as the JPDAF does, but it improves on the JPDAF by allowing a mixture of different trackers to interact—e.g., a color-based tracker and a shape-based tracker.

Persistent, unknown features Distractions and occlusions may also result from untracked, non-noise scene elements. This class of phenomena is difficult to counteract within the context of a data association filter. Rather, we will pursue a strategy

Figure 1.3: Conjunctions of parts and attributes describe objects more distinctively

of trying to reduce the incidence of persistent disturbances by defining the target more distinctively. Consider the soccer players in Figure 1.3. A color-based tracker assigned to the shirt of the player on the right may have trouble with distractions because there are other red regions in the image. However, by linking the shirt tracker to a white-shorts tracker via a *constraint*, we eliminate red regions that are not above white regions as potential distractors. A still-more distinctive description of the shirt would result from taking into account the appearance of the logo on its front. Then we would be looking for red regions with the proper insignia above white regions, a unique feature in this image.

Adding geometric parts to an object description, of course, makes the effective target bigger. A benefit of increased size is a corresponding decrease in the negative influence of typical occlusions. If we were tracking the red player's shirt alone, for example, the arm of the yellow player in front on him would bias any estimate of its size. When tracked in conjunction with the logo and the shorts, which are unoccluded, the shirt tracker has additional scale information to help improve its estimates.

The CJLF is an extension of the JLF that incorporates constraint information to more distinctively describe and thus more effectively track an object. It provides a framework for the combination of multiple attributes, such as the color of the shirt and appearance of the logo, and multiple geometric parts, such as the shirt and shorts.

1.1 Theses

This dissertation is concerned with mitigating the effects of disruptions in visual tracking caused by distractions, occlusions, and agile motions. We observe that these phenomena cause ambiguity about which image features to base estimation on. Therefore, we contend that a correspondence problem must be addressed in concert with estimation in order to ensure robust tracking. Our approach is to classify several possible sources of these disturbances—namely, noise, other tracked objects, or persistent, unknown scene elements—and construct or adapt data association filters apposite for each one. A guiding principle in the design of these filters is the integration of multiple sources of information. Multiple trackers share information about correspondences to avoid interference and single-object trackers are defined as conjunctions of multiple attributes and parts for greater distinctiveness.

Our approach to distinctiveness leads to a building-block strategy of composing complex trackers from relatively simple pieces (in terms of geometry and modality) instead of creating a monolithic algorithm specialized for a particular task. This makes our framework flexible with regard to what is being tracked and amenable to the incorporation of techniques used by other researchers or devised in the future.

Briefly, this dissertation makes the following contributions to vision-based tracking:

- We adapt and apply data association methods (PDAF, JPDAF) to vision for significantly better tracking performance in the presence of noise and multiple targets
- We define a robust, general algorithm for finding a set of good matches to the target's image projection
- We put different tracking modalities (color regions, SSD features, and snakes) into a common framework for comparison and interoperation; our approach to color region tracking is original
- We introduce a joint tracker (JLF) that supersedes the JPDAF by accommodating trackers with different modalities, handling partial occlusions, and deducing depth relationships from the image
- We extend the JLF to incorporate geometric constraints between low-level trackers, including a novel notion of "layered" attributes of the same image areas

1.2 Overview

Chapter 2 reviews the probabilistic foundations of the visual tracking problem, explains the assumptions of our Kalman filtering approach, and defines some useful terms.

Chapter 3 derives a formulation of image similarity for three modalities that rely on color, shape, or appearance to define the target. Defined more generally, we will call these modalities *homogeneous regions*, *snakes*, and *textured regions*, respectively. These ways of interpreting images will constitute the basis of the segmentation method outlined in the following chapter.

Chapter 4 addresses noise and agile motion as causes of mistracking. First, the image preprocessing necessary for successful adaptation of data association filters to vision are analyzed. Specifically, we examine gradient ascent methods for obtaining a single best hypothesis with which to update the state estimate, and then detail a procedure that extracts a set of good hypotheses using random sampling. It reviews a measurement-based tracking algorithm from the radar and sonar literature, the Probabilistic Data Association Filter (PDAF), that performs well in the presence of noise. The modalities from Chapter 3 are used to track various objects that exhibit agile movement and in the presence of background clutter.

Chapter 5 examines the problem of interference caused by other known objects. It explains two techniques for tracking that extend the work of the previous chapter to manage a known number of multiple similar or interacting objects. The first is an existing extension to the PDAF for joint tracking, the JPDAF, which is explained and adapted to vision with a refinement to the measurement generation process that combines gradient ascent with random sampling. The second technique covered, the Joint Likelihood Filter (JLF), is a new algorithm that handles targets of different modalities that overlap one another by reasoning about their depth ordering.

Chapter 6 introduces methods for describing a tracked object more distinctively in order to minimize the deleterious effects of unknown, persistent distractions in the scene. An extension to the JLF, the Constrained Joint Likelihood Filter, is explained that defines complex objects via geometric constraints between trackers of like and different modalities.

Chapter 7 surveys related work on tracking by other researchers and analyzes its relationship to the techniques promulgated here.

Finally, we sum up our contributions in Chapter 8 and discuss future research directions.

Chapter 2

Background

Mumford [89] and others have suggested that many problems in vision may be cast as an attempt to find a *maximum a posteriori* (MAP) estimate [86] of the state of the world given a signal that is a transformed version of it. The image $\mathbf{I}$ perceived by a camera or eye at any given moment is the result of a transformation of the world $\mathbf{W}$ dictated by the laws of physics. Bayes' theorem [89, 101] provides a tool for reasoning probabilistically about the world from what is seen:

$$p(\mathbf{W} \,|\, \mathbf{I}) = \frac{p(\mathbf{I} \,|\, \mathbf{W}) p(\mathbf{W})}{p(\mathbf{I})} \tag{2.1}$$

$p(\mathbf{I})$ can be deduced from the other terms by $p(\mathbf{I}) = \int p(\mathbf{I} \mid \mathbf{W}) p(\mathbf{W}) d\mathbf{W}$ (where $\mathbf{W} \in \mathcal{W}$, the space of all possible worlds), so it is typically treated as a normalizing constant $1/k$. A MAP estimate of the state of the world, which is not necessarily unique, is a maximally likely one given the observed image: $\text{argmax}_{\mathbf{W}} p(\mathbf{W} \,|\, \mathbf{I})$. The maximum likelihood (ML) estimate [38, 101] is equivalent to the MAP estimate assuming a uniform (and therefore noninformative) prior probability on the state of the world $p(\mathbf{W})$.

To track, an observer focuses its interest on a small part of the world, which we call an *object* or *target*, and takes past images into account. At time t the state $\mathbf{X}_t \in \mathcal{X}$ represents the current estimate of the object's salient parameters (where $\mathcal{X}$ is a subspace of $\mathcal{W}$). Instead of a single image, a tracking estimator uses the sequence of images $\mathbf{I}_t, \mathbf{I}_{t-1}, \ldots$ observed so far. Thus, the tracking task in the MAP framework is to estimate a state that maximizes $p(\mathbf{X}_t \mid \mathbf{I}_t, \mathbf{I}_{t-1}, \ldots)$. Applying Bayes' theorem and rearranging yields the following expression [5, 60]:

$$p(\mathbf{X}_t \,|\, \mathbf{I}_t, \mathbf{I}_{t-1}, \ldots) = k_t p(\mathbf{I}_t \,|\, \mathbf{X}_t) p(\mathbf{X}_t \,|\, \mathbf{I}_{t-1}, \mathbf{I}_{t-2}, \ldots) \tag{2.2}$$

Here $p(\mathbf{X}_t \mid \mathbf{I}_{t-1}, \mathbf{I}_{t-2}, \ldots)$, which summarizes prior knowledge about $\mathbf{X}_t$, is a prediction based on the previous state estimate and knowledge of the object's dynamics. In order to simplify the last term on the right hand side, we must assume that object dynamics are such that states form a Markov chain [101], so $p(\mathbf{X}_t \,|\, \mathbf{X}_{t-1}, \mathbf{X}_{t-2}, \ldots) = p(\mathbf{X}_t \,|\, \mathbf{X}_{t-1})$. This yields $p(\mathbf{X}_t \,|\, \mathbf{I}_{t-1}, \mathbf{I}_{t-2}, \ldots) = \int_{\mathbf{X}_{t-1}} p(\mathbf{X}_t \,|\, \mathbf{X}_{t-1}) p(\mathbf{X}_{t-1} \,|\, \mathbf{I}_{t-1}, \mathbf{I}_{t-2}, \ldots)$.

The quantity $p(\mathbf{I}_t \,|\, \mathbf{X}_t)$ describes the relative probability of observing a particular image at time t given the current state. We call this the *image likelihood*. The image likelihood depends on the physics of image formation and noise that may corrupt what is expected [75]. Assuming that the laws of optics are fixed and that noise sources are roughly stationary, we can drop the time indices and refer to the image likelihood as $p(\mathbf{I} \,|\, \mathbf{X})$.

In order to define $p(\mathbf{I} \,|\, \mathbf{X})$ more precisely, we will first introduce a few terms. Let the space of images be $\mathcal{I}$ and $\pi : \mathcal{X} \rightarrow \mathcal{I}$ be an *image prediction* function which describes the expected image projection of the target assuming that it is in a particular state. Depending on how detailed it is, $\mathbf{X}$ alone may be insufficient to predict an image, making it necessary to build information about $\mathbf{W}$ into π (besides physical laws themselves). For example, if they are not explicitly included in $\mathbf{X}$, assumptions must be made about lighting, occlusions, background, reflectance properties of the tracked object, camera variables such as focal length, and so on. In this sense, modeling the image formation process is closely related to computer graphics [43].

The image actually observed also depends on noise. Noise is a factor that increases uncertainty about the exact image projection of the tracked object. With no uncertainty, the image likelihood $p(\mathbf{I}|\mathbf{X})$ would be unity at the exactly the expected image and zero everywhere else. With it, other images tend to have a likelihood proportional to their degree of similarity to the expected image.

An efficient algorithm for computing the MAP estimate of Equation 2.2 when $p(\mathbf{X}_t \mid \mathbf{I}_t, \mathbf{I}_{t-1}, \ldots)$ is Gaussian is the Kalman filter [5, 69]. In order for $p(\mathbf{X}_t \mid \mathbf{I}_t, \mathbf{I}_{t-1}, \ldots)$ to be Gaussian, $p(\mathbf{I} \mid \mathbf{X})$, $p(\mathbf{X}_t \mid \mathbf{X}_{t-1})$, and the prior probability of the state before any images are viewed must be Gaussian. Some possible causes of and remedies for non-normality are discussed in [60]; in Chapters 4 and 5 we present data association filters that handle certain kinds of violations of the Kalman filter's assumptions.

Tracking filters such as the Probabilistic Data Association Filter (PDAF) and Joint Probabilistic Association Filter (JPDAF) [5] were originally developed for tracking aircraft radar blips, a domain that differs from vision-based tracking in a number of respects. Most importantly, many radar-type targets are simply points, limiting their state to position and dynamics. We need to include such parameters as scale, orientation, color, and shape. Furthermore, the unmodified data association filters assume that a measurement or measurements are implicitly provided to them. The meaning of a "measurement" in the context of raw images is not immediately obvious. As we noted in the previous chapter, we define measurements to be image areas that match the target's expected image projection well. A discrete list of the modes of the image likelihood function reduces the data any state estimator must process, while capturing the intuitive notion of alternative candidate states that a tracker might be in.

For radar blips, a target might be expected to be observed as a bright point on a dark background, so simple thresholding would quickly segment out all high-likelihood hypotheses for the target location. These hypotheses could be succinctly represented as a list of (x, y) pairs. Generating visual target measurements is usually more difficult than thresholding, and their extents require more information to be adequately summarized than simple image location. Possible measurement parameters include geometric characteristics relevant to the target state such as the location of the area's center and its height, width, and orientation. These parameters define a *measurement space* $\mathcal{Z}$ such that a point $\mathbf{Z} \in \mathcal{Z}$ is related to a state $\mathbf{X}$ via a continuous *measurement function* $H(\mathbf{X}) = \mathbf{Z}$. The measurement function may simply reduce the dimensionality of $\mathbf{X}$ by dropping its temporal parameters, or it may also describe a more complicated relationship between what is measured and what is estimated.

The bases for $p(\mathbf{I} \mid \mathbf{X})$, and therefore for the measurement generation procedure that is described in Chapters 4 and 5, are the form of the predicted image projection of a target $\pi(\mathbf{X})$ and the method for quantifying the similarity of the image $\mathbf{I}$ to that prediction. Both of these depend on what we call the *modality* used to identify the object. A tracking modality is a visual attribute such as a specific shape, color, direction of motion, etc. that constitutes a tracking algorithm's complete description of its target. For example, suppose we want to track a bright red ball. We might choose a color modality to predict the hue of the ball's circular image projection and to define a metric on circular areas of hue in order to gauge the similarity of our prediction to the actual image. This method does not exploit all available image information about the ball (ignoring, for instance, any designs printed on it or its motion), but makes a choice about what information is relevant and adequate. The act of selecting a modality is first an acknowledgement that there is as yet no all-purpose, exact theory of object appearance, and second an assertion that for efficiency's sake only partial information, if picked judiciously enough, can yield satisfactory tracking performance.

The question of whether the information about $\mathbf{X}$ transmitted by a modality is the same as that carried by $\mathbf{I}$ is captured by the notion of a *sufficient statistic* [27]. Most modalities are not sufficient statistics, so we should be concerned about their effect on the accuracy of estimating $\mathbf{X}$. It is certainly possible, based on the image environment and target, to use a tracking modality that removes too much state information to be useful. Of course, the image itself may not contain enough information about the object for it to be tracked. The methods described in this dissertation place the ultimate responsibility for determining the best modality for a given tracking task, or whether a task can indeed be accomplished, in the user's hands. In Chapter 8, we sketch a possible approach to automating these decisions.

The next chapter describes in detail three different, complementary modalities that we use for tracking.

Chapter 3

Tracking Modalities

In this chapter we discuss the first component of our approach to tracking solitary objects assuming they are the only salient visual features in the image. This is the form of the likelihood function $p(\mathbf{I} \mid \mathbf{X})$ for the various modalities used to analyze the image. The image likelihood is explicated for three specific types of tracker: homogeneous regions, textured regions, and snakes.

3.1 Regions

The term "region" has a precise mathematical meaning (i.e., an open, connected set [50]) that differs slightly but significantly from the sense in which this dissertation uses it. For visual tracking, we define a *region* as the projection onto the image of a simply connected mathematical region lying on a smooth surface. We will subsequently refer to the mathematical region that projects to an image region as a *patch* in order to avoid confusion and emphasize its relationship to a physical surface.

The image prediction function π defined in Chapter 2 has a few essential components for a given region R. Whether derived from the region's state $\mathbf{X}$ or taken as constants, certain values are necessary to simulate the image formation process and

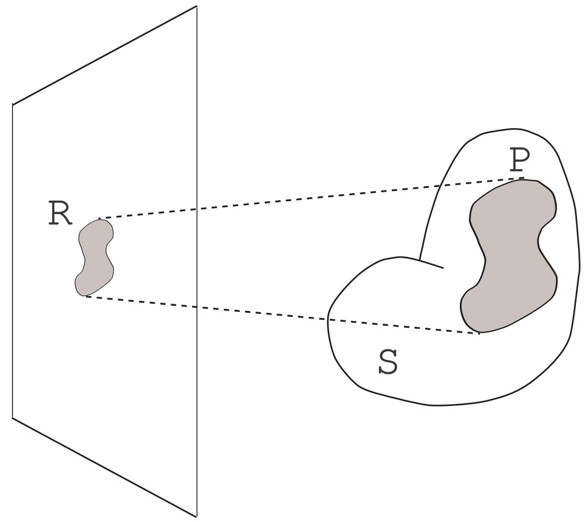


Figure 3.1: Viewing geometry for the projection of a surface patch P onto an image region R.

arrive at a predicted image $\pi(\mathbf{X})$. One set of required variables describes the geometry of the patch P that gives rise to R, including shape, location in $\mathbb{R}^3$ relative to the viewer, and orientation. A second group of variables summarizes the reflectance properties of P, such as its pattern of albedos for various spectra (e.g., coloration) and bidirectional reflectance distribution function (BRDF) [54]. Finally, information about scene illumination, camera parameters, occlusions or inter-reflections involving the surface S associated with P, and other incidentals must also be encoded.

This section examines two specific kinds of regions distinguished by the reflectance properties of P. Let $c_P : P \rightarrow \mathbb{R}^3$ be a function describing the intrinsic color pattern over P in RGB space (akin to a computer graphics texture map [43]). If $c_P(P)$ is constant, then we call R a *homogeneous* region. If $c_P(P)$ has significant intensity gradients both vertically and horizontally (P may be curved, but its simple connectedness allows it to be mapped to $\mathbb{R}^2$), then we call R a *textured* region. A Lambertian [54] BRDF is assumed for both homogeneous and textured regions. The differences in $c_P(P)$ between the two region types lead to alternative formulations of the image prediction function π and similarity between the actual and predicted images.

3.1.1 Homogeneous Regions

Let $c_R : R \to \mathbb{R}^3$ be a function describing the irradiance (image brightness) of the homogeneous region R. According to the Dichromatic Reflection Model (DRM) [74], if P is Lambertian the range of c_R lies on a line (the *matte* line) in RGB space passing through $(0, 0, 0)$. The direction of the matte line describes the intrinsic color of P, while the distribution of $c_R(R)$ along it is governed by the curvature and orientation of P, as well as the amount of ambient illumination. In practice, camera noise and departures in c_P from perfect uniformity makes the matte line a cluster, and factors such as gamma correction and the limited dynamic range of the camera can introduce nonlinearities in the cluster shape.

Furthermore, if P is glossy then specularities give rise to a second line (the *highlight* line) in RGB space that intersects the matte line in a skewed-T configuration. The direction of the highlight line is determined by the color of the light source causing the specularity. The matte line and the highlight line completely describe c_R under the DRM.

For many interesting materials such as human skin, ordinary clothing fabric, and automobile paint, the DRM is a useful, accurate reflectance model. In previous work [95], we described a method for modeling a given region R's color by parametrizing c_R's distribution along the matte line alone. Before starting tracking, the user manually selects a set of pixels in R that are non-highlighted, non-saturated, and have significant intensity variation to get a well-defined, linear distribution. Using principal components analysis (PCA) or singular value decomposition (SVD) [91, 94], an ellipsoid whose major axis is aligned with the matte line is fit to the sampled pixels' color distribution in RGB space.

This is accomplished as follows. Suppose $\mathbf{D}$ is a $3 \times K$ matrix of the K RGB

values chosen by the user. The 3×3 covariance matrix for $\mathbf{D}$ is defined by $\mathbf{\Sigma}_{ij} = \sum_{k=1}^{K} (\mathbf{D}_{ik} - \mu_i)(\mathbf{D}_{jk} - \mu_j)$, where μ_1 is the mean red component r of the pixels selected, μ_2 is the mean green component g, and μ_3 is the mean blue component b. Singular value decomposition factors any $M \times N$ matrix $\mathbf{A}$ for which $M \geq N$ into $\mathbf{A} = \mathbf{U}\,\mathbf{W}\,\mathbf{V}^T$, where $\mathbf{U}$ is an $M \times N$ column-orthogonal matrix, $\mathbf{W}$ is an $N \times N$ diagonal matrix, and $\mathbf{V}$ is $N \times N$ and orthogonal. For symmetric $\mathbf{A}$ such as covariance matrices, $\mathbf{U} = \mathbf{V}$.

The SVD of $\mathbf{\Sigma}$ has a geometric interpretation that we clarify by rewriting it as $\mathbf{\Sigma} = \mathbf{R}\,\mathbf{S}^2\,\mathbf{R}^T$, where $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{S}$ is a 3×3 scaling matrix that define an ellipsoid in $\mathbb{R}^3$. This ellipsoid, which is obtained by scaling the unit sphere by $\mathbf{S}$, rotating it by $\mathbf{R}^T$, and translating it by $\mathbf{T} = (\mu_1, \mu_2, \mu_3)^T$, fits R's matte cluster and hence models its color. The steps of the process leading from sample selection to model definition are illustrated in Figure 3.2(a-d) for a region corresponding to the front of a soccer player's orange jersey.

At the level of a pixel (x, y) within the region R described by the current state, the image prediction function π postulates that its color will simply be $\mathbf{T}$. We use the Mahalanobis distance [38], which is the number of standard deviations σ from a point to the center of a multivariate Gaussian distribution, to measure the similarity γ between the predicted pixel color $\mathbf{T}$ and the actual color $\mathbf{I}(x, y)$ at that image location. The function γ plus a representation of the region's geometry will constitute an overall image similarity metric, which is discussed below. The Mahalanobis formulation results in $\gamma(\mathbf{I}(x, y), \mathbf{T}) = [(\mathbf{I}(x, y) - \mathbf{T})^T\,\mathbf{\Sigma}^{-1}\,(\mathbf{I}(x, y) - \mathbf{T})]^{1/2}$. The inverse of the covariance matrix is easily computed from the SVD as $\mathbf{\Sigma}^{-1} = \mathbf{R}\,\mathbf{S}^{-1}\,\mathbf{S}^{-1}\,\mathbf{R}^T$, yielding a simplified form of the pixel similarity function as the magnitude of a transformed vector: $\gamma(\mathbf{I}(x, y), \mathbf{T}) = |\mathbf{S}^{-1}\,\mathbf{R}^T\,(\mathbf{I}(x, y) - \mathbf{T})|$.

The pixel similarity γ is shown in graphic form for the soccer player's jersey in

Figure 3.2(e). In it, $\gamma(\mathbf{I}(x,y),\mathbf{T}) = 0$ is represented as a white pixel (gray level 255) at (x,y), $\gamma(\mathbf{I}(x,y),\mathbf{T}) \geq 3$ as black (gray level 0), and the gray levels of intermediate image similarities are linearly interpolated.

Geometry

Another key component of the image prediction and similarity functions is the geometric representation of R. For a region, we want to emphasize rough properties such as location, scale, aspect ratio, and orientation over its precise shape. We therefore characterize a region as a rectangle parametrized by image position x, y, size w, h, and orientation ϕ. The rectangle C used to represent R is the best-fitting one according to the criterion that it minimizes the objective function $f_R(C) = |R - C| + |C - R|$, where $|R|$ is the area of region R and $R_1 - R_2 = \{(x,y) : (x,y) \in R_1 \text{ and } (x,y) \notin R_2\}$.

A rectangle is obviously only an approximation to the actual boundary of most regions, but it is a fairly good one for many interesting tracking targets such as human body parts (face, torso, and limb segments); balls (soccer, basketball); vehicle sections (car sides, aircraft fuselages); and manipulable objects (screwdriver handles, floppy disks, etc.).

Furthermore, the rectangular shape approximation assumes that the patch P giving rise to R does not rotate out of plane (i.e., out of a plane parallel to the image plane) or undergo nonrigid deformations. In general, such occurrences can decrease the closeness of fit between C and R. However, if P is defined as the portion of the surface S visible to the viewer and S is spherical, the tracked object can rotate out of plane with two degrees of freedom without degrading C's goodness-of-fit. If S is cylindrical, the tracked object has one degree of freedom out of plane. These properties are often useful when tracking, for example, people's heads, which are roughly spherical.

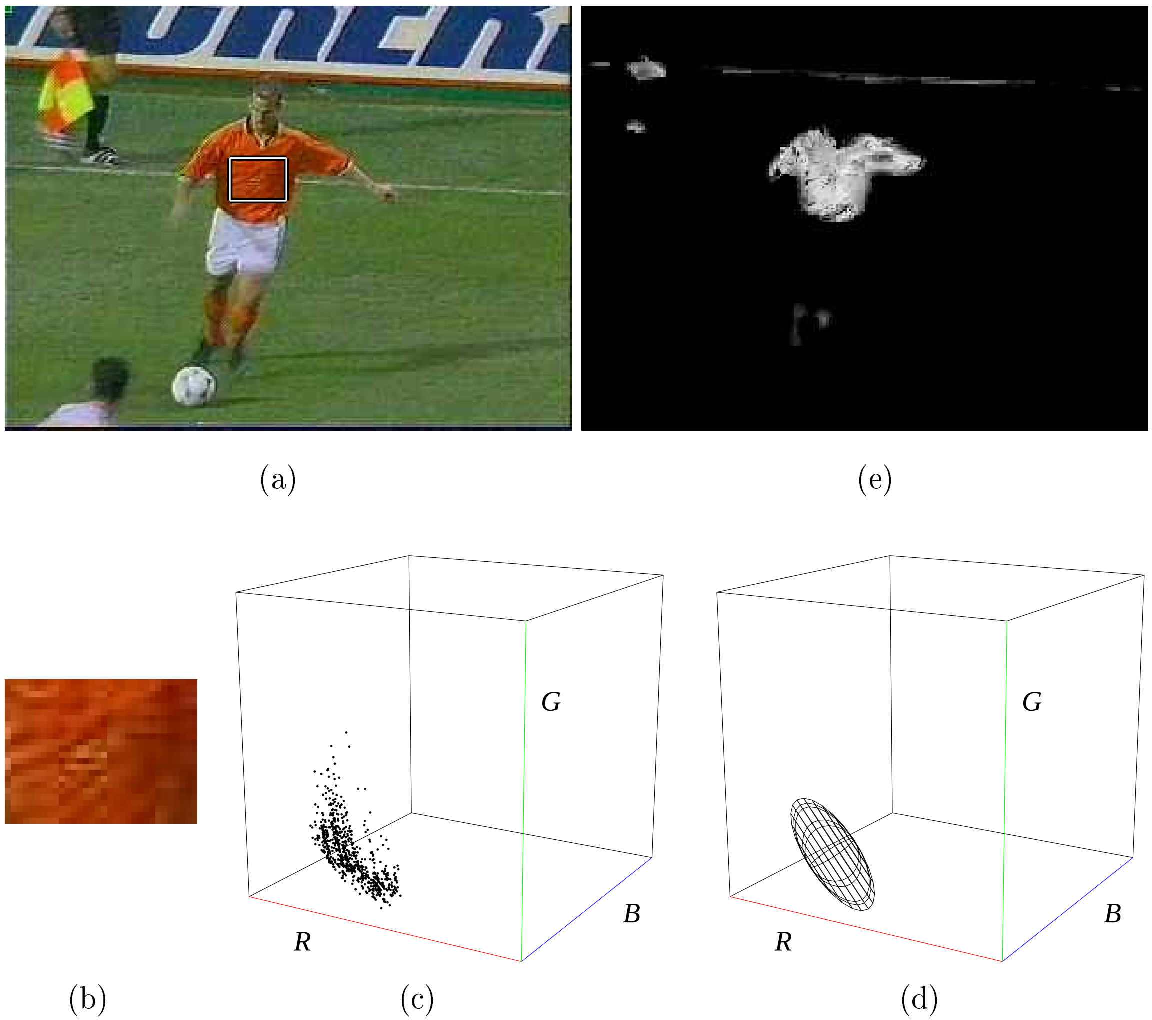
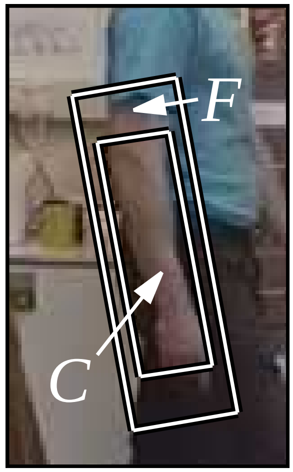
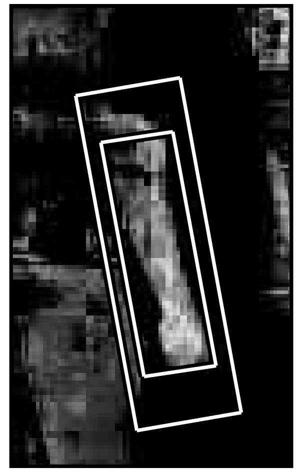


Figure 3.2: Color-based membership. (a) Initial image with location of 24×32 pixel rectangular sample overlaid; (b) Close-up of color sample; (c) RGB representation of color sample; (d) $\sigma = 1$ ellipsoid fitted to sample using principle components analysis; (e) Pixelwise color similarity γ of initial image to model via Mahalanobis distance induced by ellipsoid.



(a)



(b)

Figure 3.3: Homogeneous Region. (a) The geometry of a region for an arm tracker with the positive center C and inhibitory frame F labeled; (b) Pixel similarity γ of the image to the modeled arm skin color, with R's geometry overlaid. (Input image courtesy of J. MacCormick)

Image likelihood

The boundary of R is not known *a priori*, so two heuristics are used to identify it: (1) pixels for which γ is low are likely to be in R; and (2) pixels for which γ is high are likely to be outside R. By this logic the rectangle-fitting objective function f is minimized by minimizing the sum of γ over all pixels inside C while maximizing it outside. Efficiency allows only a relatively small area of the image outside C to be analyzed, and the reflectance contrast assumed in the second heuristic between R and its surroundings is only reasonable locally. The local image neighborhood of the *positive center* C is delineated by a rectangular border F which we call the *inhibitory frame*. To balance its influence on the rectangle-fitting objective function f, F is sized so that $|F| = |C|$ while maintaining the same aspect ratio. This means that $w_F = w_C\sqrt{2}$ and $h_F = h_C\sqrt{2}$. A good but suboptimal fit of a rectangle and its frame to a human arm region is shown in Figure 3.3.

These conditions are satisfied by the following expression for the conditional probability of a homogeneous region:

$$p_{hregion}(\mathbf{I}\,|\,\mathbf{X}) = \mathrm{sig}\,(\frac{1}{\sigma^2_{hregion}} \sum_{x,y \in C \cup F} a(x,y) \cdot \psi_{hregion}(x,y)) \tag{3.1}$$

where $\mathrm{sig}\,(x) = \frac{1}{1+e^{-x}}$ and $a(x,y)$ is the fraction of the total area $|R|$ of the region R represented by the pixel at (x,y) (e.g., $\frac{1}{|R|}$ if every pixel is counted—different if subsampling or adaptive sampling is used). $\sigma^2_{hregion}$ is a term that represents the variance of the sum; explanations of its purpose and the values assigned to it for homogeneous regions and the other modalities are given at the end of the chapter. The degree to which each pixel in the region fits the membership model is given by

$$\psi_{hregion}(x,y) = \begin{cases} -\gamma(\mathbf{I}(x,y), \mathbf{T}) & \text{if } (x,y) \in C \\ \gamma(\mathbf{I}(x,y), \mathbf{T}) & \text{if } (x,y) \in F \end{cases} \tag{3.2}$$

Referring back to the terms defined in Chapter 2, the image prediction function π for regions posits an image consisting of a rectangle of color defined by γ framed by a contrasting color, with location, size, and orientation described by the state $\mathbf{X}$.

3.1.2 Textured Regions

A *textured* region is defined as a region whose patch P has an intrinsic color pattern $c_P(P)$ with significant intensity gradients both vertically and horizontally. A strong gradient allows sum-of-squared-differences (SSD) methods [80, 109, 51] to successfully estimate the region's geometric and photometric transformations. Here we limit our attention to affine geometric transformations of an intensity patch whose projection is approximated by a rectangle.

We write $c_R(R)$ to denote the pattern, similar to a homogeneous region's color parametrization, by which a textured region R is recognized. We model $c_R(R)$ by having the user select a rectangular image sample $\mathbf{I}_R$ of the target called the *reference* image before tracking is to begin. An example of the selection step is shown in Figure 3.4(a) and the resulting reference image in Figure 3.4(c).

At any moment during tracking, the state $\mathbf{X}$ of the object specifies the shape of R as a warp of the reference image. That is, $\mathbf{X}$ postulates that the current image contains a shifted, rotated, scaled, and sheared version of $\mathbf{I}_R$, which we call the predicted image $\mathbf{I}_P$. The image prediction function π thus embeds $\mathbf{I}_P$ in the current image at the appropriate location. $\mathbf{I}_P$ can be derived from $\mathbf{X}$ by performing an affine warp $\mathbf{A}$ (assumed part of the state) with bilinear interpolation [43] on $\mathbf{I}_R$. In practice, the image inside the rectangle predicted by $\mathbf{X}$ is inversely warped using $\mathbf{A}^{-1}$ to get a comparison image $\mathbf{I}_C$ that is the same size as the reference image. An example of the predicted shape and location for the textured region referred to above is shown in Figure 3.4(b); its associated comparison image is depicted in Figure 3.4(d).

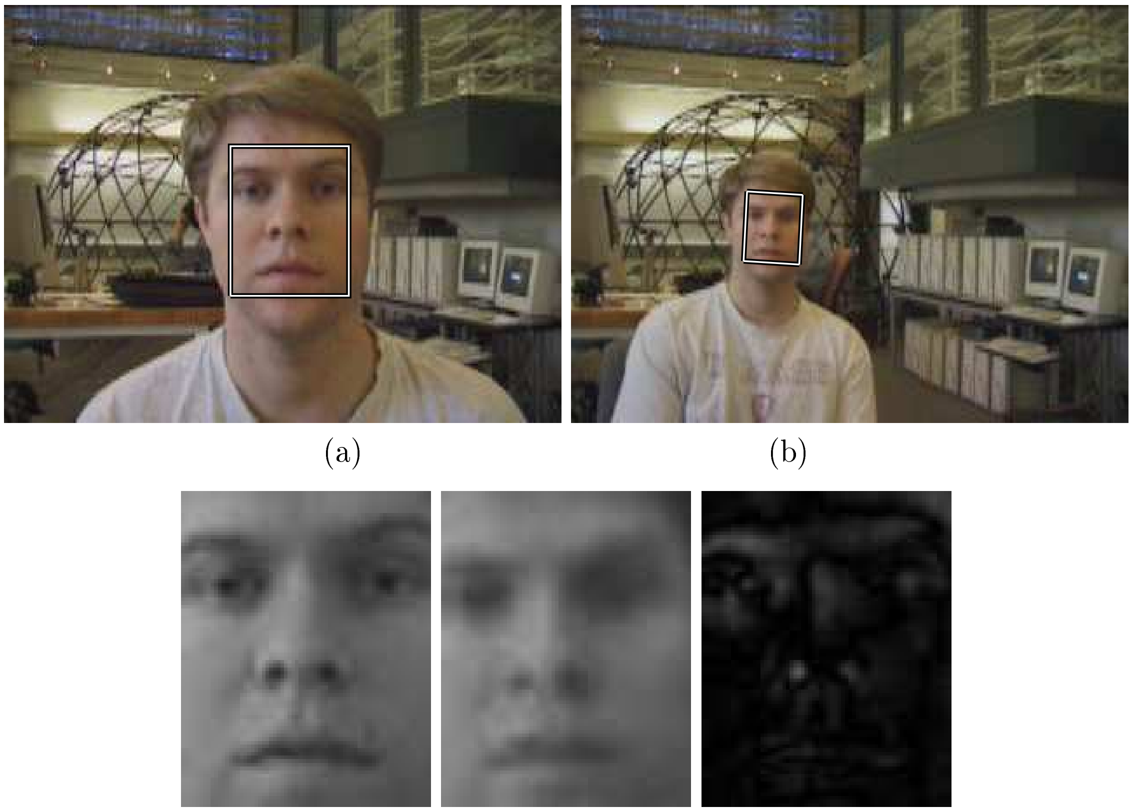
The gradient of textured regions makes feature comparison within regions sufficient to measure scaling, obviating the inhibitory frame necessary for homogeneous regions. An SSD formulation expresses the conditional probability of the image $\mathbf{I}$ given the state $\mathbf{X}$ as inversely proportional to the difference between the reference image and the comparison image:

$$p_{tregion}(\mathbf{I} \,|\, \mathbf{X}) = \exp\,(-\frac{1}{\sigma^2_{tregion}} \sum_{x,y \in \mathbf{I}_R} a(x,y) \cdot \psi_{tregion}(x,y)) \tag{3.3}$$

where $a(x,y)$ is the fraction of the total area of the reference image $|\mathbf{I}_R|$ represented by the pixel at (x,y) and

$$\psi_{tregion}(x,y) = (\mathbf{I}_R(x,y) - \mathbf{I}_C(x,y))^2 \tag{3.4}$$

An image representing the residual for the example is shown in Figure 3.4(e).



(a) (b)

(c) (d) (e)

Figure 3.4: Textured Region. (a) Selecting the reference image for a face tracker; (b) One possible state; (c) Reference image $\mathbf{I}_R$ from (a); (d) Normalized comparison image $\mathbf{I}_C$ for the state in (b); (e) Difference image $|\mathbf{I}_R - \mathbf{I}_C|$.

3.2 Snakes

We define a *snake* [18, 117] as the projection of a continuous *contour* lying on a smooth surface onto the image. Analogous to the nomenclature of patch and region in Section 3.1, we will use the term contour exclusively to refer to the curve on the surface in $\mathbb{R}^3$ and the term snake for its image projection. The contour may be either closed or open. At each point along its length the contour has a type that falls into one of the following categories: (a) it delineates an edge on the surface between regions with contrasting properties, (b) it follows the silhouette of the surface against a contrasting background, or (c) it traces a line on the surface that contrasts with the local properties on both sides. Whichever the type of the contour, the contrasting properties that define it may be intensity, color, texture, or some more complicated visual quantity. In this dissertation we assume that the contrast takes the form of an intensity difference, permitting the use of standard edge detection algorithms.

In the next two subsections we describe two edge detection techniques and a method of snake shape representation.

3.2.1 Edge Detection

A prerequisite for fitting a shape model to an intensity disparity curve is to find the edges in the image. If the image is color, it is first converted to grayscale before performing an edge detection operation. For intensity, we use the luminance component Y of the standard $RGB \rightarrow YUV$ colorspace conversion given by $Y = 0.299\,R + 0.587\,G + 0.114\,B$ [93].

Numerous approaches to extracting edges from images have been investigated in the vision literature. An in-depth discussion of the pros and cons of the various methods is beyond the scope of this dissertation, so we will simply present two that we have used successfully. The Sobel edge operator [111] is less sophisticated but has the virtue of speed, while the Canny algorithm [23] is a widely-accepted benchmark that requires considerably more processing. The two methods are described below.

Sobel

Sobel edge detection [111] approximates the gradient of the image intensity function $\nabla\mathbf{I} = (\frac{\partial \mathbf{I}}{\partial x}, \frac{\partial \mathbf{I}}{\partial y}) = (\mathbf{I}_x, \mathbf{I}_y)$ by convolving the image with a horizontal mask $\mathbf{S}_x$ and vertical mask $\mathbf{S}_y$ such that $\mathbf{I}_x \approx \mathbf{S}_x \otimes \mathbf{I}$ and $\mathbf{I}_y \approx \mathbf{S}_y \otimes \mathbf{I}$.

Each convolution mask is separable into the product of 1-D derivative masks $\mathbf{D}_x = (-1, 0, 1)$, $\mathbf{D}_y = (-1, 0, 1)^T$ and 1-D triangular smoothing masks $\mathbf{T}_x = (1, 2, 1)$, $\mathbf{T}_y = (1, 2, 1)^T$ as follows:

$$\mathbf{S}_x = \mathbf{T}_y\,\mathbf{D}_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \quad \mathbf{S}_y = \mathbf{D}_y\,\mathbf{T}_x = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{3.5}$$

The gradient magnitude of $\mathbf{I}$ indicates the strength of an edge and is approximated by $|\nabla\mathbf{I}| \approx \sqrt{(\mathbf{S}_x \otimes \mathbf{I})^2 + (\mathbf{S}_y \otimes \mathbf{I})^2}$. It can be thresholded to eliminate weak edges by setting $|\nabla\mathbf{I}(x, y)| = 0$ if $|\nabla\mathbf{I}(x, y)| < \tau$ and binarized by setting all values above τ to some non-zero constant.

The result of Sobel edge detection on a grayscale image of a person's head obtained from an infrared camera is shown in Figure 3.5(b).

Canny

Canny edge detection [23] resembles the Sobel method with extensive post-processing. The image is first convolved with a smoothing mask approximating a Gaussian with standard deviation σ. The Gaussian mask $\mathbf{G}$ is separable into horizontal and vertical

1-D masks $\mathbf{G}_x, \mathbf{G}_y$ (where $\mathbf{G}_x = \mathbf{G}_y^T$) whose widths depend on σ. Truncating the tail of the Gaussian to 0 at $2.5\,\sigma$ yields, for example, a mask width of 7 for $\sigma = 1.0$ and $\mathbf{G}_x = (0.004, 0.054, 0.242, 0.400, 0.242, 0.054, 0.004)$. Using a Gaussian instead of a the Sobel triangular smoothing mask is superior from a filtering standpoint, though somewhat more expensive.

Next, the blurred image $\mathbf{G} \otimes \mathbf{I}$ is differentiated by convolving it with the derivative masks $\mathbf{D}_x$ and $\mathbf{D}_y$. This is the same as differentiating the Gaussian mask and convolving the result with the image. Specifically, $\mathbf{G}'_x = \mathbf{D}_x \otimes \mathbf{G}$ and $\mathbf{G}'_y = \mathbf{D}_y \otimes \mathbf{G}$, yielding $\mathbf{I}_x \approx \mathbf{G}'_x \otimes \mathbf{I}$ and $\mathbf{I}_y \approx \mathbf{G}'_y \otimes \mathbf{I}$. The magnitude of the estimated gradient is computed as above: $|\nabla \mathbf{I}| \approx \sqrt{(\mathbf{G}'_x \otimes \mathbf{I})^2 + (\mathbf{G}'_y \otimes \mathbf{I})^2}$.

Due to the smoothing, image edges are correlated with rounded ridges in the gradient magnitude function rather than sharp spikes. The next step of the Canny algorithm, called non-maximal suppression, attempts to remove all but the tops of these ridges for more precise edge localization. Intuitively, we want to keep only those pixels whose edge strengths are higher than those of their two neighbors perpendicular to the direction of the edge (pixels with edge strengths of 0 are thrown out immediately). The coordinates of those neighbors for a pixel (x, y) are $(x+u, y+v)$ and $(x-u, y-v)$, where $(u, v) = (\frac{\mathbf{I}_x(x,y)}{|\nabla \mathbf{I}(x,y)|}, \frac{\mathbf{I}_y(x,y)}{|\nabla \mathbf{I}(x,y)|})$ is the unit vector along the gradient direction. $(x+u, y+v)$ and $(x-u, y-v)$ are not integer coordinates in general, so each of their gradient magnitudes must be interpolated from the four surrounding integer coordinates, one of which is always (x, y). Several different interpolation methods that trade efficiency for accuracy are commonly used; the most accurate is bilinear interpolation [43].

Finally, a form of thresholding called hysteresis is applied to remove weak edges. Rather than a fixed threshold as with the Sobel technique, upper and lower thresholds τ_{high} and τ_{low}, respectively, are employed. Pixels (x, y) for which $|\nabla \mathbf{I}(x, y)| \geq \tau_{high}$

are accepted immediately, while pixels (x, y) for which $|\nabla \mathbf{I}(x, y)| < \tau_{low}$ are rejected immediately. A pixel whose strength is between the two thresholds is only accepted if it is connected (in the eight-connected sense) to a pixel above τ_{high} by a sequence of pixels above τ_{low}. This reduces the incidence of edges being broken up, or streaking, due to minor variations in gradient magnitude around a single threshold. τ_{high} and τ_{low}, which are in units of edge strength, are typically derived from two other quantities $\hat{\tau}_{high}$ and $\hat{\tau}_{low}$ that the user sets directly. $\hat{\tau}_{high} \in [0, 1]$ is a percentile threshold on the distribution of edge strengths of the pixels surviving the non-maximal suppression step. A value of $\hat{\tau}_{high} = 0.8$, for example, indicates that τ_{high} should be set to whatever gradient magnitude is greater than 80% of the gradient magnitudes of unsuppressed pixels. $\hat{\tau}_{low} \in [0, 1]$ is just the fraction of this number that τ_{low} should be set to: $\tau_{low} = \hat{\tau}_{low}\, \tau_{high}$.

The result of Canny edge detection is a binary image of edges and non-edges. An example output based on the IR head image used in the Sobel section is shown in Figure 3.5(c).

3.2.2 Shape Representation

We represent a snake as a periodic or nonperiodic, uniform, cubic B-spline [18, 56, 120] constrained to deform affinely. The spline approach allows an arbitrarily detailed description of the shape of the tracked object, while the affine constraint efficiently captures the snake's degrees of freedom if its associated contour is a rigid, planar curve restricted to translation, scaling, and in-plane rotation. Allowing unconstrained deformations of the snake can cause mistracking because it discards information about the correct relative locations of snake segments. More complex situations such as space curves and perspective and nonrigid transformations, however, can be handled straightforwardly by adding dimensions to the state [18].

Formally, a uniform, cubic B-spline is a curve $\boldsymbol{\Lambda}$ comprising N equal-length cubic polynomial curve segments $\boldsymbol{\Lambda}_i$ between which there is second-order (C^2) continuity. The ith curve segment is defined parametrically over $0 \leq u \leq 1$ as a blend of four control points $\mathbf{p}_i, \ldots, \mathbf{p}_{i+3}$ according to:

$$\boldsymbol{\Lambda}_i(u) = \sum_{k=1}^{4} B_k(u)\, \mathbf{p}_{i+k-1} \tag{3.6}$$

Using the C^2 continuity condition between curve segments plus a constraint that $\sum_{k=1}^{4} B_k(u) = 1$ for $0 \leq u \leq 1$ (i.e., every point along a curve segment is a weighted average of the segment's control points), we can deduce four unique cubic blending functions that are valid for any curve: $B_1 = \frac{1}{6}(1-u)^3$, $B_2 = \frac{1}{6}(3u^3 - 6u^2 + 4)$, $B_3 = \frac{1}{6}(-3u^3 + 3u^2 + 3u + 1)$, and $B_4 = \frac{1}{6}u^3$. Written in matrix form, Equation 3.6 becomes:

$$\boldsymbol{\Lambda}_i(u) = \frac{1}{6}\,(u^3\ u^2\ u\ 1) \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \\ \mathbf{p}_{i+3} \end{pmatrix} \tag{3.7}$$

Let $\mathbf{P} = (\mathbf{X}, \mathbf{Y}) = (\mathbf{p}_1, \ldots, \mathbf{p}_M)^T$ be a vector of the entire B-spline's control points. For periodic curves, $M = N$; for nonperiodic curves, $M = N+3$ (assuming a cubic B-spline). The analog of the parameter u for the length of the whole curve is s, where $0 \leq s \leq N$. A vector of global blending functions $\mathbf{G}(s) = (G_1(s), \ldots, G_M(s))$ serves to transform s into its local equivalent u and pick out the supporting local blending functions B_i according to $\boldsymbol{\Lambda}(s) = \mathbf{G}(s)\,\mathbf{P}$. Each G_i is defined as:

$$G_i(s) = \begin{cases} B_4(s' - i + 4) & i - 4 \leq s' < i - 3 \\ B_3(s' - i + 3) & i - 3 \leq s' < i - 2 \\ B_2(s' - i + 2) & i - 2 \leq s' < i - 1 \\ B_1(s' - i + 1) & i - 1 \leq s' < i \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

where $s' = s$ if $\mathbf{\Lambda}$ is nonperiodic and $s' = s - 4$ if $\mathbf{\Lambda}$ is periodic and $s \geq M - 3$.

Since the snake may deform or move over multiple video frames, we index the curve and hence its control points by time: $\mathbf{\Lambda}(s,t) = \mathbf{G}(s)\,\mathbf{P}(t)$. The initial configuration of the control points $\mathbf{P}(0) = (\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ is derived from a user-selected set $\hat{\mathbf{\Lambda}}$ consisting of n points along the curve which we call the *shape template*. A simple method for doing this is to take the shape template points to be the endpoints of $\mathbf{\Lambda}$'s curve segments. For nonperiodic splines, the user chooses $n = N + 1$ roughly equally-spaced points so that $\hat{\mathbf{\Lambda}} = (\mathbf{\Lambda}(0,0), \mathbf{\Lambda}(1,0), \ldots, \mathbf{\Lambda}(N,0))^T$. The condition that the curve endpoints coincide with the first and last control points is added to ensure a unique solution, yielding the following set of simultaneous equations for $\mathbf{P}(0)$:

$$\begin{pmatrix} \mathbf{\Lambda}(0,0) \\ \mathbf{\Lambda}(0,0) \\ \vdots \\ \mathbf{\Lambda}(N,0) \\ \mathbf{\Lambda}(N,0) \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ G_1(0) & G_2(0) & \cdots & G_M(0) \\ \vdots & \vdots & & \vdots \\ G_1(N) & G_2(N) & \cdots & G_M(N) \\ 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}_1(0) \\ \mathbf{p}_2(0) \\ \vdots \\ \mathbf{p}_{M-1}(0) \\ \mathbf{p}_M(0) \end{pmatrix} \tag{3.9}$$

For a periodic curve, one fewer point in the shape template is required to specify $n = N$ segments, making $\hat{\mathbf{\Lambda}} = (\mathbf{\Lambda}(0,0), \mathbf{\Lambda}(1,0), \ldots, \mathbf{\Lambda}(N-1,0))^T$. Also, there are two fewer unknowns and thus no need for extra conditions. The periodic analog of Equation 3.9 is thus $\hat{\mathbf{\Lambda}} = (\mathbf{G}(0), \ldots, \mathbf{G}(N-1))^T \, \mathbf{P}(0)$.

The affine representation $\mathbf{Q}(t)$ of the snake is derived from the B-spline representation $\mathbf{P}(t)$ as follows [18]:

$$\mathbf{Q}(t) = \mathbf{M} \left[\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t) \end{pmatrix} - \begin{pmatrix} \widehat{\mathbf{X}} \\ \widehat{\mathbf{Y}} \end{pmatrix} \right] \tag{3.10}$$

where

$$\mathbf{M} = (\mathbf{W}^T \, \boldsymbol{\mathcal{H}} \, \mathbf{W})^{-1} \, \mathbf{W}^T \, \boldsymbol{\mathcal{H}} \tag{3.11}$$

$\mathbf{W}$ is an affine basis defined as:

$$\mathbf{W} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \widehat{\mathbf{X}} & \mathbf{0} & \mathbf{0} & \widehat{\mathbf{Y}} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \widehat{\mathbf{Y}} & \widehat{\mathbf{X}} & \mathbf{0} \end{pmatrix} \tag{3.12}$$

where $\mathbf{0} = (0, \ldots, 0)^T$, $\mathbf{1} = (1, \ldots, 1)^T$ are M-vectors, and $\boldsymbol{\mathcal{H}}$ is the metric matrix [16] given by:

$$\boldsymbol{\mathcal{H}} = \begin{pmatrix} \sum_{s=0}^{N} \mathbf{G}^T(s) \, \mathbf{G}(s) & 0 \\ 0 & \sum_{s=0}^{N} \mathbf{G}^T(s) \, \mathbf{G}(s) \end{pmatrix} \tag{3.13}$$

Variations in the snake's control points over time are described by an affine transformation of the shape template's control points:

$$\begin{pmatrix} \mathbf{X}(t) \\ \mathbf{Y}(t) \end{pmatrix} - \begin{pmatrix} \widehat{\mathbf{X}} \\ \widehat{\mathbf{Y}} \end{pmatrix} = \mathbf{W} \, \mathbf{Q}(t) \tag{3.14}$$

Thus, the full B-spline can be deduced from $\mathbf{Q}$, but is restricted to a smaller-dimensional affine subspace of the configurations than would be possible if $\mathbf{P}$ were unconstrained.

3.2.3 Image likelihood

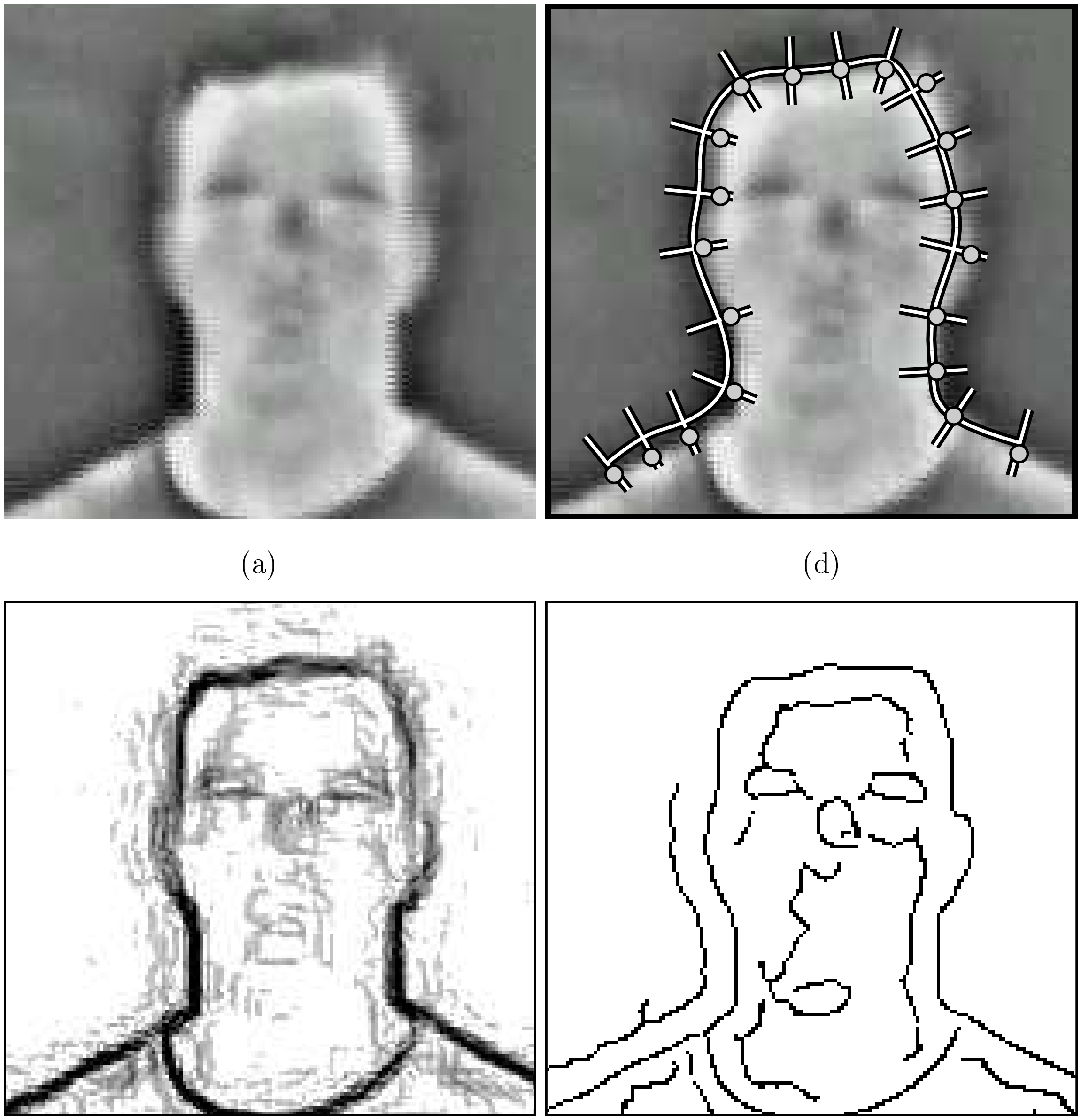
The image prediction function π for snakes hypothesizes a curve derived from $\mathbf{Q}$ along which there is an intensity disparity. To compute the image similarity between the image and this prediction, we define $p(\mathbf{I}\,|\,\mathbf{X})$ by adapting the formula for "$p(z\,|\,x)$" as described in [59].

For each of the n segment borders comprising the B-spline parametrized by a particular $\mathbf{Q}$, edge detection is performed along a line of length L (typically 10-20 pixels) that is normal to and bisected by the curve at that point. Let $\mathbf{\Lambda}(i)$ be the image location of the curve at segment i, where $0 \leq i < n$. Using the Canny algorithm, we let $\mathbf{z}(i)$ be the location of the edge segment along the ith normal that is found nearest to $\mathbf{\Lambda}(i)$. For the Sobel method, $\mathbf{z}(i)$ is the strongest edge along the normal whose strength is over the threshold τ. Any failure to find a suitable edge is noted and dealt with as described in Equation 3.16 below. The shape of a nonperiodic snake and the Sobel edges detected on its normals are illustrated in Figure 3.5(d).

Assuming the state $\mathbf{X}$ includes $\mathbf{Q}$, we express the likelihood as:

$$p_{snake}(\mathbf{I}\,|\,\mathbf{X}) = \exp\left(-\frac{1}{\sigma^2_{snake}}\sum_{i=0}^{n-1} l(i)\cdot\psi_{snake}(i)\right) \tag{3.15}$$

where $l(i)$ is the fraction of the total length $|\mathbf{\Lambda}|$ of the snake represented by normal i (e.g., $N/|\mathbf{\Lambda}|$ if the normals are evenly spaced). The degree to which the location of each detected edge fits the shape model is given by



(a) (d)

(b) (c)

Figure 3.5: Snake. (a) Human head (infrared image); (b) Sobel edge detection on head image ($\tau = 45$; brightness at pixel (x, y) is linearly interpolated after thresholding between $|\nabla \mathbf{I}(x, y)| = 0$ as white and $|\nabla \mathbf{I}(x, y)| \geq 255$ clamped to black); (c) Canny edge detection ($\sigma = 2.0$, $\hat{\tau}_{low} = 0.25$, $\hat{\tau}_{high} = 0.75$); (d) State of a sample snake head tracker. Circles on curve normals indicate locations of strongest Sobel edges.

$$\psi_{snake}(i) = \begin{cases} |\mathbf{\Lambda}(i) - \mathbf{z}(i)| & \text{if an edge is found} \\ \xi & \text{otherwise} \end{cases} \tag{3.16}$$

ξ serves as a penalty value for $\psi(i)$ when there is no edge detected along the normal to the curve at segment i.

3.3 Addenda

A factor in the construction of the image likelihood that deserves scrutiny is the value of the variance σ^2 for each modality. The purpose of σ^2 is to ensure that the ranges of the functions sig in Equation 3.1 and exp in 3.3 and 3.15 are within machine precision. To see the necessity of this, observe that the ψ term of each of the modalities has a natural scale depending on its definition. For textured regions, the value of $\psi_{tregion}$ is a squared pixel intensity difference ranging from $0^2 = 0$ to $255^2 = 65025$. For homogeneous regions, $\psi_{hregion}$ is a Mahalanobis distance between pixel colors which can range from 0 to roughly 10 depending on a region's color definition. A snake's ψ_{snake} is a distance in the image between a predicted and measured edge; with the penalty term ξ it ranges from 0 to ξ (which is usually 10-30). Appropriate modality-specific values for σ^2 can be empirically derived—e.g., by averaging the inverse of the mean ψ for a modality over many object models, samples, and images—but we simply use reasonable approximations: $\sigma^2_{tregion} = 4000$, $\sigma^2_{hregion} = 25$, and $\sigma^2_{snake} = 400$.

The ellipse-fitting idea for homogeneous regions is easily adaptable to grayscale images by carrying out one-dimensional principal components analysis (PCA) on the sampled pixel intensities. Such grayscale images might be the output of a depth-estimating stereo algorithm [31, 70]. In order to reduce sensitivity to variations in illumination intensity, we may also use a similar method on the two chrominance dimensions of color space representations such as YUV [93]. Computer graphics or synthetic image sources where a region's color may be perfectly uniform are not amenable to statistical techniques such as PCA. In this case it is straightforward to model color similarity as simple Euclidean distance in color space.

Variations on the method of fitting an ellipsoid to a color sample include using robust methods [85] for dealing with outliers in the reflectances of sampled pixels (such as the logo on the player's shirt in Figure 3.2). Instead of rectangles for representing shape of regions (homogeneous and textured), we might use periodic B-spline outlines for more detail. This would require a simple generalization of the notion of the inhibitory frame for homogeneous regions.

For all of the modalities discussed in this chapter, the model of the image prediction function $\pi(\mathbf{X})$ neglects depth-dependent focus, motion blur, and other such photorealistic factors. Ignoring the effects of these phenomena has a negligible impact on the quality of tracking for the image sequences we use.

Chapter 4

Single Object Tracking

In this chapter we discuss techniques for tracking single, unoccluded, *atomic* objects. We use the term *atomic* in the sense of an atom being the smallest indivisible unit, meaning that the object is identified by only one of the modalities presented in the previous chapter. The combination of an identifying modality and the observable parameters of that modality (size, color, shape, etc.) constitute an *attribute* of an object.

In addition to solitary objects, the methods we cover can be applied to tracking multiple objects simultaneously, but with degraded performance when similar objects are close to one another or the objects occlude one another. Moreover, if the objects to be tracked are physically linked (e.g., parts of a human body) or are identified by multiple attributes, these methods ignore information provided by the inter-object constraints or extra attributes that may provide greater robustness. Explicit approaches to multiple object tracking are presented in Chapter 5, and constrained and multi-attribute tracking are investigated in Chapter 6.

Referring back to the exposition in Chapter 2, we cast the problem of visually tracking an atomic object as one of following the area of the image that is the

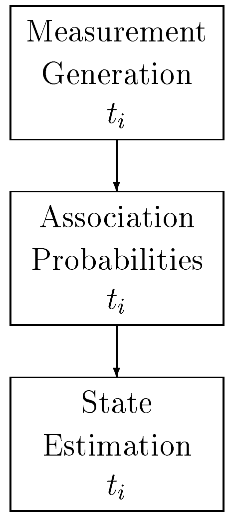


Figure 4.1: State update algorithm pipeline for a single PDAF tracker

best match to it. The Kalman filter predicts the most likely location and other characteristics of this image area, indicating where to begin searching for it. In the first section of this chapter, we discuss methods for finding and parametrizing a set of hypotheses for good matches. The best match thus found is suitable for input to a standard Kalman filter as the measurement. Later in the chapter we present the Probabilistic Data Association Filter (PDAF) [5], an extension to the Kalman filter that considers other highly likely alternatives.

4.1 The Measurement Process

The measurement extraction process is essentially a search for local maxima of the image likelihood $p(\mathbf{I} \mid \mathbf{X})$ in the neighborhood of $\widehat{\mathbf{X}}$, the state predicted from the filter at time t. The geometric characteristics of the image areas corresponding to these maximally likely states are derived as measurements $\mathbf{Z}$. There are many approaches to this problem, each of which trades off speed for thoroughness. Perhaps the simplest class of techniques, which we call *gradient ascent* methods, follows from the condition that $p(\mathbf{I}|\mathbf{X})$ is differentiable. Randomized methods such as simulated annealing [87] which sample $p(\mathbf{I}\,|\,\mathbf{X})$ at discrete locations, sequentially or in parallel, have proven successful at finding global maxima of multimodal functions and do not require differentiation in order to work. Alternatives to random sampling include sampling over a regular grid in state space or using a quasi-random method such as Antonov-Saleev [94] to achieve a Poisson-like distribution and thus avoid duplication of effort by samples that are too close to one another.

In this chapter we use either gradient ascent or random sampling methods for measurement extraction. The basics of each approach are covered in the next two subsections. We have found that a hybrid of the two methods yields good results. An algorithm for combining these techniques and an analysis of why such a combination may be necessary or preferable are presented in the next chapter.

4.1.1 Gradient Ascent

Common gradient algorithms include *steepest ascent* and *conjugate gradient* [94]. These methods work as follows. Let the function to be maximized be $f(\mathbf{X}) = p(\mathbf{I}\,|\,\mathbf{X})$, and the starting point of the search be $\mathbf{p}_0 = \mathbf{Z}$, the current predicted state. The aim of the gradient algorithms is to construct a sequence of points $\{\mathbf{p}_i\}$ that converge to a maximum of f that is local to $\mathbf{p}_0$. Each new point $\mathbf{p}_{i+1}$ is derived from its predecessor according to $\mathbf{p}_{i+1} = \mathbf{p}_i + \lambda_i\,\mathbf{g}_i$, where $\mathbf{g}_i$ is a direction derived (indirectly, if using the conjugate method) from the gradient $\nabla f(\mathbf{p}_i)$. λ_i is a step size that can be computed from minimizing f along an interval of the line passing through $\mathbf{p}_i$ in the direction of $\mathbf{g}_i$, or it may simply be constant. The algorithm is terminated when $|f(\mathbf{p}_{i+1}) - f(\mathbf{p}_i)| \leq \delta$ for some small δ or the number of steps taken reaches a threshold N.

While conjugate gradient uses derivative information on f for efficiency, *Powell's method* [94] is a related algorithm that does not. Powell's method is sometimes used in this dissertation, especially for the modified version of $p(\mathbf{I} \mid \mathbf{X})$ introduced in Chapter 5. Unless otherwise noted, though, the gradient ascent technique used is conjugate gradient.

We will now describe some specific methods of implementing gradient ascent for the tracking modalities introduced in Chapter 3. Assuming that the objective function $f(\mathbf{X})$ can be evaluated for any particular state, we can always approximate the partial derivative of f with respect to some parameter x in $\mathbf{X}$ by $\frac{\partial f}{\partial x} \approx \frac{f(\mathbf{X}+\boldsymbol{\Delta x})-f(\mathbf{X}-\boldsymbol{\Delta x})}{2\Delta x}$, where $\boldsymbol{\Delta x}$ is a vector that is all 0's except for a small number Δx assigned to the x parameter. The gradient can easily be constructed from these approximated derivatives. For consistency, we use this method of gradient approximation for all three modalities, although some tracking modalities permit the use of more direct gradient ascent methods.

For example, a method commonly used in snake tracking (introduced in [117] and extended to the affine case in [18]) is to simply take the locations of the best edges $\mathbf{z}(0), \ldots, \mathbf{z}(n-1)$ found along the search normals of the predicted snake, use these to parametrize a new B-spline, and create a measurement from this B-spline. The maximal motion that can be estimated in this manner obviously depends on the length of the normals.

The motion of textured regions can be directly estimated by solving the image brightness equation $\mathbf{I}_x u + \mathbf{I}_y v + \mathbf{I}_t = 0$ [51, 54] ($\mathbf{I}_x, \mathbf{I}_y$ are the spatial image derivatives, $\mathbf{I}_t$ is the temporal image derivative, and u, v are pixel velocities within the part of the image of interest) subject to, for example, an affine constraint [9]. This approach can be extended to estimate larger motions by using a multi-resolution image pyramid to iteratively move from coarse to fine estimates [9].

Gradient methods are often used to efficiently obtain a single best measurement with which to update a tracking filter. A number of assumptions must be met to ensure their successful application. First, gradient ascent works best when $p(\mathbf{I}\,|\,\mathbf{X})$ is unimodal. The user must also hope that the object state is changing slowly enough that the filter can keep up if the algorithm is terminated after a maximum number of steps (meaning that it may only get some fraction of the way to the true maximum for each new image), or if the posterior is multimodal that the predicted state will not wind up in another basin of attraction. Another assumption if there is multimodality is that there will not be significant interference between modes, such as two modes (the correct one and an incorrect one) merging and splitting, creating the possibility of the filter making the wrong choice after the split.

These difficulties with gradient methods are why in many situations we favor other algorithms that allow for faster state changes and multiple modalities in the state posterior. These methods, though retaining the notion of locality around a predicted state, shade into global optimization algorithms. We use gradient ascent methods for measurement generation in some of the examples at the end of this chapter in order to compare their performance to the randomized peak finder described in the next section.

4.1.2 Random sampling

For speed and simplicity, we use a measurement generation method which we call *measurement sampling*, adapted from the factored sampling approach of the Condensation algorithm [59] (the Condensation algorithm is discussed in detail in Chapter 7). Intuitively, we sample $p(\mathbf{I}\,|\,\mathbf{X})$ in the neighborhood of the current predicted state $\widehat{\mathbf{X}}$, select those state samples most likely to be on its nearby peaks, and convert them to measurements.

The random sampling method of measurement generation for a given tracking modality obtains measurements by picking points from a distribution determined by the prior on the state $p(\mathbf{X})$, computing their image likelihoods $p(\mathbf{I} \mid \mathbf{X})$, throwing away all but the top fraction, and deriving the measurement parameters of what remains. The form of the prior $p(\mathbf{X})$ can be determined by learning from examples if desired; we use a hand-tuned Gaussian.

In practice, N samples are taken from a normal distribution in the target's state space $\mathcal{X}$ centered on its current predicted state $\widehat{\mathbf{X}}$. A hypothetical prior and one set of samples derived from it are diagrammed in Figures 4.2(a) and (b). N and the covariance of the distribution $\mathbf{\Sigma}_{\mathcal{X}}$ are chosen to give adequate coverage to a "tracking window" about the target. Furthermore, sampling from the normal distribution ensures that the image likelihood function is examined more precisely where it is expected to be highest. $p(\mathbf{I} \mid \mathbf{X})$ is computed for each sample by scoring the degree of fit between the hypothesized target and the current image, as shown in Figure 4.2(c). Finally, a winnowing step sorts the samples by their likelihoods and keeps only the n most likely ones ($n \ll N$) to be converted to measurements for input to the tracking filter. This last step is illustrated in Figure 4.2(d).

Several enhancements to this basic procedure are possible. First, we could enforce a minimum distance in state space between samples to reduce the chance of multiple samples being drawn from the same underlying image feature. Also, performing gradient ascent on the samples, either before or after winnowing, would improve their quality and consistency. Both of these steps are optional at this stage because of the PDAF's tolerance of multiple samples, but we show in the next chapter that they are necessary for joint tracking.

4.1.3 Measurement Terminology

We use the following nomenclature to describe state and measurement parameter subspaces for homogeneous and textured regions as well as snakes: X indicates

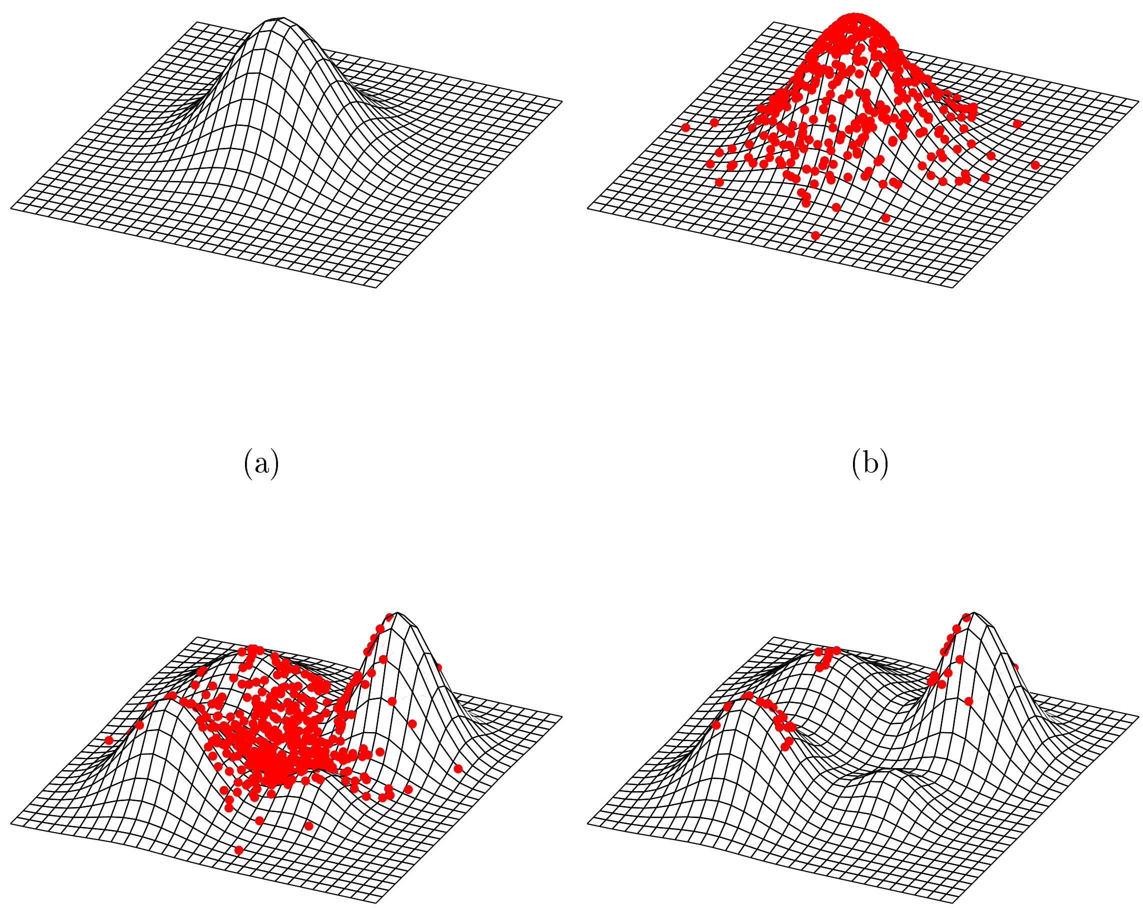


Figure 4.2: Measurement generation. (a) Hypothetical prior distribution; (b) Prior distribution on state with $N = 500$ samples; (c) Hypothetical image likelihood function with samples; (d) Image likelihood function with $n = 50$ best samples.

the range of horizontal image locations of the target, Y is the range of its vertical image locations, Φ is its possible image orientations in radians, and S is its possible scales as a fraction of its initial size. Thus, image location variables in the state are denoted by $x \in X$ and $y \in Y$, orientations by $\phi \in \Phi$, and scales by $s \in S$. Other possible state parameters to be estimated from these measurable quantities are velocity and acceleration. Horizontal image velocity and acceleration, for example, are indicated by $\dot{x} \in \dot{X}$ and $\ddot{x} \in \ddot{X}$, respectively. In all of the examples in this dissertation, an object's measurement space is simply its state space without any temporal parameters.

Because the state of an object may be as simple as, for example, its image location (x, y), we should note that the image likelihood $p(\mathbf{I} \,|\, \mathbf{X})$ is implicitly conditioned on other object information. The color of a homogeneous region, the reference image of a textured region, and the control points of a snake all go into the calculation of the image likelihood and yet are not expressed explicitly in the state. Moreover, what we call an object's geometric image processing parameters are also part of this calculation. The geometric image processing parameters are those that totally define the shape of an object and are thus necessary to compute $p(\mathbf{I} \,|\, \mathbf{X})$ for their respective modalities. For a region, they are the position $\tilde{x}, \tilde{y}$, size $\tilde{w}, \tilde{h}$, and orientation $\tilde{\phi}$ of the rectangle, and for a snake they are the six affine parameters $(\tilde{q}_0, \tilde{q}_1, \ldots, \tilde{q}_5)$ in $\mathbf{Q}$. If not included in the state, the geometric parameters must be accounted for via some other means.

As a matter of implementation, it is useful to define a formalism for referring to these geometric parameters. This formalism, which we call the *measurement key*, is also used as a means of constraint enforcement in Chapter 6. The measurement key $\mathbf{K}$ for a particular object modality is a vector of functions k_i, where $k : \mathcal{X} \to \mathbb{R}$. Each of these functions is named after the geometric parameter it corresponds to. The measurement key is different depending on the tracker modality and state space, but in general we define $\mathbf{K}_{hregion}(\mathbf{X}) = \mathbf{K}_{tregion}(\mathbf{X}) \equiv (\tilde{x}(\mathbf{X}), \tilde{y}(\mathbf{X}), \tilde{w}(\mathbf{X}), \tilde{h}(\mathbf{X}), \tilde{\phi}(\mathbf{X}))^T$ and $\mathbf{K}_{snake} \equiv (\tilde{q}_0(\mathbf{X}), \tilde{q}_1(\mathbf{X}), \tilde{q}_2(\mathbf{X}), \tilde{q}_3(\mathbf{X}), \tilde{q}_4(\mathbf{X}), \tilde{q}_5(\mathbf{X}))^T$.

We now explain the details of $\mathbf{K}$ using regions as an example. When state space $\mathcal{X}$ contains $X \times Y \times W \times H \times \Phi$, the functions in $\mathbf{K}$ simply select the appropriate entry of $\mathbf{X} \in \mathcal{X}$. If, for example, $\mathbf{X} = (x, y, w, h, \phi)^T$, then $\tilde{x}(\mathbf{X}) = x$, $\tilde{y}(\mathbf{X}) = y$, and so on. If state space does not contain all of the geometric image processing variables, though, then some are left unspecified by this method. Suppose $\mathcal{X} = X \times Y$. In this case constants are used for the unspecified parameters—specifically, their initial, user-chosen values: $\tilde{w}(\mathbf{X}) = \bar{w}$, $\tilde{h}(\mathbf{X}) = \bar{h}$, and $\tilde{\phi}(\mathbf{X}) = \bar{\phi}$. This functional formulation also allows scale s (initially 1) and other quantities to be state variables. If $\mathcal{X} = X \times Y \times S$, then $\tilde{x}(\mathbf{X}) = x$, $\tilde{y}(\mathbf{X}) = y$, $\tilde{w}(\mathbf{X}) = s\bar{w}$, $\tilde{h}(\mathbf{X}) = s\bar{h}$, and $\tilde{\phi}(\mathbf{X}) = \bar{\phi}$.

An analogy is easily drawn between the geometric parameters of regions and the affine representation of snakes. $\tilde{q}_0$ and $\tilde{q}_1$ are the *relative* translational components of the affine parameters, and thus are equal to $\tilde{x} - \hat{\mathbf{\Lambda}}_x$ and $\tilde{y} - \hat{\mathbf{\Lambda}}_y$, respectively ($\hat{\mathbf{\Lambda}}_x$ is the mean x value of the initial positions of the B-spline template points and $\hat{\mathbf{\Lambda}}_y$ is their mean y value). $\tilde{q}_2, \tilde{q}_3, \tilde{q}_4$, and $\tilde{q}_5$ are the rotational and scaling parameters of the B-spline. They are coefficients of a 2×2 matrix $\left(\begin{smallmatrix} \tilde{q}_2 & \tilde{q}_5 \\ \tilde{q}_4 & \tilde{q}_3 \end{smallmatrix}\right) = \left(\begin{smallmatrix} s\cos(\phi) & -\sin(\phi) \\ \sin(\phi) & s\cos(\phi) \end{smallmatrix}\right)$ that takes the initial shape of the B-spline to its current size and orientation. The snake's angle ϕ can thus be expressed as $\sin^{-1}(\tilde{q}_4)$ and its scale s as $\tilde{q}_2/\cos(\phi)$. Henceforth, we will use the names of the geometric parameters of regions with the knowledge that we can convert back and forth to the snake parameters.

Although the random sampling procedure for measurement generation uses the full state space, as noted in the previous chapter our model of image formation does not currently account for an object's velocity. Therefore velocity parameters in the state never figure in the calculation of the geometric parameters described above or, consequently, the image likelihood $p(\mathbf{I}\,|\,\mathbf{X})$. This means that the temporal dimensions of state samples are irrelevant to the image likelihood. In our results, we ignore them by referring to only the non-temporal dimensions of the state sampling covariance $\mathbf{\Sigma}_{\mathcal{X}}$. As a matter of shorthand, when state space includes velocity parameters such as, for example, $X \times Y \times \dot{X} \times \dot{Y}$, we describe only the relevant part of the sampling covariance, or $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} \sigma_X^2 & 0 \\ 0 & \sigma_Y^2 \end{smallmatrix}\right)$.

Homogeneous Regions

The process of random sampling to obtain measurements is illustrated for homogeneous regions in Figure 4.3. The input image is the soccer player from the previous chapter, and we would like to track his orange jersey. Allowing translation, rotation, and scaling of the rectangle fit to the jersey, state space is $\mathcal{X} = X \times Y \times \Phi \times S$. In Figure 4.3(b-c), there are a large number of samples $N = 2000$ and measurements $n = 100$, with a large sampling covariance about the predicted state in order to explore most of the image: $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 4000 & 0 & 0 & 0 \\ 0 & 4000 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.05 \end{smallmatrix}\right)$. In Figure 4.3(d-e), the sampling covariance is more focused: $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 400 & 0 & 0 & 0 \\ 0 & 400 & 0 & 0 \\ 0 & 0 & 0.02 & 0 \\ 0 & 0 & 0 & 0.01 \end{smallmatrix}\right)$. The number of samples $N = 500$ and measurements $n = 10$ are also smaller, and as a result the player's orange socks are not found, making the measurement distribution unimodal instead of bimodal.

For certain image conditions, a fast approximation to state sampling for homogeneous regions is offered by connected components (CC) analysis. This procedure follows from the observation that the moments of the largest connected components of pixels for which $\gamma(\mathbf{I}(x, y))$ is high frequently correspond to peaks in the image likelihood function $p(\mathbf{I}\,|\,\mathbf{X})$. The process, illustrated in Figure 4.4, approximates $p(\mathbf{I}\,|\,\mathbf{X})$ when $\mathbf{X}$ consists solely of position by calculating $\gamma(\mathbf{I}(x, y))$ for each image pixel (x, y). We then threshold these likelihoods to remove everything but the tops of the peaks and perform some number E of morphological expansion operations to join pixels

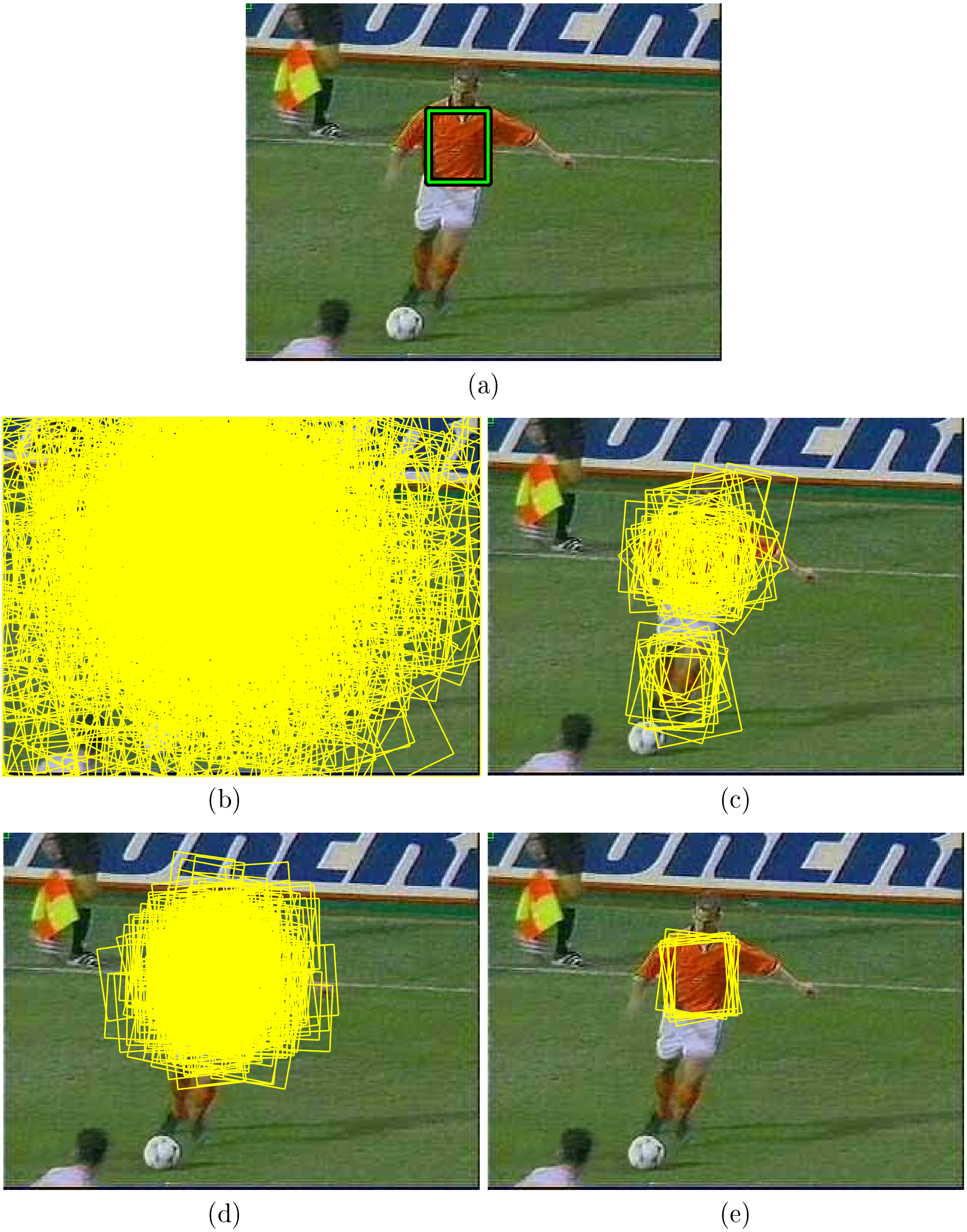


Figure 4.3: Random sampling for homogeneous region measurement generation. (a) Predicted state; (b) Samples with large covariance (about predicted state); (c) Measurements for large covariance sample; (d) Samples with small covariance (about same predicted state); (e) Measurements for small covariance sample.

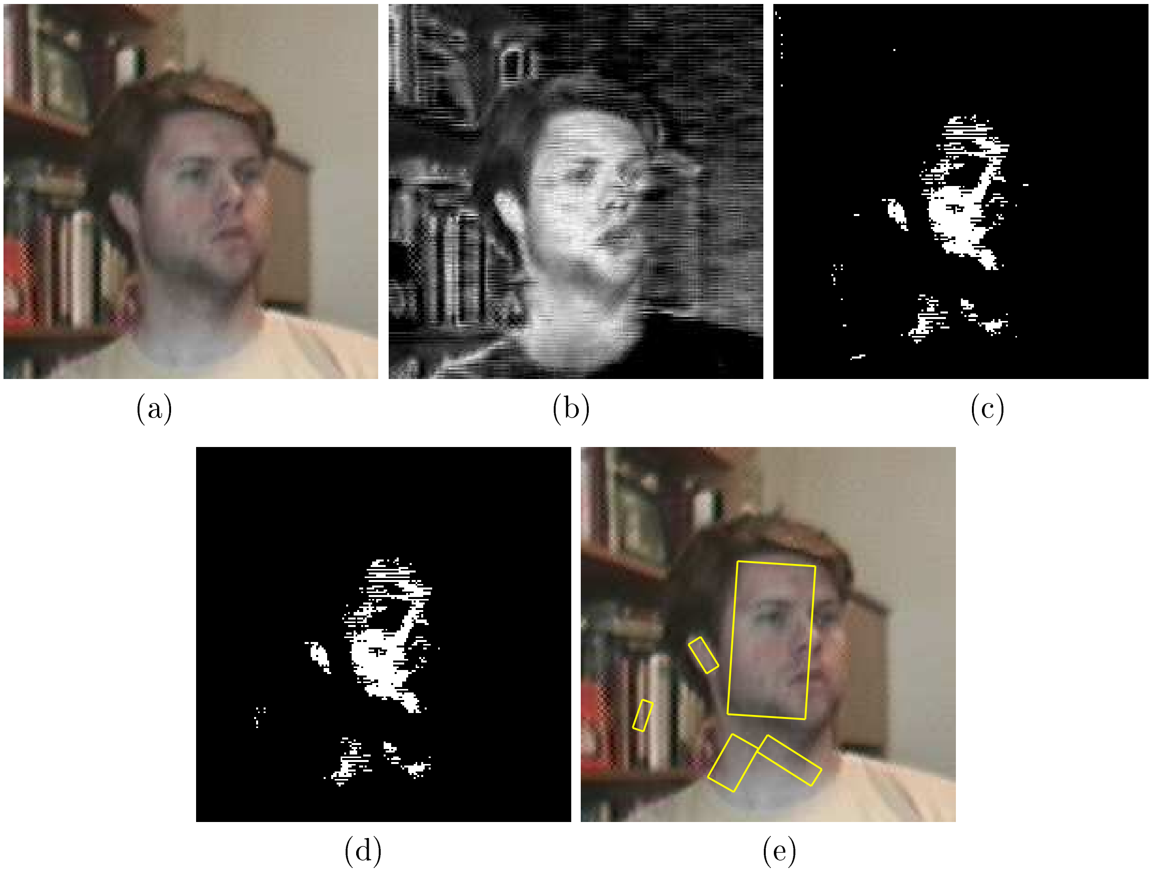


Figure 4.4: Connected components for homogeneous region measurement generation. (a) Tracking window; (b) Mahalanobis distance of pixels in RGB space to skin color; (c) Pixels over a threshold of color similarity; (d) Largest connected components of skin-colored pixels ($E = 2$); (e) Measurements derived from connected components.

separated by image artifacts such as video interlacing and camera noise. Next, we compute the connected components (in the eight-connected sense) of what remains, throw away CC's whose areas fall below a threshold. Finally, a measurement is created from each remaining CC by fitting an ellipse to it with principal components analysis and using the major and minor axes the ellipse to define a rectangle.

With the CC method, the surface patch being tracked may project to multiple regions if it is nonplanar, resulting in one target giving rise to multiple measurements. This is not a problem for position estimation, but it prevents accurate computation of the target's image size and orientation. Moreover, thresholding color similarity inevitably omits some worthy pixels falling just below the threshold, as with the right side the of face in Figure 4.4, biasing the measurement for that connected component. The quality of the CC method's results is best when $p(\mathbf{I} | \mathbf{X})$ has distinct peaks with compact support.

A similar approach to extracting measurements by calculating the connected components of thresholded grayscale images was reported by Kumar *et al.* [78].

Textured Regions

The state sampling process for a textured region tracker constrained to translation is shown in Figure 4.5. The object of interest is the face of a player in a cropped photograph of the 1926 New York Yankees baseball team. The multiple faces in this group picture clearly illustrate the potential multimodality of $p(\mathbf{I} \mid \mathbf{X})$ and its implications for measurement generation. For simple translation of the region, state space is $\mathcal{X} = X \times Y$. In Figure 4.5(b-c), relatively many samples and measurements ($N = 3000, n = 50$) are used, plus a large sampling covariance $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 1000 & 0 \\ 0 & 1000 \end{smallmatrix} \right)$. Figure 4.5(b) shows the initial samples and 4.5(c) shows the measurements obtained from the top fraction of the samples. The results of a smaller sampling covariance, $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 100 & 0 \\ 0 & 100 \end{smallmatrix} \right)$, and fewer samples and measurements ($N = 250, n = 5$) are similarly displayed in Figure 4.5(e-f).

Snakes

The steps of the sampling process for a chess pawn-shaped snake constrained to translation only are illustrated in Figure 4.6. Allowing only translation of the snake, state space is $\mathcal{X} = X \times Y$. In Figure 4.6(c-d), a relatively large number of samples $N = 2000$ and measurements $n = 50$ are used, as well as a large sampling covariance $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 3000 & 0 \\ 0 & 3000 \end{smallmatrix} \right)$. A tighter sampling covariance is shown in Figure 4.6(e-f): $\mathbf{\Sigma}_{\mathcal{X}} =$

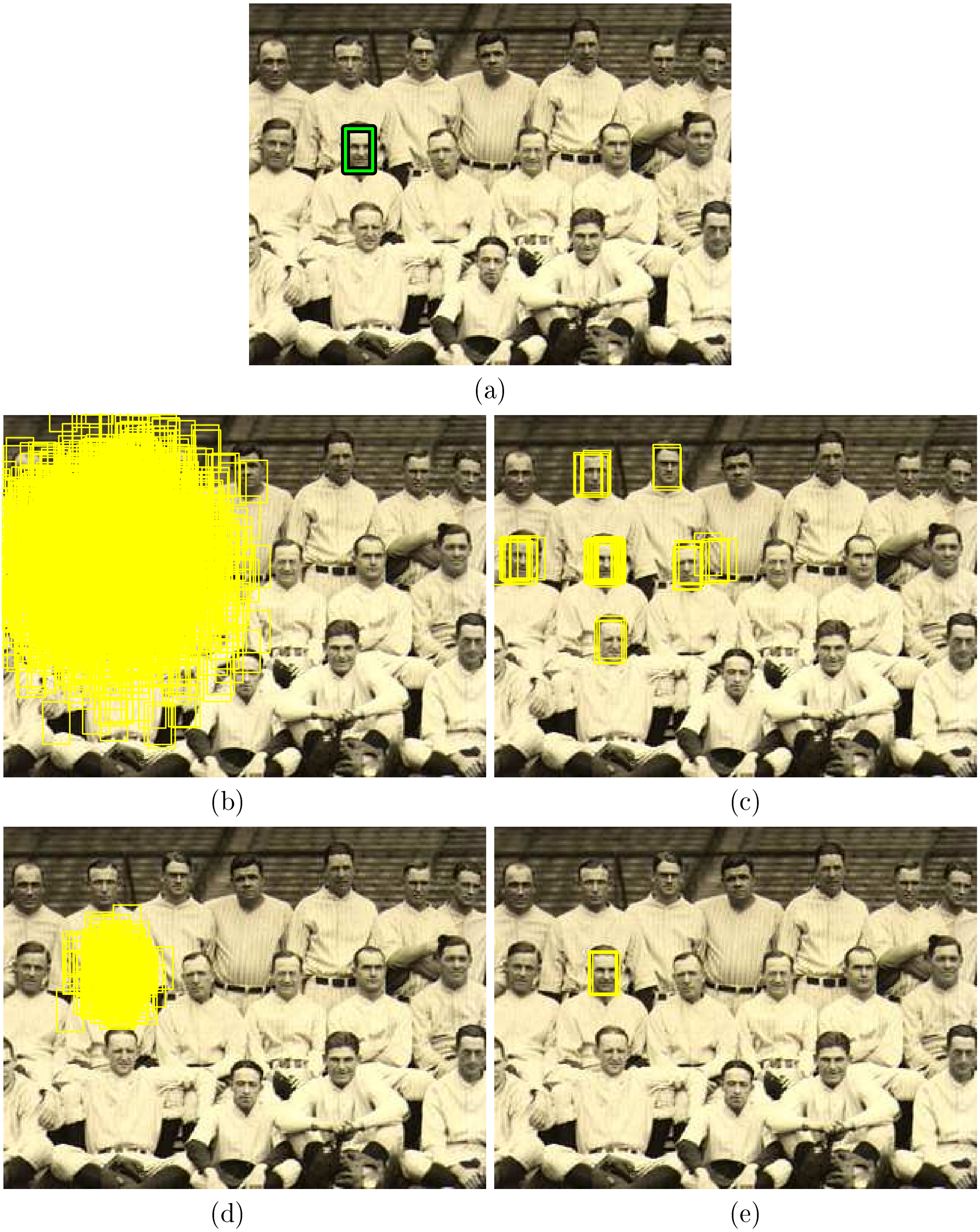
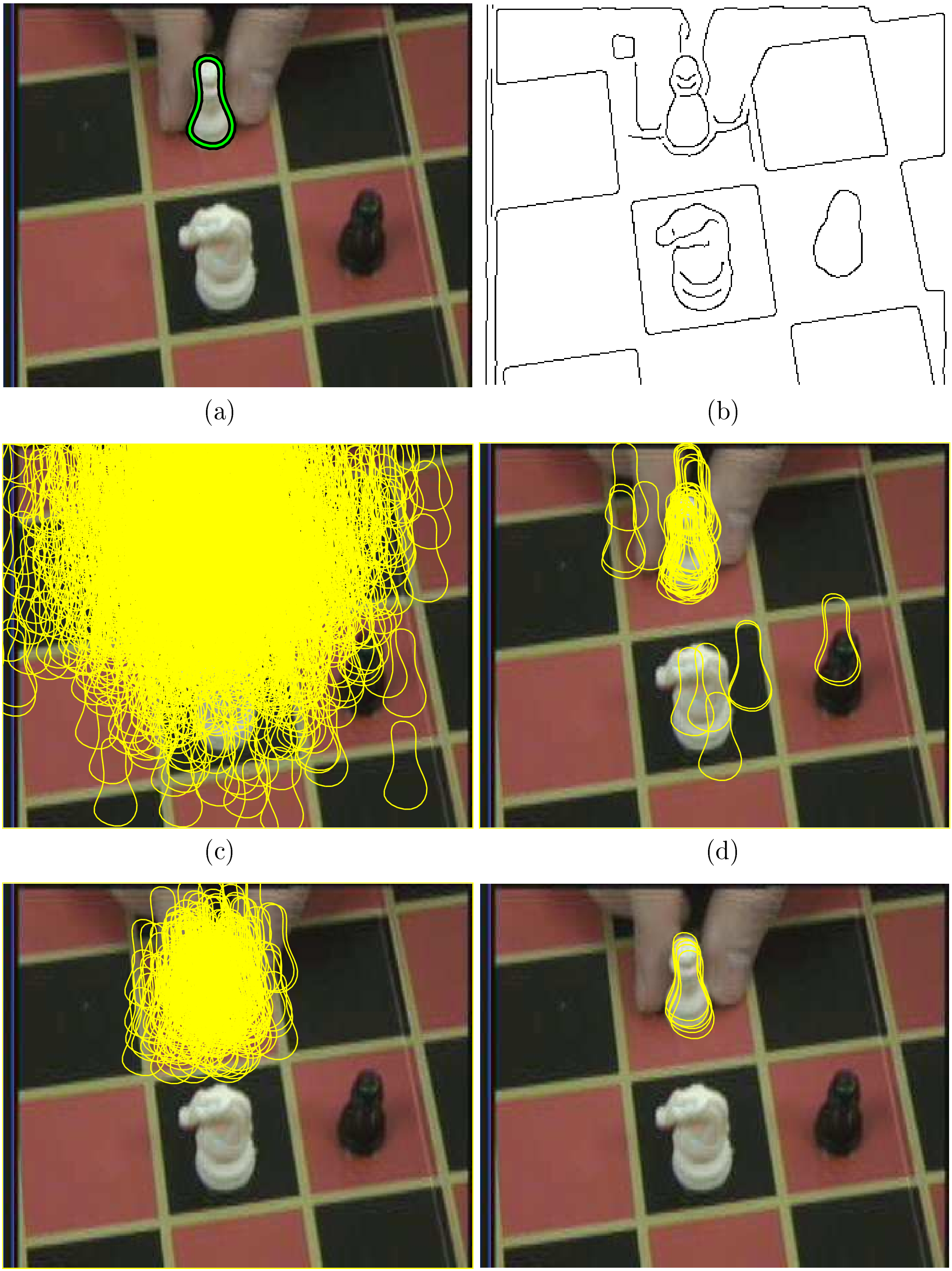


Figure 4.5: Random sampling for textured region measurement generation. (a) Predicted state; (b) Samples with large covariance (about predicted state); (c) Best samples from large covariance; (d) Samples with small covariance; (e) Best samples from small covariance.



(a) (b) (c) (d) (e) (f)

Figure 4.6: Random sampling for snake measurement generation. (a) Predicted state; (b) Canny edges in image ($\sigma = 2.0$, $\hat{\tau}_{low} = 0.25$, $\hat{\tau}_{high} = 0.75$); (c) Samples with large covariance (about predicted state); (d) Measurements for large covariance sample; (e) Samples with small covariance; (f) Measurements for small covariance sample.

$\left(\begin{smallmatrix}400 & 0\\ 0 & 400\end{smallmatrix}\right)$. The number of samples $N = 250$ and measurements $n = 5$ are also smaller, eliminating the distractions of the gap between the chess player's fingers and the other pieces.

4.2 Filtering Methods

Kalman filtering [5, 69] is an efficient method for tracking when the distribution on measurements is Gaussian. This filter is an algorithm for linear estimation of a set of time-varying parameters typically called the state $\mathbf{X}$. Suppose that the evolution of the system at time t is described by the *dynamic equation*:

$$\mathbf{X}_t = \mathbf{F}\,\mathbf{X}_{t-1} + \mathbf{q}_t \tag{4.1}$$

where $\mathbf{q}_t$ is a sequence of zero-mean, white, Gaussian noise with dynamic covariance $\mathbf{Q}$. The state is related to observable data $\mathbf{Z}$ by the *measurement equation*:

$$\mathbf{Z}_t = \mathbf{H}\,\mathbf{X}_t + \mathbf{r}_t \tag{4.2}$$

where $\mathbf{r}_t$ is also zero-mean, white, Gaussian noise with measurement covariance $\mathbf{R}$. Using the previous (or initial) state estimate and the current data, the Kalman filter arrives a new estimate for $\mathbf{X}$ by calculating the quantities in Table 4.1 (except for the identity matrix $\mathbf{Id}$, every variable not subscripted by $t-1$ is implicitly subscripted by time t).

A common modification to the plain Kalman filter to handle nonlinearities in the dynamic and measurement equations is the first-order Extended Kalman Filter (EKF) [5]. A nonlinear dynamic equation $\mathbf{X}_t = F(\mathbf{X}_{t-1}) + \mathbf{q}_t$ is linearized by assigning the first term of the Taylor series expansion of F about $\mathbf{X}_{t-1}$ at each filter update

$\widehat{\mathbf{X}} = \mathbf{F}\,\mathbf{X}_{t-1}$	Predicted state
$\hat{\mathbf{Z}} = \mathbf{H}\,\widehat{\mathbf{X}}$	Predicted measurement
$\widehat{\mathbf{P}} = \mathbf{F}\,\mathbf{P}_{t-1}\,\mathbf{F}^T + \mathbf{Q}$	State prediction covariance
$\mathbf{S} = \mathbf{H}\,\widehat{\mathbf{P}}\,\mathbf{H}^T + \mathbf{R}$	Measurement prediction covariance
$\boldsymbol{\nu} = \mathbf{Z} - \hat{\mathbf{Z}}$	Innovation
$\mathbf{W} = \widehat{\mathbf{P}}\,\mathbf{H}^T\,\mathbf{S}^{-1}$	Filter gain
$\mathbf{X} = \widehat{\mathbf{X}} + \mathbf{W}\,\boldsymbol{\nu}$	State estimate
$\mathbf{P} = (\mathbf{Id} - \mathbf{W}\,\mathbf{H})\,\widehat{\mathbf{P}}$	State covariance estimate

Table 4.1: Kalman filter equations

to the matrix $\mathbf{F}$. A nonlinear measurement equation $\mathbf{z}_t = H(\mathbf{X}_t) + \mathbf{r}_t$ is dealt with similarly by expanding H about $\widehat{\mathbf{X}}$ every filter update to obtain $\mathbf{H}$.

Situations in which there are departures from the assumption that the posterior is Gaussian require extensions to the Kalman filter. For example, noise might temporarily create multiple measurements or cause the target-originated measurement to disappear. Or we might be tracking T objects as independent entities, and thus expect there to be a persistent measurement for each one. Proper target-measurement correspondences are maintained by continually computing the *association probabilities* of the various possibilities.

Modifications to the Kalman filter to deal with these phenomena are often called data association methods [5]. Such methods are ways of rationally modeling part of the world outside of the state in order to estimate $\mathbf{X}$ more accurately. In the course of visual tracking, occlusions, distractions, and multiple targets of interest are common complications, so it seems natural to adapt data association techniques to combat them.

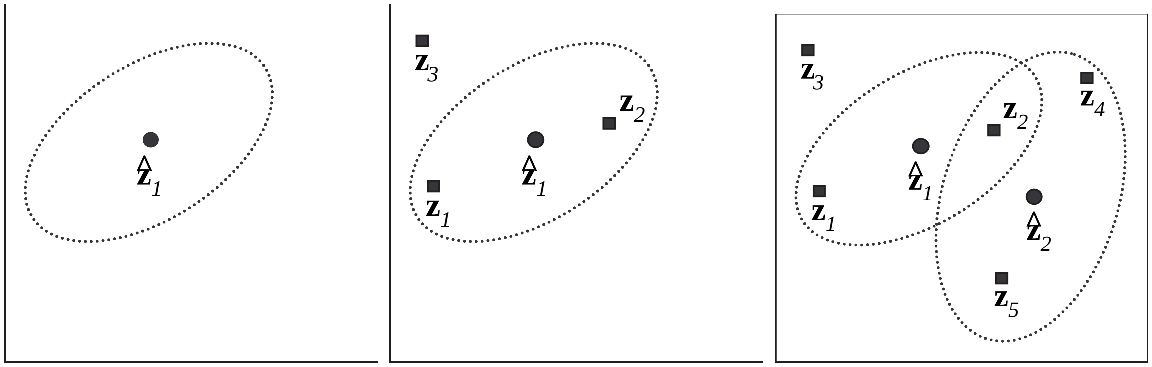
Next, we examine an extension of the Kalman filter to a basic data association filter, the Probabilistic Data Association Filter (PDAF) [5]. The three key steps of a vision-based PDAF tracking algorithm are shown in Figure 4.1. These are: (1) generate measurements, (2) weight them by association probability, and (3) update the state estimate based on the weighted measurements.

4.2.1 Probabilistic Data Association Filter

The measurement processes described above derive a group of candidate states for each tracker. The Probabilistic Data Association Filter (PDAF) [5, 28] is an extension of the Kalman filter [5] that uses a Bayesian approach to the problem of data association, or how to update the state when there is a single target and possibly no measurements or multiple measurements due to noise.

The simplest form of data association is the nearest neighbor (NN) method [5, 28], which picks the data closest to what is expected in order to update the state. With some finite probability, though, this choice will be wrong, leading to biases in the state estimate or outright mistracking. The PDAF, on the other hand, attempts to hedge its bets by weighting the various possibilities. Weights are assigned to the measurements based on two major assumptions. First, the PDAF assumes that there is exactly one target, giving rise to one "true" measurement, which may sporadically disappear either because the target is temporarily occluded or because of suboptimal feature detection at any stage of the pipeline between the camera and (for example) the edge detection algorithm. Second, the PDAF assumes that all other measurements are "false" and arise from a uniform noise process.

The relevant step in the Kalman filter is the computation of the innovation $\boldsymbol{\nu}$. The PDAF introduces a notion of the *combined* innovation, computed over the n measurements detected at a given time step as the weighted sum of the individual innovations: $\boldsymbol{\nu} = \sum_{i=1}^{n} \beta_i \boldsymbol{\nu}_i$. Each β_i is the probability of the *association event* Θ_i that the ith measurement is target-originated. Also computed is β_0, the probability of the event that none of the measurements is target-originated (i.e., the target is associated with the null measurement). These events encompass all possible inter-



(a) (b) (c)

Figure 4.7: The data association problem. (a) No measurements (ellipse indicates validation gate); (b) Multiple measurements; (c) Multiple targets.

pretations of the data, so $\sum_{i=0}^{n} \beta_i = 1$. The association probabilities are as follows for a given time step:

$$\beta_0 = \frac{b}{b + \sum_{j=1}^{n} e_j} \tag{4.3}$$

$$\beta_i = \frac{e_i}{b + \sum_{j=1}^{n} e_j}, \; i > 0 \tag{4.4}$$

where $e_i = \exp(-\frac{1}{2}\boldsymbol{\nu}_i^T \mathbf{S}^{-1} \boldsymbol{\nu}_i)$ and $b = \varphi(1 - P_D P_G)/P_D$. Here P_G is the probability that the correct measurement is in the *validation gate* (explained below), P_D is the probability that the measurement will be detectable if it is in the validation gate, and φ is a variable dependent on the number and dimensionality of the measurements [5]. For the tracking examples throughout this dissertation we will use values of $P_G = 0.99$ and $P_D = 0.9$.

The validation gate is an ellipsoidal volume in measurement space centered on $\hat{\mathbf{Z}}$, the measurement predicted from the current state estimate, and whose shape is defined by $\mathbf{S}$, the estimated covariance of the predicted measurement, such that the probability of a target-originated measurement appearing outside of it is negligible.

The size of the validation gate is set so that it contains $\sigma \geq 3$ standard deviations of the Gaussian distribution corresponding to $\mathbf{S}$, making the probability of such an event less than or equal to 0.01. Little accuracy is thus lost by disregarding measurements falling outside the gate.

Limiting image processing to a tracking window, or small rectangular subimage around the current target state [52], is a common approximation of a validation gate on the image spatial component of the state. Here we implement a validation gate through the sampling covariance $\mathbf{\Sigma}_{\mathcal{X}}$.

Randomly sampling from a normal distribution and selecting the top fraction of the samples as measurements, as we do, does not precisely satisfy the PDAF assumption of a uniform distribution of false measurements, but it is usually a reasonable approximation. Multiple measurements coming from the true, target-originated peak in the image likelihood function $p(\mathbf{I} \mid \mathbf{X})$, as seen in Figure 4.3(c) and (e) and Figure 4.6(d) and (f), are tightly clustered in one part of the validation gate rather than uniformly distributed throughout it. This is a harmless departure from the uniformity assumption because the effect of the PDAF association probabilities is to average the contribution of the measurements to the state estimate, and measurements tightly arranged around a maximum of $p(\mathbf{I} \mid \mathbf{X})$ average out to that maximum.

Measurements from false, non-target-originated peaks in $p(\mathbf{I} \mid \mathbf{X})$ are a different matter. Many false peaks are truly due to noise sources such as the atmosphere (a factor that is more important outdoors and over long distances), camera, and video capture device. It is also reasonable to model some dynamic scene elements as noise because of their unpredictable movements and ability to be appear and disappear, such as reflections off of a rippling water surface. However, many scene elements—other parts of a complex object being tracked, a static background, other moving objects, etc.—are too persistent to be regarded in this fashion. If peaks in $p(\mathbf{I} \mid \mathbf{X})$

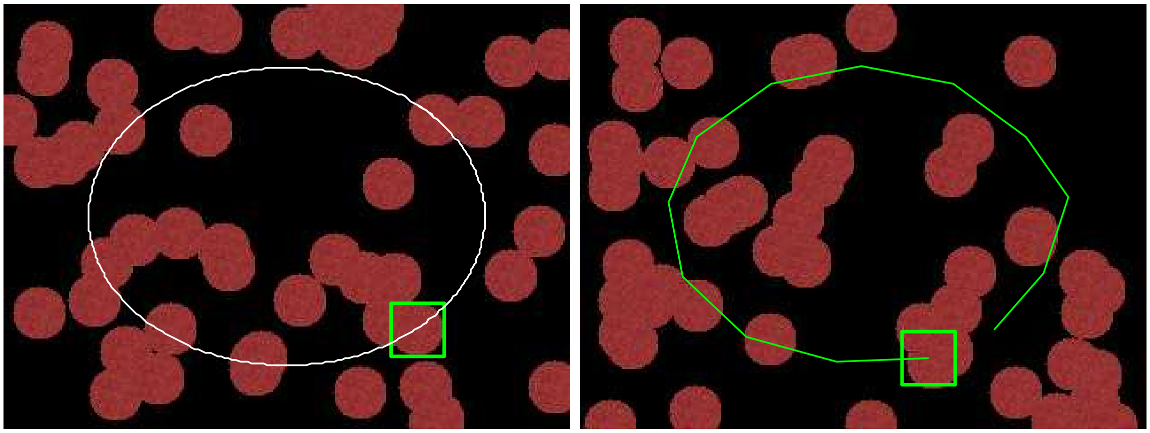
corresponding to such visual phenomena are weak compared to the target-originated peak and uniformly distributed, random sampling will generate measurements from these peaks relatively uniformly and their influence will be cancelled out. If the false peaks are strong enough, though, measurements from them will be generated disproportionately, biasing the PDAF filter's state estimates. We investigate techniques for successfully tracking when there are multiple persistent peaks in $p(\mathbf{I} \mid \mathbf{X})$ in the next chapter.

As a final note in this discussion of the PDAF, the introduction of association probabilities alters the calculation of the error covariance of the state estimate $\mathbf{P}$ given in Table 4.1. For the standard Kalman filter, $\mathbf{P}$ is independent of the measurements, but the uncertainty of the state estimate with the PDAF filter is highly dependent on the data. Specifically, $\mathbf{P} = \beta_0 \widehat{\mathbf{P}} + (1 - \beta_0) \mathbf{P}^c + \widetilde{\mathbf{P}}$, where $\mathbf{P}^c = (\mathbf{Id} - \mathbf{W}\,\mathbf{H}) \widehat{\mathbf{P}}$ and $\widetilde{\mathbf{P}} = \mathbf{W} \left[\sum_{i=1}^{n} \beta_i \boldsymbol{\nu}_i \boldsymbol{\nu}_i^T - \boldsymbol{\nu} \boldsymbol{\nu}^T \right] \mathbf{W}^T$ (see [5] for a derivation).

4.3 Results

Some examples of PDAF tracking are given below in Figures 4.8, 4.10, and 4.11 for homogeneous regions, Figures 4.12 for snakes, and Figures 4.9 and 4.13 for textured regions.

Figure 4.8 illustrates the resistance of the PDAF to distractions due to noise. To perfectly control noise conditions, we created a computer graphics simulation of red circles on a black background. There is one target which moves in a counterclockwise elliptical orbit at a rate of 0.02 radians per frame and 50 random distractors from a uniform distribution in each frame. The target has a state of $\mathbf{X} = (x, y)$ and the same measurement parameters; it is tracked with a homogeneous region tracker. 100 samples are chosen with a sampling covariance of $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 100 & 0 \\ 0 & 100 \end{smallmatrix}\right)$. In one series of



(a) (b)

Figure 4.8: PDAF: Tracking a synthetic, circular homogeneous region with uniform noise (CG). (a) Frame 0 with the initial position and ground truth for the entire orbit overlaid; (b) Frame 300 with a history of estimates at 25 frame intervals.

experiments, only the single best sample was used as a measurement. The tracker was able to follow the circle through a full orbit in only 5 out of 20 trials. In another series of experiments, the 10 best samples were used as measurements. This tracker was much less vulnerable to distraction, and succeeded in tracking the circle through a full orbit in 17 out of 20 trials. Two representative frames from a trial of the latter series are shown in Figure 4.8. The actual path followed by the circle is shown in Figure 4.8(a) overlaid on the initial frame, and the estimates made by the tracker up to frame 300 are drawn in Figure 4.8(b) at 25 frame intervals.

In Figure 4.9, a textured region tracker is attached to a mouse embryo as the microscope slide is moved and the embryo is poked and rotated with a probe. The state of the tracker is simply position and orientation: $\mathbf{X} = (x, y, \phi)$, and measurement space is $\mathcal{Z} = X \times Y \times \Phi$. As the figure shows, a tracker that uses gradient ascent (Powell's method) alone to generate a single measurement is thrown off when the embryo moves abruptly after frame 60. A tracker that uses random sampling for measurement generation, on the other hand, is able to recover from these agile motions. In this case, 5 measurements are culled from 250 samples, where $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0.04 \end{smallmatrix}\right)$.

A homogeneous region tracker would be inappropriate because of the lack of contrasting color in the image, and a snake tracker around the contour of the embryo would be able to estimate position but not rotation.

Figure 4.10 shows a homogeneous region tracker following the forearm of a person as he walks from left to right. There is not much scaling, but the motion is fairly dynamic, so the state includes the forearm's image position, orientation, and the velocities of these parameters: $\mathbf{X} = (x, y, \phi, \dot{x}, \dot{y}, \dot{\phi})$. Each measurement is a translation and rotation of a fixed size rectangle, so $\mathcal{Z} = X \times Y \times \Phi$. The rectangles overlaid on the figure indicate the 10 best measurements taken from 1000 samples in that frame, where $\mathbf{\Sigma}_{\mathcal{X}} = \begin{pmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0.02 \end{pmatrix}$. In general, the most probable measurements form a tight cluster around the measurement predicted from the current state (except where distractions induce a multimodal distribution).

Another example of homogeneous region tracking is shown in Figure 4.11. Here the target is the orange front end of a race car approaching the camera along a banked oval, so the state includes image position, orientation, and scale: $\mathbf{X} = (x, y, \phi, s)$. Measurement space is $\mathcal{Z} = X \times Y \times \Phi \times S$. The tracker state at 40-frame intervals is overlaid on the figure. For this example there are 5 measurements and 1000 samples, and $\mathbf{\Sigma}_{\mathcal{X}} = \begin{pmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.005 \end{pmatrix}$.

In Figure 4.12, we track two human heads in infrared (IR) imagery as they primarily translate and scale, one with a closed contour and the other with an open curve. Ordinarily, an affine B-spline has six degrees of freedom, but we have found that shearing and independent scaling of the vertical and horizontal axes are rare for many simple object motions, so for this example the aspect ratios of the snakes are constrained to stay constant, resulting in three degrees of freedom. This heuristic is an approximation of what is learnable about appropriate priors on snake dynamics by more sophisticated procedures [17, 18]. Thus, the state of each tracker is expressed

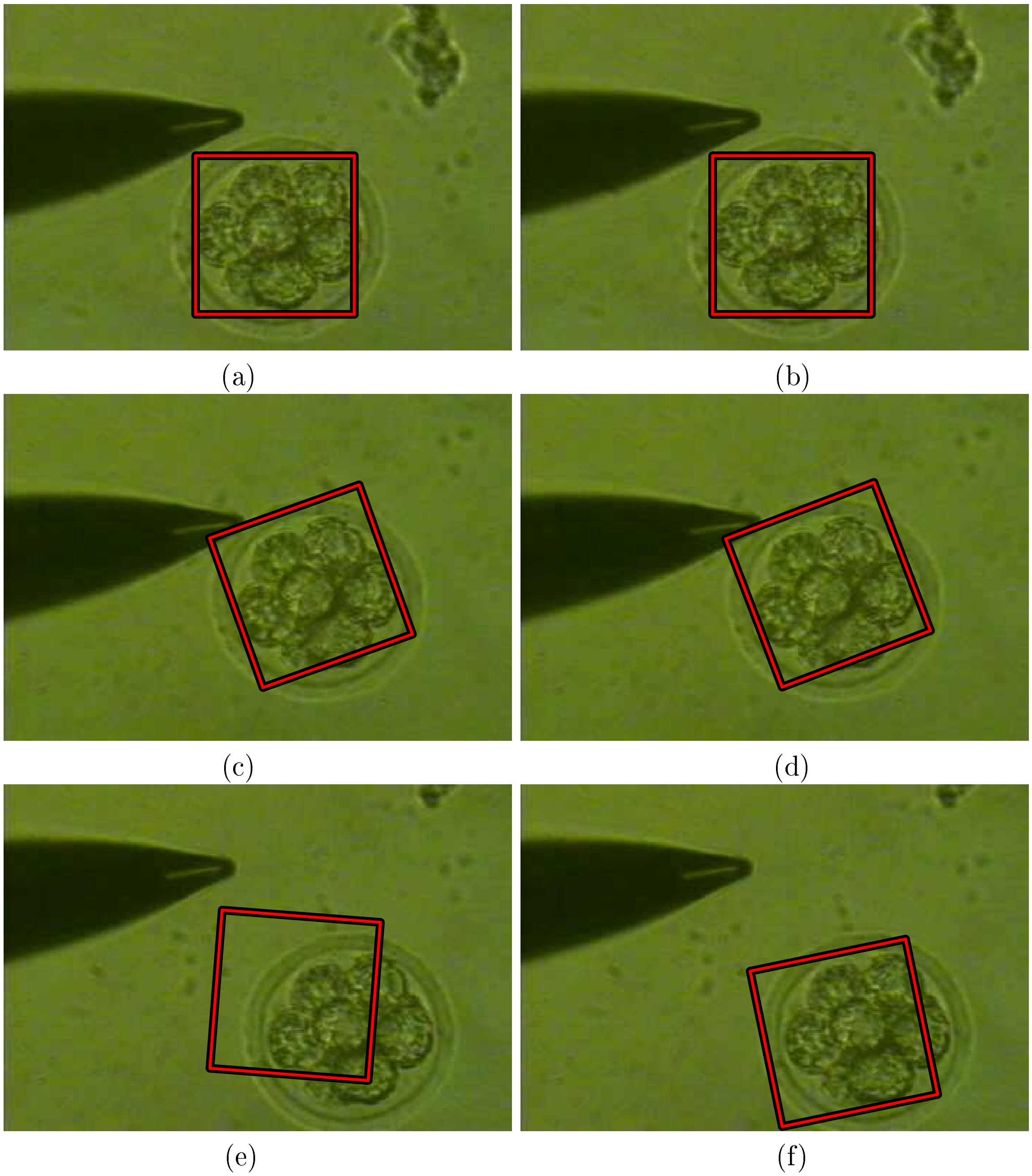
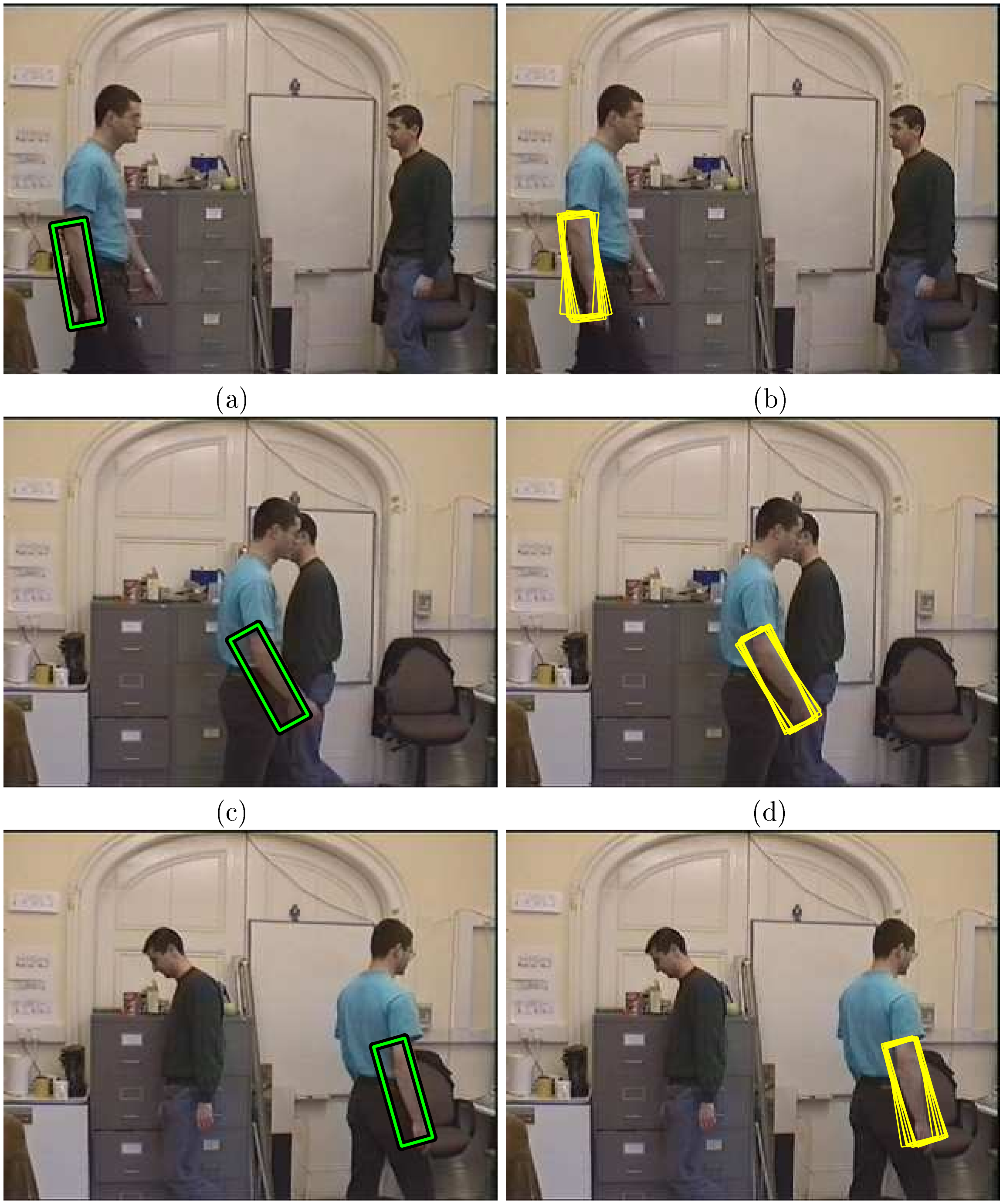
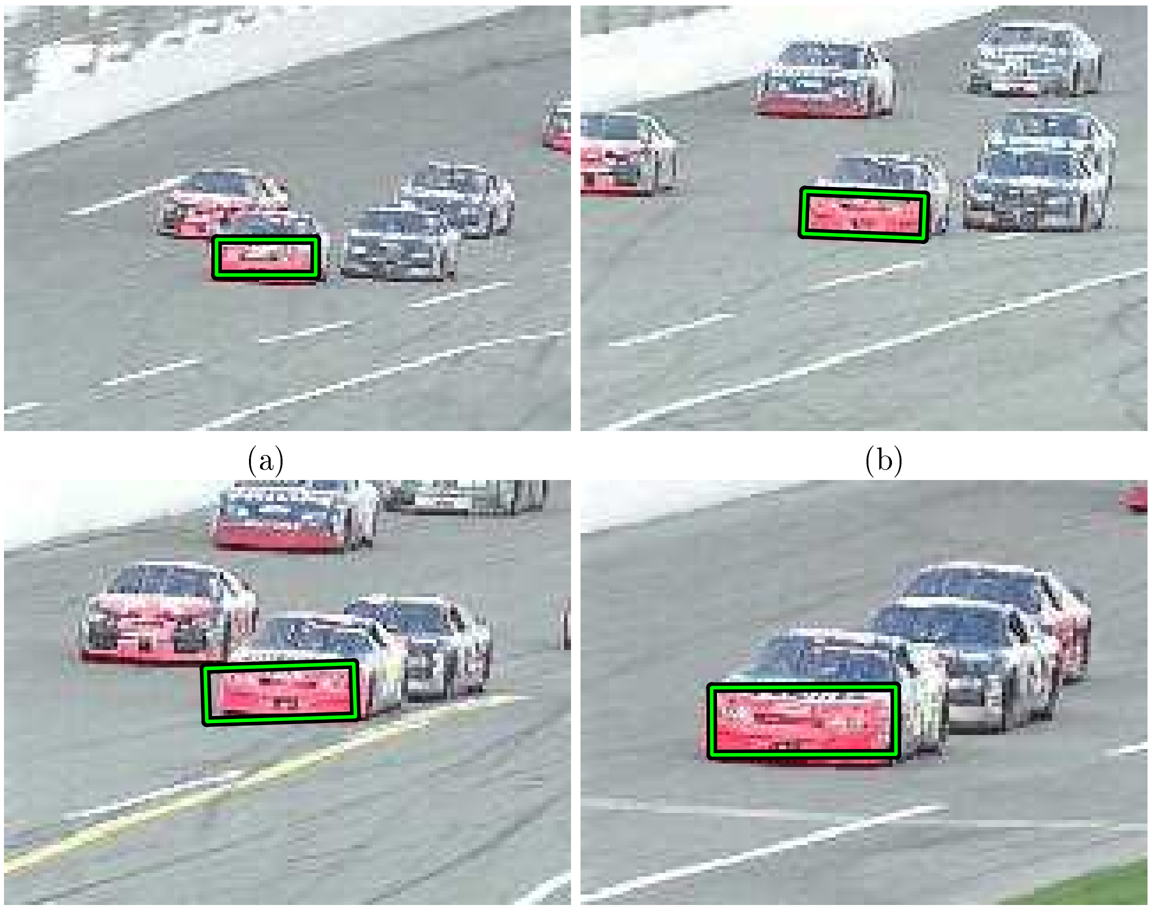


Figure 4.9: Gradient ascent (GA) alone vs. random sampling (RS): tracking a mouse embryo with a translating, rotating textured region (MPEG). (a) GA state in frame 0; (b) RS state in frame 0; (c) GA frame 60; (d) RS frame 60; (e) GA frame 120; (f) RS frame 120. (Sequence courtesy of G. Danuser).



(a) (b) (c) (d) (e) (f)

Figure 4.10: PDAF: Tracking a swinging arm with a translating, rotating homogeneous region (MPEG). (a) Region state in frame 0; (b) Region measurements in frame 0; (c) State in frame 17; (d) Measurements in frame 17; (e) State in frame 34; (f) Measurements in frame 34. (Sequence courtesy of J. MacCormick).



(a) (b) (c) (d)

Figure 4.11: PDAF: Tracking a race car with a translating, scaling, rotating homogeneous region (MPEG). (a) Region state in frame 0; (b) State in frame 40; (c) State in frame 80; (d) State in frame 120.

as $\mathbf{X} = (x, y, s)$ and $\mathcal{Z} = X \times Y \times S$. Each tracker selects the best 5 measurements from 250 samples; $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0.01 \end{smallmatrix}\right)$. The good contrast provided by the IR means that there are not many distracting edges, but the sequence shows the ability of the PDAF/state-sampling method of snake tracking to estimate the state accurately.

A scaling, translating textured region tracker follows a jumping motorcyclist in Figure 4.13. The tracker state is given by $\mathbf{X} = (x, y, s)$, and measurement space is $\mathcal{Z} = X \times Y \times S$. Five measurements are taken from 1000 samples, where $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 0.001 \end{smallmatrix}\right)$. Some biasing is introduced because the perspective on the motorcycle changes slightly from frontal to a three-quarters view, an unmodeled transformation.

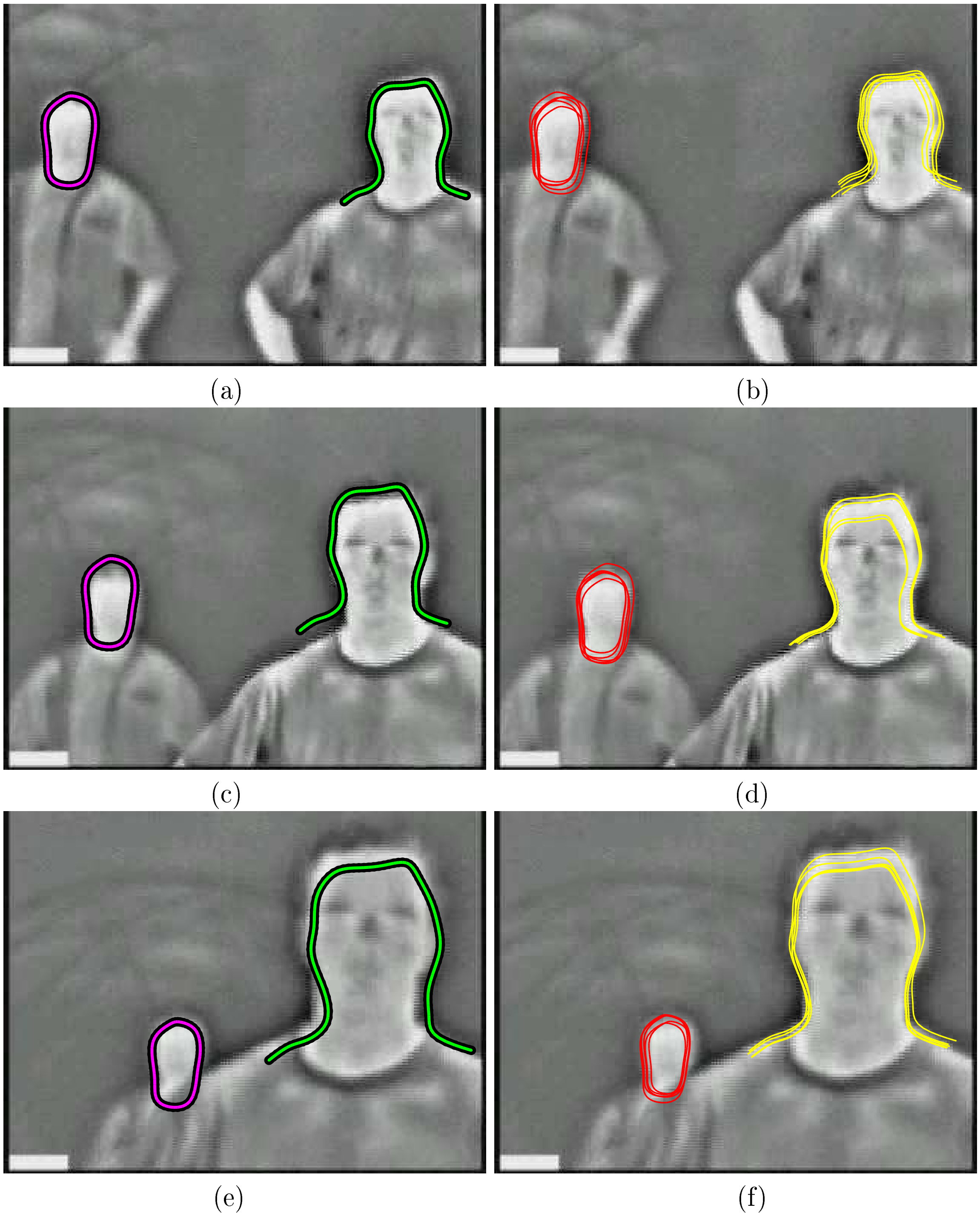
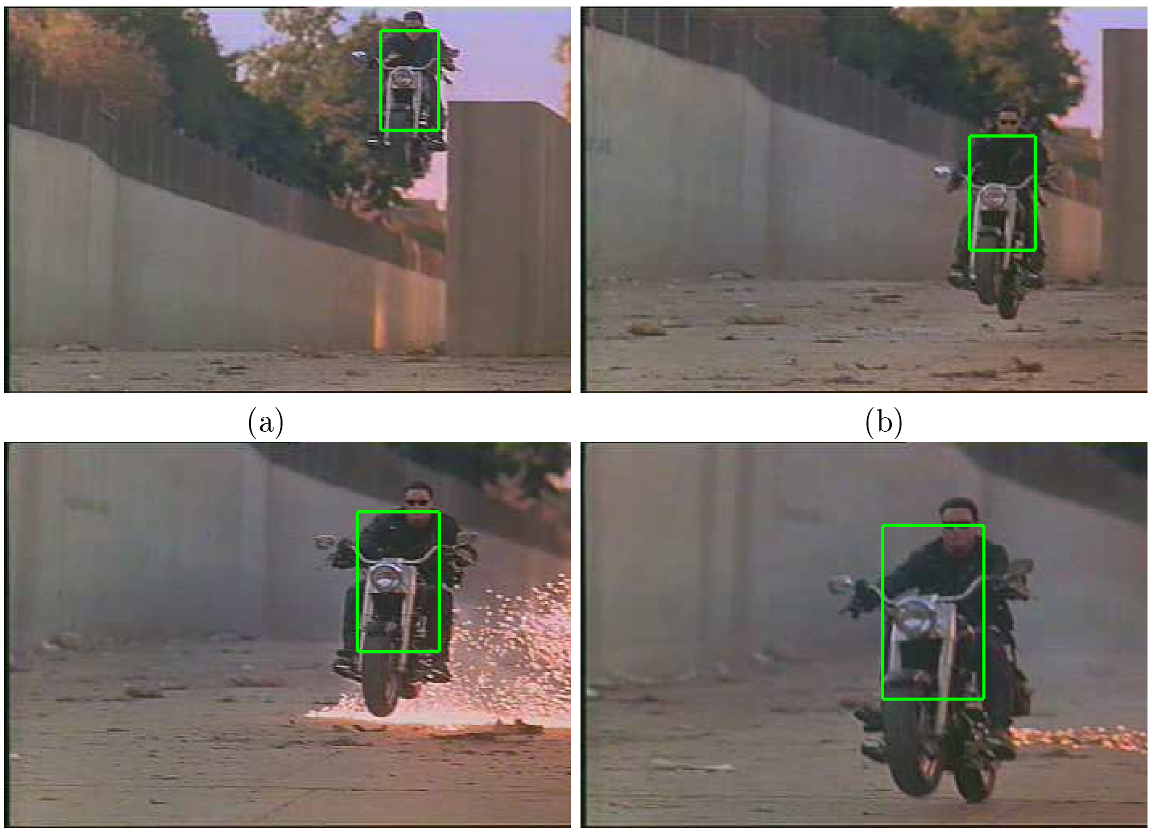


Figure 4.12: PDAF: Tracking two faces with translating, scaling snakes (MPEG of infrared imagery, Canny). (a) Snake states in frame 0; (b) Snake measurements in frame 0; (c) States in frame 70; (d) Measurements in frame 70; (e) States in frame 140; (f) Measurements in frame 140.



(a) (b) (c) (d)

Figure 4.13: PDAF: Tracking a motorcycle with a translating, scaling textured region (MPEG). (a) Region state in frame 0; (b) State in frame 30; (c) State in frame 60; (d) State in frame 90.

Chapter 5

Joint Tracking

In the previous chapter we assumed that there are no other strong, persistent features in the image that have attributes similar to those of the tracked target. This is a fair approximation for many visual situations, but it certainly does not hold when tracking multiple similar or identical objects or one object with multiple similar parts. If and when the states of the individual parts become proximate, one target-originated measurement may often fall within another target's overlapping validation gate. Such persistent interference, were one to simply run a separate PDAF tracker on each part, could lead to multiple trackers locked onto the same part. Even when using a pure gradient ascent technique for tracking, which greatly reduces the size of each tracker's validation gate and hence its susceptibility to distraction, two similar targets crossing paths may cause confusion.

An example of a visual situation that may lead to mistracking because it contains multiple similar features is shown in Figure 5.1. This image was captured from a video of two tandem kayaks being paddled alongside one another. One might want to track the boats as a whole, the torsos of the paddlers in them, or even the tips of their oars. Each of these objects can be readily characterized by color—the boats

Figure 5.1: Ambiguity when tracking multiple objects simultaneously (captured from a Quicktime video)

are red, two of the people are wearing yellow life jackets, and the blades of the oars are bright blue—and thus would seem to be amenable to the techniques introduced in the previous chapter. The problem is that multiple areas of the image satisfy each of these descriptions. That is, searching the image for an oar blade by looking for a homogeneous blue region will return multiple good candidates, or measurements. Many persistent measurements can lead to an ambiguity of association, often making a PDAF tracker peel off of the correct image feature and attach itself to a nearby feature that is similar but incorrect.

This phenomenon occurs because PDAF state estimation is essentially solving a weighted least squares problem, where the association probabilities are the weights and the measurements are the data. Two or more measurements that persist in the tracker's validation gate will drive its state to a position that minimizes the distance between the predicted measurement and the observed ones. Because of the randomness of the state dynamics and noise (i.e., whether the measurements are consistently refound from frame to frame), when image conditions again produce one persistent measurement in the validation gate, the tracker may have slipped onto a non-target feature. In particular, if there are multiple correctly-initialized PDAF trackers looking (for example) for blue regions, because of their ignorance of each other they might eventually all clump onto the same oar blade and erroneously ignore the others.

This chapter reviews methods for dealing with this class of problems by sharing information between trackers. If we are tracking all of the image features that may mutually cause distraction, it seems reasonable to surmise that adding an overarching layer of reasoning may help ensure that the trackers are efficiently and correctly distributed over the measurements. One such technique that we discuss is an existing extension to the PDAF called the Joint Probabilistic Data Association Filter (JPDAF) [5]. The JPDAF often mitigates problems of persistent distraction that occur when tracking multiple objects. The first part of the chapter investigates the issues involved in adapting the JPDAF to vision; one limitation is that it can only be used for groups of objects of the same modality. In the second half of the chapter we introduce a new approach called the Joint Likelihood Filter (JLF). This method captures the crux of the JPDAF but is readily applicable to mixtures of different tracking modalities, is more efficient than the JPDAF, and incorporates explicit reasoning about occlusion relationships between objects.

5.1 Joint Probabilistic Data Association Filter

The Joint Probabilistic Data association Filter (JPDAF) [5] deals with the problem of interference between multiple trackers by sharing information among separate PDAF filters in order to more accurately calculate association probabilities. This joint calculation of association probabilities for multiple objects is illustrated in Figure 5.2. The essential effect of the JPDAF is an exclusion principle of sorts that prevents two or more trackers from latching onto the same target.

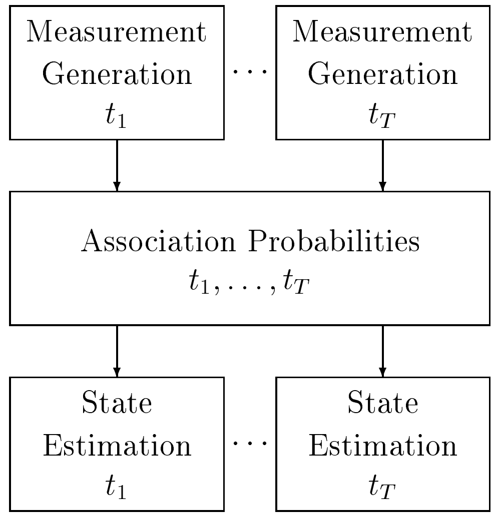


Figure 5.2: JPDAF pipeline

Suppose that we are tracking T objects, for which a total of n measurements have been generated from the current image. These measurements may be generated independently for each tracked object as in the previous chapter, but this leads to difficulties as we will see below. Methods for deriving measurements for all objects *jointly* are presented in Section 5.1.1. For simplicity, we assume for the moment that every target is being tracked using the same tracking modality (in the sense of Chapter 3). This means that every tracker shares the same image likelihood function $p(\mathbf{I} \,|\, \mathbf{X})$. If this is not the case—if targets have different modalities—then the JPDAF is not applicable. An alternative method that accommodates different modalities is discussed in the next section.

A key notion in the JPDAF is that of a *joint event* $\mathbf{\Theta}$, or conjunction of association events Θ_{jt_j}. The subscript t_j has been added to the definition of association event introduced in the last chapter in order to indicate the index of the target to which measurement j is matched. For the PDAF there was only one target, making this unnecessary. A particular joint event is thus defined over T targets and n measurements as $\mathbf{\Theta} = \bigwedge_{j=1}^{n} \Theta_{jt_j}$. The event Θ_{j0} indicates that measurement j

is associated with no target—that is, it is assumed to be due to noise—while Θ_{0t} amounts to a hypothesis that target t is occluded or simply undetected.

A useful concept to consider at this point is that of a *feasible* joint event [5]. The probability of a given joint event $\mathbf{\Theta}$ depends, as with the PDAF, on the distances between each target's predicted measurement and the actual measurement it is associated with in $\mathbf{\Theta}$, as well as the associated measurements' image likelihoods. Thus, one condition for $\mathbf{\Theta}$'s feasibility is that for every association Θ_{jt_j} in $\mathbf{\Theta}$, measurement j is in the validation gate of target t_j. However, an additional influence on the probability of $\mathbf{\Theta}$ stems from the interaction of the various association events in $\mathbf{\Theta}$. Suppose that a single, unified measurement process generates at most one measurement for each peak in the image likelihood function $p(\mathbf{I}\,|\,\mathbf{X})$ and that each target induces at most one peak in $p(\mathbf{I}\,|\,\mathbf{X})$. According to these conditions, two kinds of combinations of associations are logically incompatible or *infeasible*. In the first case, a joint event $\mathbf{\Theta}$ contains two associations $\Theta_{jt_1}, \Theta_{jt_2}$ such that $t_1 \neq t_2$ and $j \neq 0$, implying that two different targets are responsible for the same measurement. This is a contradiction. In the second case, $\mathbf{\Theta}$ includes associations $\Theta_{it_i}, \Theta_{jt_j}$ such that $i \neq j$ but $t_i = t_j$. This amounts to an interpretation that a single target has spawned multiple measurements—also an impossibility.

Infeasible joint events thus have zero probability and can be disregarded. A joint event is feasible only when each target is associated with one or no measurements and each measurement is associated with one or no targets. To denote these constraints mathematically, we first define a *measurement association indicator* τ and *target detection indicator* δ [5]. τ_j is defined to be the number of targets associated with measurement j ($\tau_j = 0$ indicates a hypothesis that the measurement is due to noise), while δ_t is the number of measurements matched to target t. A joint event is thus feasible when $\tau_j \leq 1$ for all measurements j and $\delta_t \leq 1$ for all targets t. The process of generating all feasible joint events for an example set of targets and measurements is illustrated in Figure 5.3.

Without some form of the preceding feasibility logic in the state estimation pipeline, it is possible for nearby targets t_1 and t_2 to consistently be associated with one measurement which properly belongs to t_1. Over time, such a condition will lead to their states inappropriately converging, and t_2 will have lost track of its correct measurement.

Our assertion that each object causes a single peak in $p(\mathbf{I} \mid \mathbf{X})$ (or none if it is fully occluded) is not always strictly true. Reflective surfaces around a target can cause multiple copies to be perceived by the viewer; sharp shadows cast by the target can project slightly-transformed copies of its silhouette nearby. We assume that these imaging phenomena can be neglected as improbable for the bulk of visual situations. Also, the condition that a single measurement be created for each peak is not automatically met. Using random sampling alone, the method of measurement generation described in the previous chapter typically extracts multiple measurements not attributable to noise for each target, violating one of the presumptions of the JPDAF. Even if this problem is remedied by simply limiting the number of measurements created by each tracker to 1, each of these measurements may still be derived from the same image feature. As we will see, gradient ascent in fact becomes a necessary ingredient in the measurement process for the JPDAF. In section 5.1.1 we address these issues by introducing a single, joint measurement process over all targets that apportions measurements rationally.

For now, though, we assume that the pool of n measurements derived from the T targets contains no more than T measurements that are due to a target-derived image feature as opposed to noise, and that all of the target-derived measurements are due to unique targets instead of the same one. Next we will describe how the JPDAF

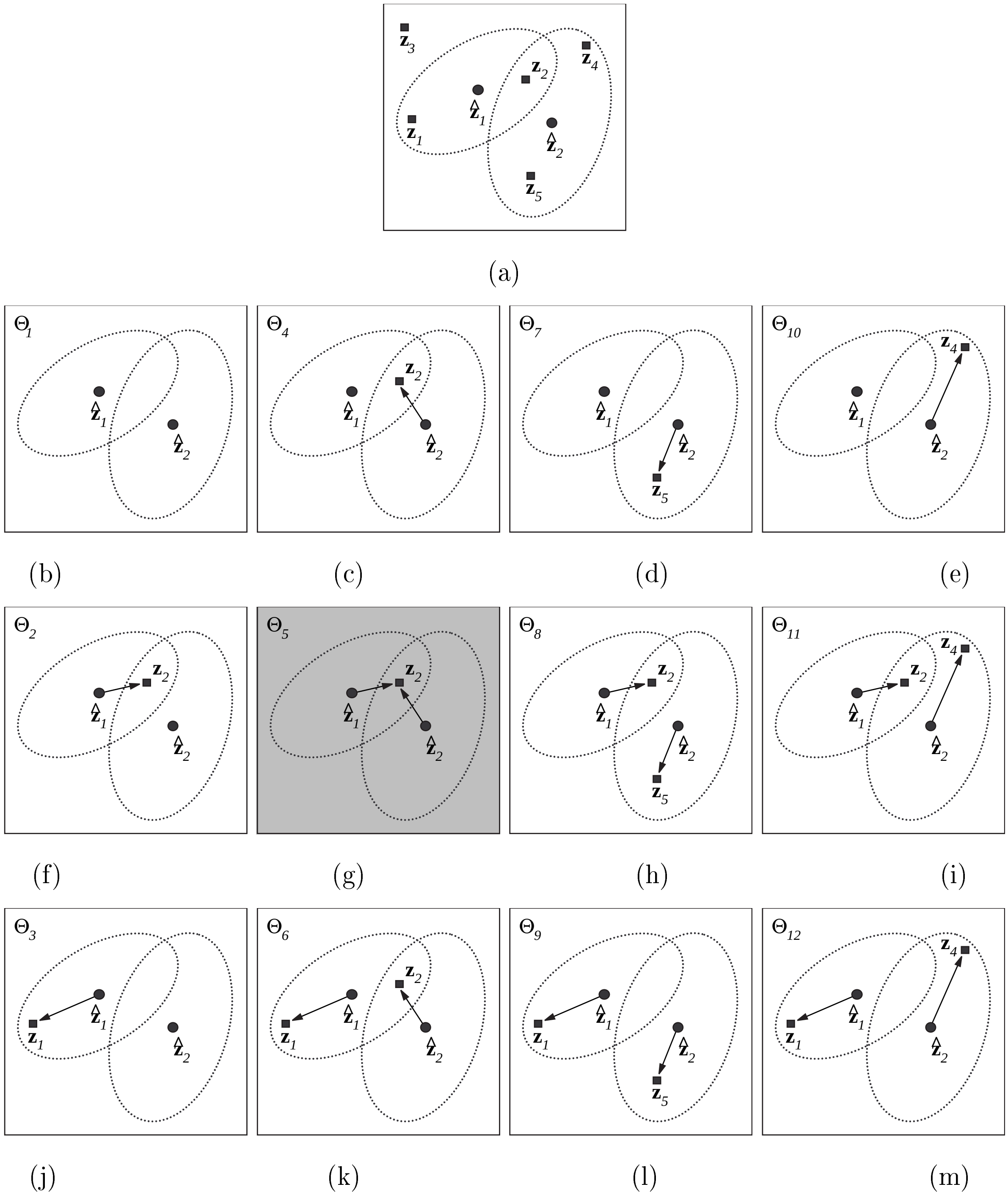


Figure 5.3: Joint events. (a) Targets and measurements; (b-m) Joint events generated. Joint event $\boldsymbol{\Theta}_5$ in (g) is infeasible. Any joint event including an association Θ_{jt_j} for which measurement j is outside t_j's validation gate is automatically disregarded and thus not included in the enumeration.

modifies the association probabilities computed for each tracker from the standard established for the PDAF in order to account for the additional information available from the other trackers.

Let $\omega_{jt}(\mathbf{\Theta}) = 1$ if $\Theta_{jt} \subset \mathbf{\Theta}$ and 0 otherwise. From [5], the probability of association between measurement j and target t given measurements $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$ is given by $\beta_{jt} = \sum_{\mathbf{\Theta}} P(\mathbf{\Theta} \,|\, \mathbf{Z}_1, \ldots, \mathbf{Z}_n)\omega_{jt}(\mathbf{\Theta})$, where:

$$P(\mathbf{\Theta} \,|\, \mathbf{Z}_1, \ldots, \mathbf{Z}_n) = \kappa \prod_{j=1}^{n} [N_j]^{\tau_j} \prod_{t=1}^{T} (P_D^t)^{\delta_t} (1 - P_D^t)^{1-\delta_t} \tag{5.1}$$

where κ contains terms for normalization and scaling, and N_j is the Gaussian probability density $N[\mathbf{Z}_j; \hat{\mathbf{Z}}_{t_j}, \mathbf{S}_{t_j}]$ for measurement j. Here $\mathbf{Z}_j$ is the measurement value, $\hat{\mathbf{Z}}_{t_j}$ is the predicted measurement value for target t_j, and $\mathbf{S}_{t_j}$ is the associated innovation covariance.

The above formula gives an estimate of the probability of each particular target-measurement association. However, for a given tracker t we cannot directly employ the set of JPDAF-computed association probabilities β_{jt} in the same manner that the association probabilities β_j would be used for the PDAF. This is because the PDAF linearly combines innovations, in effect averaging the contributions of the measurements in order to update the state. When, as with the PDAF, there is only one persistent, target-derived measurement plus some noise, the noise will cancel out over time because of its transient nature (if it is not too severe). If there are multiple persistent measurements, though, as with the JPDAF, averaging their contributions can lead to a state estimate such that the predicted measurement is the average of the persistent measurements (depending on the proximity of the measurements to the predicted measurement and the value of the measurement covariance $\mathbf{R}$). In other words, by preventing multiple trackers from clumping onto one feature the

JPDAF could cause them to all congregate around some hypothetical mean feature. This fact does not appear to have been noted in [5].

In order to avoid this phenomenon we reset the probability of one special association event to unity and the rest to zero for each target, and then update exactly as with the PDAF. This prevents any unwarranted combination of innovations. How shall the the favored association be selected for each tracker t? It is tempting to greedily choose the most likely association for each tracker t—the association event Θ_{jt} such that $\beta_{jt} \geq \beta_{it}$ for all $i \neq j$. This does not work, however, because of the possibility that the two targets t_1 and t_2 will be infeasibly associated with the same measurement—i.e., $\Theta_{j_1 t_1}$ and $\Theta_{j_2 t_2}$ are picked, but $j_1 = j_2$. (This happens immediately, for example, when there is one measurement $\mathbf{Z}_1$ equidistant from the predicted measurements $\hat{\mathbf{Z}}_1$ and $\hat{\mathbf{Z}}_2$ and the initial filter parameters as enumerated in Table 4.1 are identical for the two trackers). Instead, we use those associations of the most probable joint event $\hat{\boldsymbol{\Theta}}$ which are not of the form Θ_{j0}. These association events represent the most probable mutually feasible set available.

5.1.1 JPDAF Measurement Generation

One desirable characteristic of any approach to joint measurement generation is that only one measurement be created for each peak in $p(\mathbf{I} \mid \mathbf{X})$. Consistently finding multiple measurements within the same basin of attraction for a local maximum violates a key assumption of the JPDAF and can lead to multiple targets becoming associated with that peak by circumventing the JPDAF's feasibility constraint. Occasional deviations from the one measurement per peak ideal are acceptable but the method of the previous chapter, which results in a fixed number of measurements n for each target (typically, $n > 1$), must clearly be modified.

The joint method we use for T targets is based on the random sampling technique presented in the previous chapter. First, N_j samples $\mathbf{Z}_i$ are randomly generated for each tracker around $\hat{\mathbf{Z}}_j$, target t_j's predicted measurement. Based on their measurement image likelihoods $p(\mathbf{I} \,|\, \mathbf{Z}_i)$, some fraction of the less-likely samples is eliminated from further consideration.

In the second step, each remaining sample $\mathbf{Z}_i$ serves as the starting point for gradient ascent to a local maximum of the measurement image likelihood function $p(\mathbf{I} \,|\, \mathbf{X})$. We use the conjugate gradient algorithm [94] for hill-climbing, where the gradient of $p(\mathbf{I} \mid \mathbf{X})$ is estimated numerically as described in the previous chapter. The result of this step for each sample is $\mathbf{Z}'_i$.

The purpose of the hill-climbing step is twofold. First, the resulting samples $\mathbf{Z}'_i$ are, as a whole, better and less noisy (in the sense that they are more consistent from frame to frame) candidates for state estimation. Second, states that are on the slopes of the same peak of $p(\mathbf{I} \,|\, \mathbf{X})$ but somewhat separated by the randomness of the sampling process tend to converge in state space $\mathcal{X}$ as they ascend (provided certain local conditions on $p(\mathbf{I} \,|\, \mathbf{X})$ hold). If this is true, we can deduce that aggregations of samples after hill-climbing will be relatively tightly clustered around local maxima. Since the JPDAF relies on the measurement process to generate only one measurement per peak on average, we can choose the best sample in each cluster as representative of a peak.

The third step is therefore to try to choose one exemplar for each group of samples. This is done by enforcing a *minimum separation* between samples in $\mathcal{X}$. First, the hill-climbed samples are sorted by fitness. Starting with the most fit sample $\mathbf{X}_{best}$, all less fit samples $\mathbf{X}_i$ such that $|\mathbf{X}_{best} - \mathbf{X}_i| \leq \mathbf{\Delta}$ are eliminated. In practice, we use a different threshold $\mathbf{\Delta}_k$ for each parameter of the joint measurement and eliminate samples which are too close along any dimension. Unless otherwise noted, we use $\mathbf{\Delta}_X = \mathbf{\Delta}_Y = 10$ pixels, $\mathbf{\Delta}_\Phi = 0.1$ radians, and $\mathbf{\Delta}_S = 0.01$. The purpose of $\mathbf{\Delta}$ is to attempt to compensate for any lack of precision in the hill-climbing algorithm and to ignore maxima whose basins of attraction are too small.

The thinning process is repeated for the next most fit sample still remaining, and so on until the least fit sample left is reached. This is essentially a clustering process, but we do not need a general clustering algorithm because of the assumption that the gradient ascent step has brought the members of each cluster sufficiently close together. This process yields a variable number of measurements n generally equal to the number of tracked objects T. The value of n can vary due to the randomness of the sampling procedure and the degree to which the condition that the image only has T target-like features holds.

This method is applied in Figure 5.4 to the hypothetical posterior distribution and the Yankees picture from the previous chapter. Compare Figure 5.4(a) to Figure 4.2(d) and Figures 5.4(b) and (c) to Figures 4.5(c) and (e), respectively.

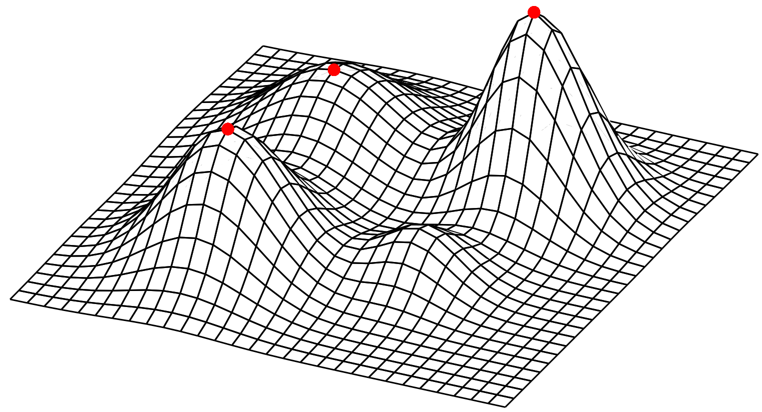
5.1.2 Algorithmic Complexity

For a large number of targets, the JPDAF can become combinatorially problematic. This is because the number of joint events $\mathbf{\Theta}$ which must be considered for each filter update is the following exponential function of the number of validated measurements n and targets T:

$$F(n,T) = 1 + \sum_{i=1}^{\min(n,T)} \binom{n}{i} \prod_{j=0}^{i-1} (T-j) \tag{5.2}$$

Example values of F for some different n and T are shown in Table 5.1.

Such complexity can often be avoided by partitioning the target set into groups of targets with overlapping validation gates and running independent JPDAFs on each one [28].



(a)

(b)

(c)

Figure 5.4: Gradient ascent and minimum separation. (a) Best samples for hypothetical posterior distribution after gradient ascent with Powell's method. Each maximum has multiple samples in almost the same location, so the minimum separation procedure results in 3 measurements; (b) Measurements resulting from gradient ascent and minimum separation on best samples of large covariance; (c) Measurement for small covariance.

$n\backslash T$	1	2	3	4	5	6
0	1	1	1	1	1	1
1	2	3	4	5	6	7
2	3	7	13	21	31	43
3	4	13	34	73	136	229
4	5	21	73	209	501	1045
5	6	31	136	501	1546	4051
6	7	43	229	1045	4051	13327

Table 5.1: Number of joint events as a function of the number of measurements n and targets T.

5.2 Joint Likelihood Filter

The JPDAF, though a useful advance over the PDAF, lacks certain desirable properties. First and foremost among these is its inapplicability to mixtures of different kinds of trackers. This limitation stems from the JPDAF's requirement that every tracker have the same image likelihood $p(\mathbf{I} \mid \mathbf{X})$, meaning that a candidate image feature for one tracker can be plausibly associated with any other. The joint target-measurement association stage in which measurements are pooled between trackers is only meaningful if such a stipulation is met. Thus, when tracking one object with a snake and another with a homogeneous region, for example, the JPDAF must be replaced by two independent PDAF trackers because there is no information sharing between modalities. Furthermore, tasks that involve the tracking of multiple differently-colored homogeneous regions, different-appearing textured regions, or differently-shaped snakes also require PDAF-based tracking. This is because dissimilar color models $(\mathbf{\Sigma}, \mathbf{T})$, reference images $\mathbf{I}_R$, or predicted edge locations $\mathbf{\Lambda}(i)$ between trackers of the same modality also engender different image likelihoods, leading to the same problem.

The JPDAF has another practical shortcoming. A key requirement which we have tried to address above is its expectation that T measurements are generated for

T tracked objects. Although our measurement process, which combines random sampling, gradient ascent, and minimum separation, works well in most circumstances to ensure this, it can encounter difficulties when some of the targets overlap one another. The primary reason is the JPDAF's assumption that the image likelihoods of multiple objects are independent when they actually are not. We can see this by examining the analog of Equation 2.2 for multiple object states (assuming conditioning on previous images):

$$p(\mathbf{X}_1, \ldots, \mathbf{X}_T \,|\, \mathbf{I}) = kp(\mathbf{I} \,|\, \mathbf{X}_1, \ldots, \mathbf{X}_T)p(\mathbf{X}_1, \ldots, \mathbf{X}_T) \tag{5.3}$$

The last term on the right hand side of this equation, which we shall call the *joint state prior*, is essentially embodied in the JPDAF by the joint feasibility logic in Equation 5.1. However, thus far we have assumed that the first term on the right hand side, which we call the *joint image likelihood*, can be factored as $p(\mathbf{I} \,|\, \mathbf{X}_1, \ldots, \mathbf{X}_T) = p(\mathbf{I} \,|\, \mathbf{X}_1) \cdots p(\mathbf{I} \,|\, \mathbf{X}_T)$. Evaluating image likelihoods independently and taking their product as the joint event's image likelihood is an approximation. This approximation tends to break down when targets are very close or overlapping because this is exactly when their appearances become dependent on one another. When object A occludes or abuts object B, it affects our expectations about the appearance of object B and at least part of the immediate background of both objects. The best case outcome of ignoring this effect is that noise in the state estimate can increase, but at worst a systematic bias can be introduced in the position, angle, or scale estimate that leads to mistracking when the overlap ends. In order to track objects most accurately, it is necessary to evaluate their image likelihoods jointly.

In order to factor $p(\mathbf{I} \,|\, \mathbf{X}_1, \ldots, \mathbf{X}_T)$ properly, we need a depth ordering, relative to the camera, of the tracked objects. Knowing which object is in front of which when they overlap is the key to properly predicting the image's appearance $\pi(\mathbf{X}_1, \ldots, \mathbf{X}_T)$ from the objects jointly. When tracking non-planar objects, it is possible that two objects will be shaped and positioned in such a way that they are mutually occlusive. We neglect this configuration as highly unlikely for the types of objects that we track, and let the relative depth D of some representative point on the object suffice for that of the whole. Such a relative depth can be constructed straightforwardly from an object's state if it includes a numerical depth estimate. Otherwise, we can either assume a fixed ordering of the objects being tracked, if warranted, or attempt to deduce the ordering from the image.

The joint image likelihood effectively functions as the joint event probability in Equation 5.1 since it encodes a measurement association (as well as the likelihood of that measurement) for every target. By sampling the prior in state space for each tracker we can build up a joint measurement $\mathbf{Z}^J$ and directly assess its likelihood without incurring the combinatorial penalty associated with the JPDAF. Repeating this joint sampling step yields a pool of joint samples. For feasibility reasons identical to those outlined above, only the most likely joint sample can be used as a joint measurement to update the tracker states. We call the process that results from these changes the Joint Likelihood Filter (JLF). This details of this process are presented in the next two sections.

As a final note, we should be careful to distinguish a situation in which T different objects are to be tracked separately (possibly with an assortment of modalities) from one where a single object will be tracked with T different modalities—e.g., a person's head is tracked by a snake for its silhouette and a homogeneous region for its skin color. In this chapter we address only the former case. The latter case implies a physical linkage between the T putative "objects" in the sense that they are distinct attributes of the same underlying object. Consequently, there is a

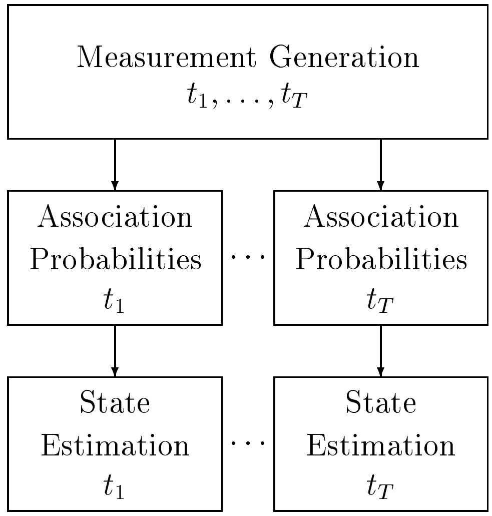


Figure 5.5: Joint Likelihood Filter pipeline

dependent relationship between the objects' states, and the joint state prior is not decomposable: $p(\mathbf{X}_1, \ldots, \mathbf{X}_T) \neq p(\mathbf{X}_1) \cdots p(\mathbf{X}_T)$. Constraints between objects are covered in Chapter 6.

Of course, the state priors are not truly independent anyway because solid objects cannot occupy the same point in space. Nonetheless, we assert that the approximation is close enough to be valid.

5.2.1 Joint Measurement Process

The JLF replaces the independent measurement generation processes for each tracker with a *joint measurement* process, diagrammed in Figure 5.5.

The first step in the measurement process is to generate n joint samples. A given joint sample $\mathbf{X}_i^J$, $1 \leq i \leq n$, is built from T *component samples* $\mathbf{X}_j$, $1 \leq j \leq T$, each generated by one of the trackers in its state space $\mathcal{X}_j$. The component sampling process is the same as that used by PDAF and JPDAF trackers: a sample is generated either randomly from the distribution defined by the predicted state $\widehat{\mathbf{X}}_j$ and sampling covariance $\mathbf{\Sigma}_{\mathcal{X}_j}$, or nonrandomly (when, for example, pure gradient ascent is being used). No gradient ascent is yet performed, however. The component samples are then stacked to get a joint sample: $\mathbf{X}_i^J = (\mathbf{X}_1, \ldots, \mathbf{X}_T)^T$, so $\mathcal{X}^J = \mathcal{X}_1 \times \ldots \times \mathcal{X}_T$. Component measurements can be derived from component state samples via $\mathbf{Z}_j = \mathbf{H}(\mathbf{X}_j)$. Associations, in the JPDAF sense, are implicit: target j is associated with component sample $\mathbf{Z}_j$.

An example of a joint sample comprising a textured region and a snake is shown in Figure 5.6(a). The textured region is tracking a chess pawn and the snake is tracking a knight.

The second step for each joint sample is to pick the most likely depth ordering of the T component samples in it. If we are assuming a fixed ordering or the actual depth of each object can be derived from state information, this is a constant-time operation and the algorithm proceeds to step three. Otherwise, we attempt to deduce the ordering in a principled way. To do this, all permutations of depth orderings are enumerated, tagging each component sample with a depth order index in the process. For example, if three objects t_1, t_2, t_3 are being tracked, we hypothesize the following set of orderings: $\{(t_1, t_2, t_3), (t_1, t_3, t_2), (t_2, t_1, t_3), (t_2, t_3, t_1), (t_3, t_1, t_2), (t_3, t_2, t_1)\}$, where (t_i, t_j, t_k) indicates that $D(t_i) \leq D(t_j) \leq D(t_k)$. Different depth orderings of non-overlapping component samples are visually equivalent, inducing equivalence classes of depth orderings, so we automatically eliminate all but one representative of each class. If none of the component samples of a joint sample overlap, there is only one such class and thus only one depth ordering must be examined. In the worst case, each sample overlaps every other sample and all $T!$ permutations must be examined, but this occurrence becomes very unlikely for large T.

Since there are two overlapping component samples in the joint sample of the chess example referred to above, there are two depth ordering hypotheses. Hypotheses corresponding to the pawn being in front of the knight and the knight being in front of the pawn are represented in Figure 5.6(b) and (e), respectively. For illustrative purposes, nearer objects are drawn brightly and distant objects more darkly.

Let $\mathbf{D}_{\mathbf{X}_i^J} = \{\mathbf{d}_1, \ldots, \mathbf{d}_{K_{\mathbf{X}_i^J}}\}$ be the set of visually distinct depth order permutations of joint sample $\mathbf{X}_i^J$. From above, we have that $1 \leq K_{\mathbf{X}_i^J} \leq T!$. Two options at this point are: (1) Perform gradient ascent on the joint sample for each $\mathbf{d}_i$ (a *joint image likelihood* objective function is described in the next section) before choosing the most likely ordering, or (2) Only do gradient ascent on the most likely ordering. The latter choice is at least as efficient in all situations as the former, often much more so, and empirically works quite well, so we have adopted it for all of the examples unless otherwise noted.

The third and final step of the joint measurement process is to select the most probable of all of the joint samples $\mathbf{X}_i^J$ and convert it to a *joint measurement* $\mathbf{Z}^J$. The component measurements $\mathbf{Z}_1, \ldots, \mathbf{Z}_T$ of $\mathbf{Z}^J$ are then plugged into Kalman filters for their associated trackers. Only one measurement is used for state estimation for the same reason outlined in the JPDAF section: linear combinations of joint measurements (the analog of the JPDAF joint event) can result in incorrect image interpretations.

The depth ordering of the joint state is derived directly from the depth ordering of the joint measurement. Two consequences of this choice are that objects can change their depth ordering from frame to frame (i.e., there is no model of solidity) and that the joint tracker may assert that one object occludes another when their states are only very close to intersecting.

5.2.2 Joint Image Likelihood

To evaluate the likelihood of a particular joint sample $\mathbf{X}^J$ and its depth ordering $\mathbf{D}_{\mathbf{X}^J}$, the probabilities of its component samples need to be computed *jointly*. A key difference between this operation and the independent approach of the PDAF and JPDAF is our ability to predict occlusions between objects. When one object is hypothesized to be in front of another, our expectations about the occluded object's appearance change. Trackers of snakes will not expect edges to be found where they are blocked from view, homogeneous region trackers will not expect occluded pixels to fit the color model, and textured region trackers will not expect occluded pixels in the comparison image to match the corresponding pixels in the reference image. Specifically, $\mathbf{D}_{\mathbf{X}^J}$ allows us to *mask* [76, 98] occluded portions of objects such that the occluding objects take precedence in the formation of a jointly predicted image $\pi(\mathbf{X}_1, \ldots, \mathbf{X}_T)$. Those pixels which are predicted to be obstructed are ignored and those predicted to be visible are matched normally.

The masking procedure induced by $\mathbf{D}_{\mathbf{X}^J}$ is fairly simple. Its output is a binary mask $\mathbf{M}_j$ for each target t_j that is the size of the image $\mathbf{I}$. $\mathbf{M}_j(x, y) = 1$ indicates that the image pixel $\mathbf{I}(x, y)$ comes from target t_j (i.e., t_j is visible at that pixel) and $\mathbf{M}_j(x, y) = 0$ indicates that the pixel belongs to either another object or the background. We are assuming that objects are completely opaque and that the resolution of the imaging device is high enough to neglect the effect of multiple objects contributing to individual pixels.

Iterating over the jointly-tracked objects from closest to farthest away, $\mathbf{M}_j$ is constructed for each object by setting every pixel (x, y) in the object's *interior* to 1 provided that $\mathbf{M}_i(x, y) = 0$ for all objects t_i for which $D(t_i) < D(t_j)$. The interiors of regions are simple rectangles, but the frames of homogeneous regions are not masked because they represent expectations about the background. The interiors of periodic (closed) snakes are defined by their B-splines, and the interiors of nonperiodic snakes are completed by connecting their endpoints. The latter method assumes that the connecting line will not cross any segment of the spline itself.

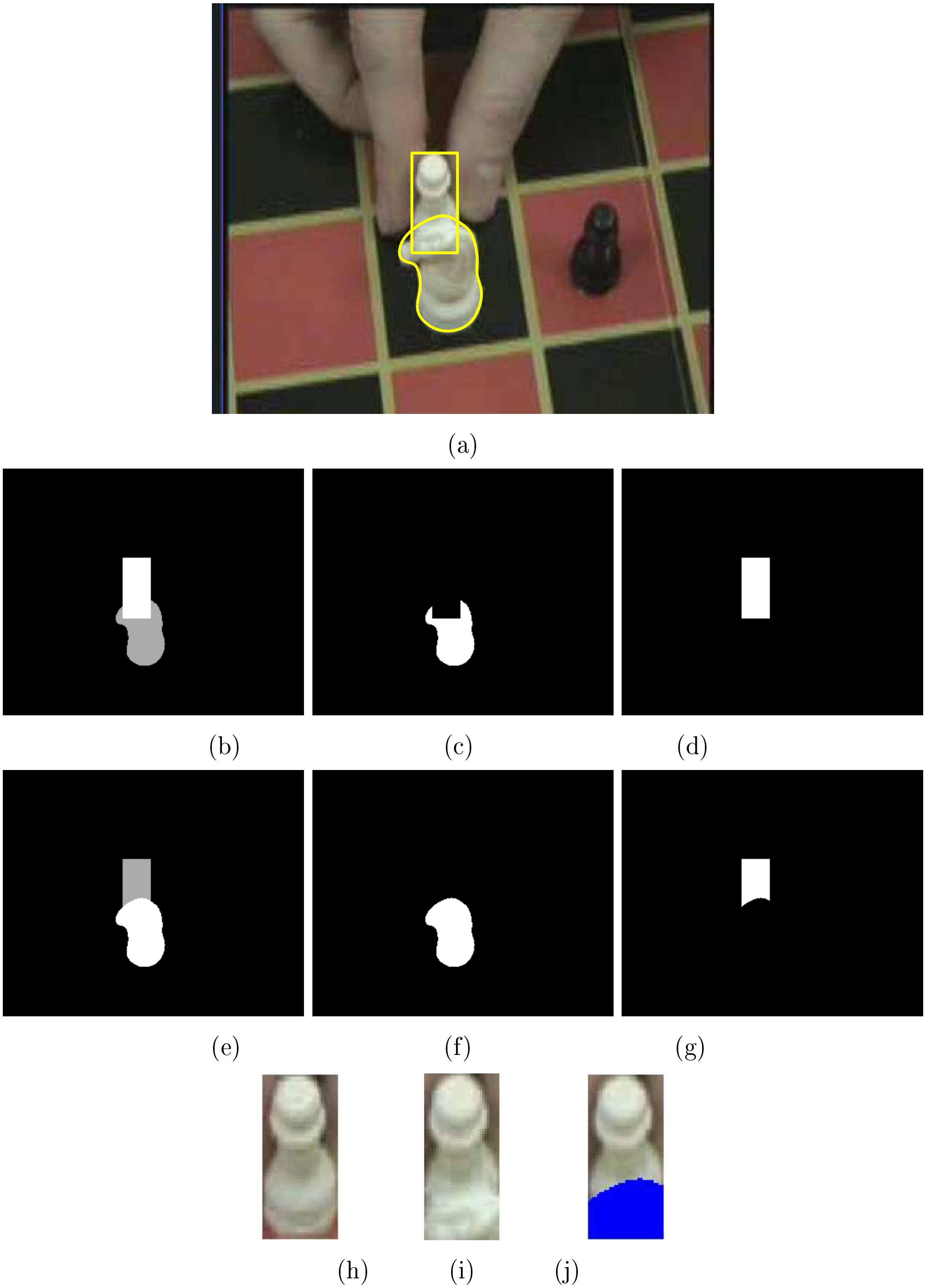


Figure 5.6: Joint Likelihood Filter: Hypothesizing depth orderings. (a) Joint measurement; (b) 1st depth ordering; (c) 1st knight mask; (d) 1st pawn mask; (e) 2nd depth ordering; (f) 2nd knight mask; (g) 2nd pawn mask; (h) Pawn reference image; (i) 1st pawn comparison image; (j) 2nd pawn comparison image.

$\mathbf{M}_{knight}$ is shown for the two depth ordering hypotheses of the chess example in Figure 5.6(c) and (f). $\mathbf{M}_{pawn}$ is shown for those two hypotheses in Figure 5.6(d) and (g). Note the alteration in shape of the mask when an object is partially occluded.

A basic technique behind the formulation of the independent image likelihoods for the various modalities, as set out in Chapter 3, is to compute a mean match value ψ over the extent or around the perimeter of the object. This mean match is transformed into a likelihood by an exponential or sigmoidal function that takes better mean matches closer to 1 and worse ones closer to 0. For homogeneous regions, the sum of the distances between every pixel and the color model is calculated, then divided by the area of the region's rectangle (with some additional complications involving the inhibitory frame). Virtually the same operation is carried out for textured regions (with a different model for each pixel and no inhibitory frame). For snakes the sum of the distances between the predicted locations and the edges found along the normals is divided by the number of normals.

Under the JLF, the set of masks $\{\mathbf{M}_j\}$ is used to modify this technique for two reasons. First, some pixels are erroneously counted more than once by the PDAF and JPDAF when tracked objects overlap; each pixel should only be used as evidence by one tracker. Second, the masks are used to try to ensure that each pixel is counted by the correct tracker. An approach that meets these criteria only counts target pixels that are predicted to be visible in the calculation of that target's mean match value.

For a textured region t_j, this means that only those interior pixels (x, y) for which $\mathbf{M}_j(x, y) = 1$ contribute to the mean match value. That is, portions of the region's interior that are not visible do not have a match value computed and are subtracted from the effective area. This method is illustrated for the textured-region pawn of the chess example in Figures 5.6(h),(i), and (j). Figure 5.6(h) shows the reference

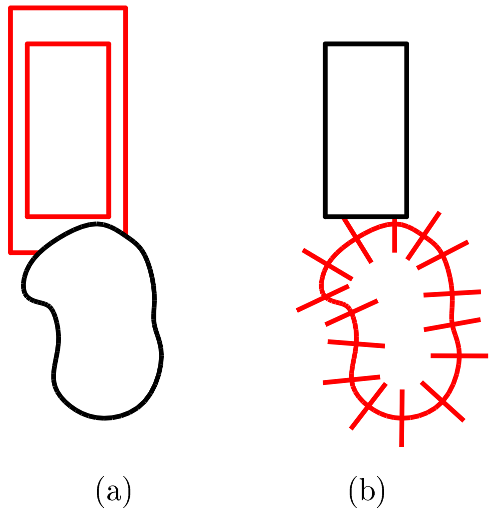


Figure 5.7: Joint Likelihood Filter: Depth-independent object interactions. (a) Nearby snake occludes expected background (frame) of homogeneous region; (b) Nearby region limits edge detection along normals of snake.

image for the pawn. Figure 5.6(i) shows the comparison image for the hypothesis that the pawn is in front of the knight and Figure 5.6(j) shows the comparison image for the opposite hypothesis. In the latter case, the nearer knight masks out the area of pixels shown in blue.

Homogeneous regions are slightly more subtle. The central area is handled in the same fashion as textured regions, but the inhibitory frame is not in the mask $\mathbf{M}_j$ of the tracker. Rather, only those pixels (x, y) in the inhibitory frame for which $\mathbf{M}_i(x, y) = 0$ for all $i \neq j$ are counted. The same method is also used for snakes: only edges found at locations (x, y) such that $\mathbf{M}_i(x, y) = 0$ for all $i \neq j$ are considered. This pixel exclusion is shown in Figure 5.7(a) and (b) for homogeneous regions and snakes, respectively.

Finally, any pixels in the interior, frame, or on the normals of an object that are also outside of the image are treated as masked out.

The mechanics of masking, though important, are only part of a proper joint formulation of the image likelihood. Without an additional processing step, a different form of the state collapse that joint methods are intended to eliminate is possible. Consider a tracking task involving two textured regions. Joint measurements for which the component measurements overlap will result in one of the regions having some fraction f of its area masked out. As $f \to 1$, the image likelihood for that component will be based on fewer and fewer pixels. Suppose that for some large f the visible portion of the region perfectly matches the corresponding part of the reference image. The mean match for such a component measurement will be the same as if the entire region were visible and perfectly matched to the reference image. The latter scenario is much less probable than the former, so once they go into a large or total occlusion configuration the trackers are unlikely to separate.

Such "stickiness" can occur even if the two textured regions are subsequently both completely visible. Since we are assuming that the number of objects is known, if two image features match the targets then the trackers should know that there cannot be a copy hiding somewhere else. Clearly, we need a heuristic that favors image interpretations containing more visible objects over those with fewer visible ones, all else being equal. Not dividing by the area (i.e., going from the mean match to the total match) would solve this problem, but normalization by the area is necessary to eliminate a bias toward smaller measurements.

Since they represent a lack of information, masked pixels should not count for or against a particular hypothesis, whereas the above formulation tends to increase the likelihood of a masked hypothesis. This intuition can be implemented by classifying visible pixels as either positive or negative evidence for the hypothesis that the target is in a certain state, and putting masked pixels in a third, neutral category rather than ignoring them. What makes a pixel a *match*, or positive evidence instead of negative, varies between modalities, but fundamentally it is a threshold Υ in ψ. To quantify this approach, matching pixels will be assigned a value of 1, non-matching pixels will be assigned a value of -1, and masked pixels will get 0. Measurements with more total evidence in their favor are assigned a higher likelihood than those with no or negative evidence by using the sigmoid function $\mathrm{sig}\,(x) = \frac{1}{1+e^{-x}}$ on the sum of the pixel match values.

Specifically, we replace the independent image likelihoods $p(\mathbf{I} \mid \mathbf{X})$ for homogeneous regions, textured regions, and snakes from Chapter 3 with the following component image likelihoods $p^J(\mathbf{I}\,|\,\mathbf{X}_j)$:

Textured region

$$p^J_{tregion}(\mathbf{I}\,|\,\mathbf{X}_j) = \mathrm{sig}\,(\sum_{x,y \in \mathbf{I}_R} a(x,y) \cdot \psi^J_{tregion}(x,y)) \tag{5.4}$$

where

$$\psi^J_{tregion}(x,y) = \begin{cases} 1 & \text{if } \mathbf{M}_j(x,y) = 1 \wedge (\mathbf{I}_R(x,y) - \mathbf{I}_C(x,y))^2 \leq \Upsilon_{tregion} \\ -1 & \text{if } \mathbf{M}_j(x,y) = 1 \wedge (\mathbf{I}_R(x,y) - \mathbf{I}_C(x,y))^2 > \Upsilon_{tregion} \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

Homogeneous region

Recall that C is the interior of the homogeneous region and F is its inhibitory frame.

$$p^J_{hregion}(\mathbf{I}\,|\,\mathbf{X}_j) = \mathrm{sig}\,(\sum_{x,y \in C \cup F} a(x,y) \cdot \psi^J_{hregion}(x,y)) \tag{5.6}$$

where

$$\psi^J_{hregion}(x,y) = \begin{cases} 1 & \text{if } \mathbf{M}_j(x,y) = 1 \wedge (x,y) \in C \wedge \gamma(\mathbf{I}(x,y),\mathbf{T}) \leq \Upsilon_{hregion} \vee \\ & \mathbf{M}_i(x,y) = 0 \;\forall\; i \neq j \wedge (x,y) \in F \wedge \gamma(\mathbf{I}(x,y),\mathbf{T}) > \Upsilon_{hregion} \\ -1 & \text{if } \mathbf{M}_j(x,y) = 1 \wedge (x,y) \in C \wedge \gamma(\mathbf{I}(x,y),\mathbf{T}) > \Upsilon_{hregion} \vee \\ & \mathbf{M}_i(x,y) = 0 \;\forall\; i \neq j \wedge (x,y) \in F \wedge \gamma(\mathbf{I}(x,y),\mathbf{T}) \leq \Upsilon_{hregion} \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

Snake

Let $\mathbf{z}_x(k), \mathbf{z}_y(k)$ be the x and y image coordinates, respectively, of the best edge (if any) found along normal k. We have:

$$p^J_{snake}(\mathbf{I}\,|\,\mathbf{X}_j) = \text{sig}\,(\sum_{k=0}^{n-1} l(k) \cdot \psi^J_{snake}(k)) \tag{5.8}$$

where

$$\psi^J_{snake}(k) = \begin{cases} 1 & \text{if } \mathbf{M}_i(\mathbf{z}_x(k),\mathbf{z}_y(k)) = 0 \;\forall\; i \neq j \wedge |\mathbf{\Lambda}(k) - \mathbf{z}(k)| \leq \Upsilon_{snake} \\ -1 & \text{if } \mathbf{M}_i(\mathbf{z}_x(k),\mathbf{z}_y(k)) = 0 \;\forall\; i \neq j \wedge |\mathbf{\Lambda}(k) - \mathbf{z}(k)| > \Upsilon_{snake} \\ & \vee \text{ edge not found} \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

For the Joint Likelihood Filter, it is not necessary to choose appropriate values of σ^2 to weight each modality's contribution to achieve a normalized "common" scale. This is because the value of ψ^J for each modality has the same range of $[-1, 1]$. A common range for ψ^J is achieved through a different scale factor: the threshold Υ. The results in this dissertation, unless otherwise noted, use the values $\Upsilon_{tregion} = 30$,

$\Upsilon_{hregion} = 3$, and $\Upsilon_{snake} = L/2$, where L is the length of one of the snake's edge-search normals.

Let the joint tracker, which has T component trackers, consist of a set $\mathcal{H}$ of homogeneous region trackers, a set $\mathcal{T}$ of textured region trackers, and a set $\mathcal{S}$ of snake trackers such that $T = |\mathcal{H}| + |\mathcal{T}| + |\mathcal{S}|$. With the component image likelihoods defined as above, the image likelihood of the joint sample $\mathbf{X}^J$ is simply their product:

$$p^J(\mathbf{I}\,|\,\mathbf{X}^J) = \prod_{t_j \in \mathcal{H}} p^J_{hregion}(\mathbf{I}\,|\,\mathbf{X}_j) \prod_{t_j \in \mathcal{T}} p^J_{tregion}(\mathbf{I}\,|\,\mathbf{X}_j) \prod_{t_j \in \mathcal{S}} p^J_{snake}(\mathbf{I}\,|\,\mathbf{X}_j) \tag{5.10}$$

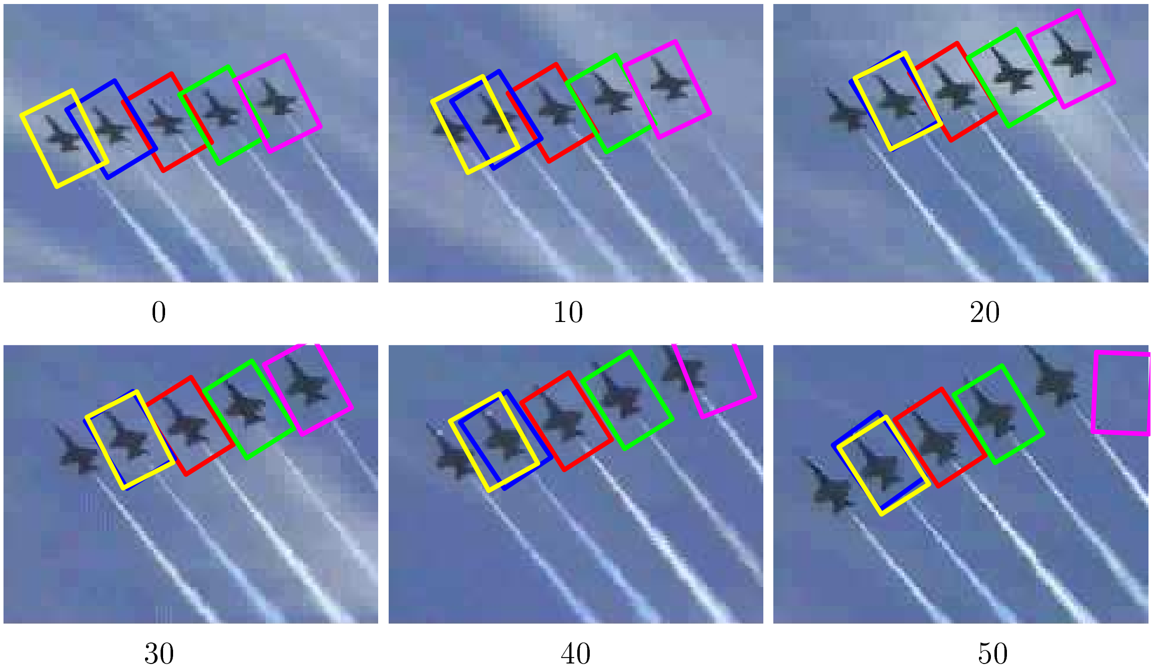
It is straightforward to perform gradient ascent on the joint image likelihood to improve the component samples. Note that gradient ascent does not change the depth ordering of the component samples, however. We have also observed better results using Powell's method [94] rather the conjugate gradient method for the joint image likelihood, possibly because of the effect on its differentiability introduced by the nonlinear match classification step.

5.3 Results

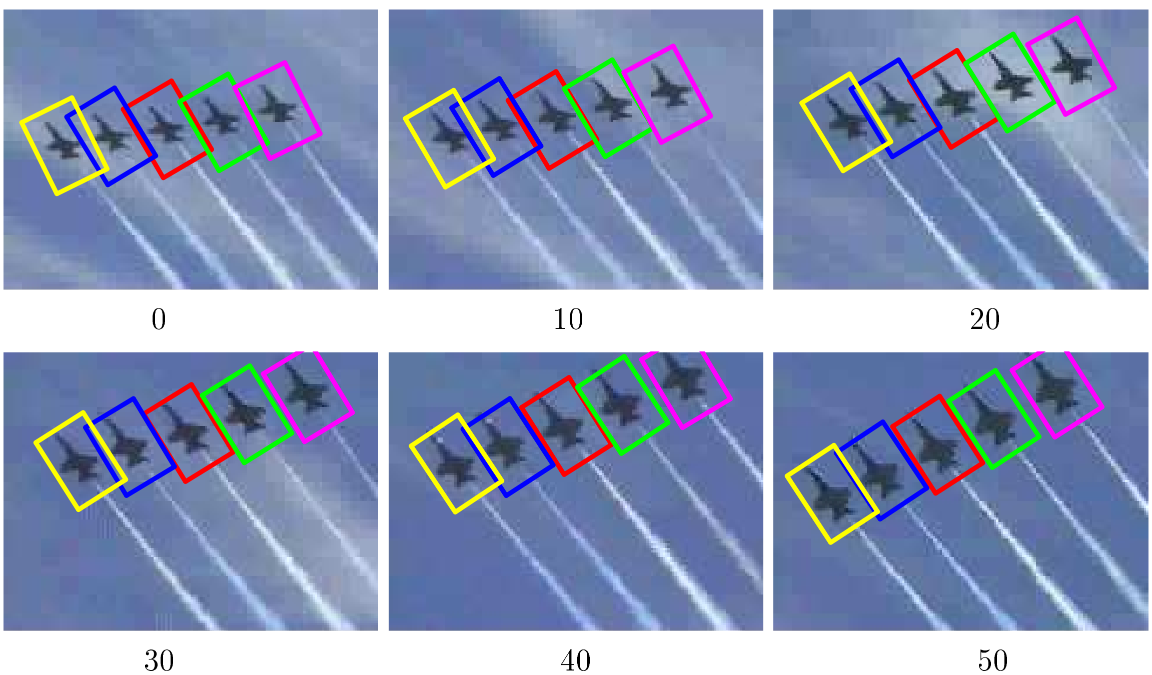
Figure 5.8 illustrates the superiority of the JPDAF over the PDAF for tracking multiple objects in close proximity. In this example, five airplanes flying in formation are tracked using textured regions. The planes scale slightly, but their primary motion is translational and rotational, so the state of each tracker is expressed as $\mathbf{X} = (x, y, \phi, \dot{x}, \dot{y})$, making measurement space $\mathcal{Z} = X \times Y \times \Phi$. For both the PDAF and JPDAF examples, each tracker selects the best 3 of 100 samples, where the state sampling covariance is $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 0.01 \end{smallmatrix}\right)$. Each of these samples is then improved using Powell's method for gradient ascent on the image likelihood function $p(\mathbf{I}\,|\,\mathbf{X})$. For the PDAF, each tracker's set of hill-climbed samples is thinned independently by enforcing a minimum separation of 10 pixels horizontally and vertically, and 0.1 radians. The resulting samples become the measurements for each tracker. The JPDAF creates a combined pool of measurements by thinning the union of the two sets of hill-climbed samples using the same minimum separation.

The image likelihood function has a peak for each of the five planes, and the independent sampling and hill-climbing done by the PDAF trackers results in much instability. From frame to frame, each tracker may choose the correct peak as its measurement or wind up with the one to the left or right. This engenders much noise in the state estimation at best, but when the wrong peak is consistently used as the measurement, a tracker can be pulled off of the right plane. This phenomenon is illustrated in the figure from frame 0 to frame 20. The JPDAF avoids such eventualities by rejecting two trackers claiming the same measurement as infeasible. The mistracking of the far right plane by the PDAF at the end of the sequence is due to the difficulty of matching at the edge of the image. The JPDAF most likely tracks this plane successfully because even if the rightmost tracker does not find its peak, the tracker to its immediate left does, adding the measurement to the shared pool. This effectively increases the number of samples examined by each tracker.

Figure 5.9 demonstrates the efficacy of the JPDAF vs. the PDAF for tracking the faces of two people in profile as they walk toward and then past one another. Using translating homogeneous regions with identical dimensions and the same skin color model, the state of each tracker is expressed as $\mathbf{X} = (x, y, \dot{x}, \dot{y})$, making measurement space $\mathcal{Z} = X \times Y$. For both the PDAF and JPDAF examples, each tracker selects the best 10 of 50 samples, where the state sampling covariance is $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 100 & 0 \\ 0 & 100 \end{smallmatrix}\right)$. Each of these samples is then improved by performing conjugate gradient ascent on the



(a) PDAF



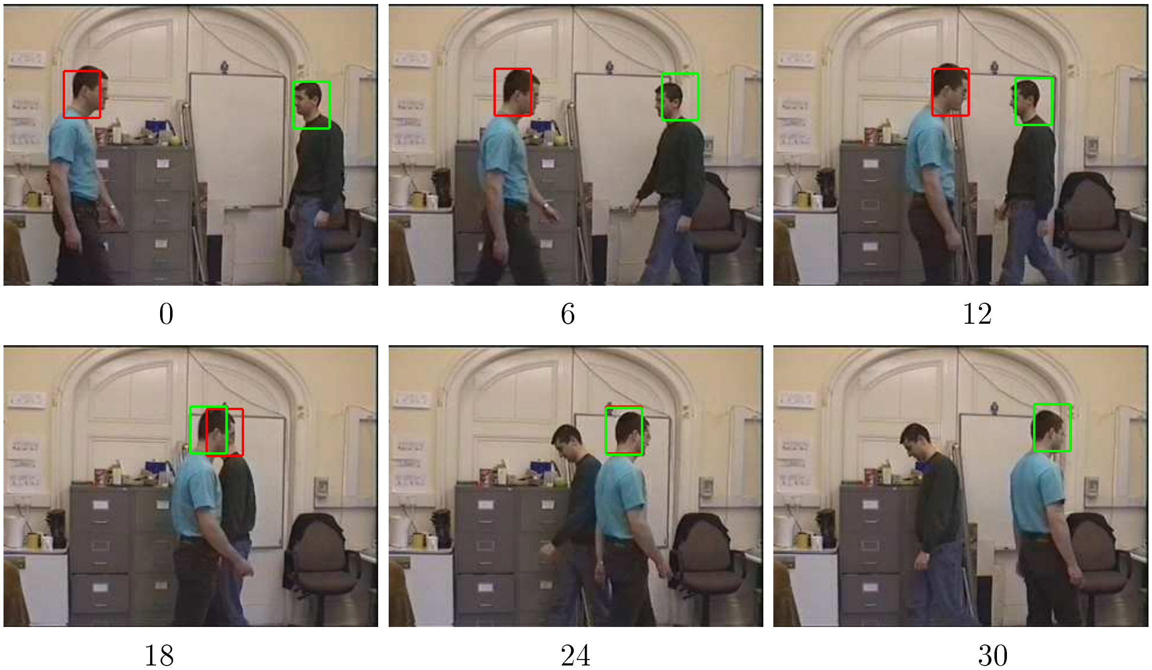
(b) JPDAF

Figure 5.8: JPDAF: Handling nearby textured regions (MPEG). (a) Frame sequence using five PDAF trackers; (b) Frame sequence using JPDAF tracking.

image likelihood function $p(\mathbf{I} \,|\, \mathbf{X})$. For the PDAF, each tracker's set of hill-climbed samples is thinned independently by enforcing a minimum separation of 10 pixels horizontally and vertically. The resulting samples become the measurements for each tracker. The JPDAF arrives at a combined pool of measurements by thinning the union of the two sets of hill-climbed samples using the same minimum separation. With these parameters, the JPDAF successfully tracked both heads through the crossing, maintaining the correct associations, in 10 out of 10 trials. The PDAF failed in 10 out of 10 trials. In every case, the tracker assigned to the head of the person walking to the left was distracted by the rightward-moving head, most likely because it was nearer to the camera and thus larger, resulting in both trackers locking onto the same image feature.

The ability of the Joint Likelihood Filter to infer the depth ordering of tracked objects is illustrated in Figure 5.10. A white pawn chess piece is tracked by a textured region as it moves behind and is thus partially occluded by a white knight, which is tracked by a snake. There is negligible scaling or rotation over the duration of the sequence, so the state of each tracker is simply its image position and velocity, and is expressed as $\mathbf{X} = (x, y, \dot{x}, \dot{y})$, making each component's measurement space $\mathcal{Z} = X \times Y$. The snake has 16 segments. Measurement generation is done using pure gradient ascent with Powell's method. A tracker's outline, normally red or green, is drawn in gray when the joint measurement and its depth ordering indicate that it is partially occluded. The fact that the pawn is behind the knight during the middle section of the tracking sequence is correctly deduced. There is some noise in the occlusion inference, however, at the very beginning and end of the two pieces' overlap because it is based on so little information.

Another example of depth-order inference is given in Figure 5.11. Here, two homogeneous regions with states $\mathbf{X} = (x, y)$ and the same measurement parameters



(a) PDAF

(b) JPDAF

Figure 5.9: JPDAF: Handling crossing homogeneous regions (MPEG). (a) Frame sequence using two PDAF trackers; (b) Frame sequence using JPDAF tracking.

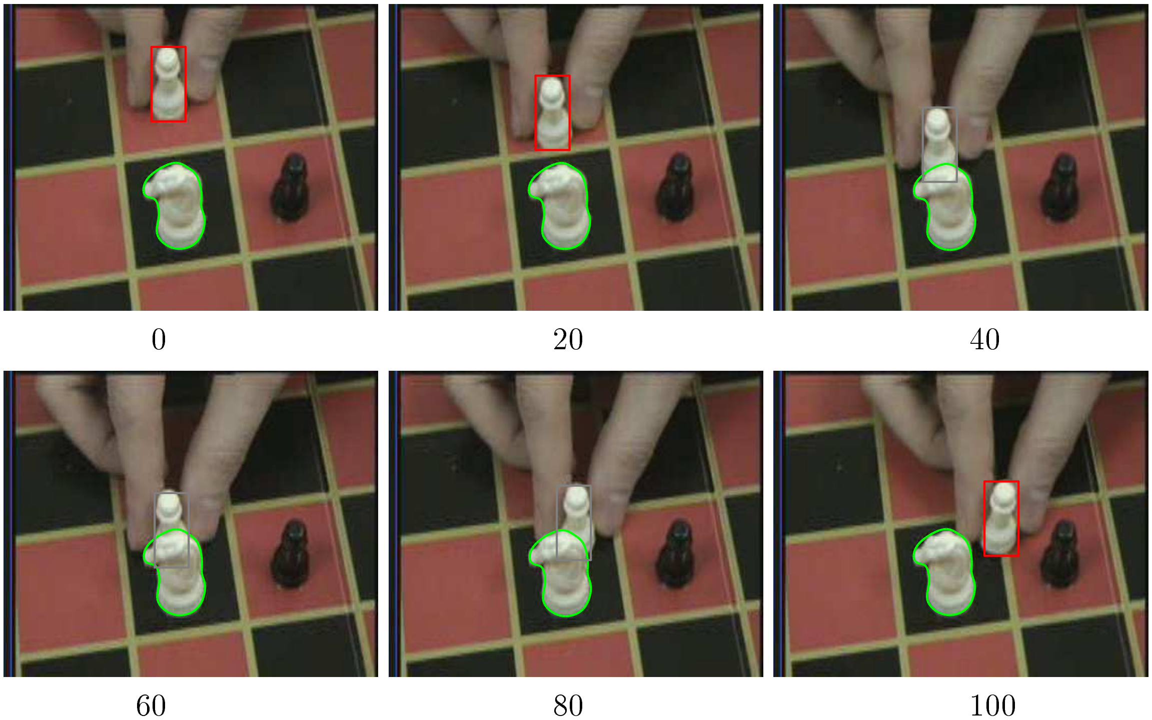
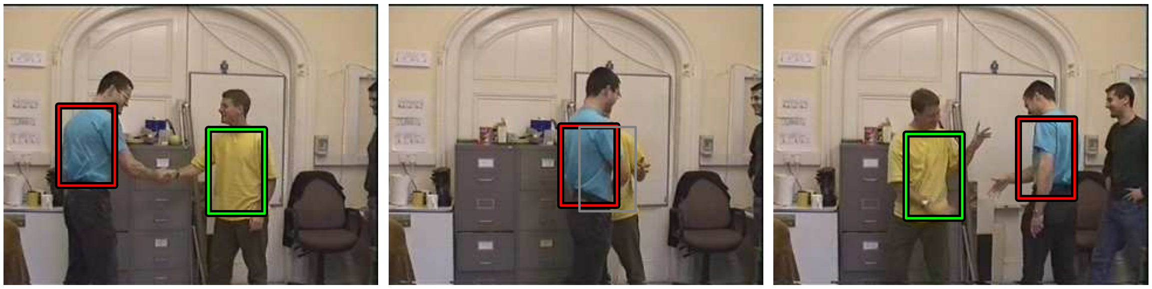


Figure 5.10: Joint Likelihood Filter: Deducing the occlusion relationship between a textured region and snake (MPEG).

track the colorful t-shirts of two people shaking hands. Powell's method is used to hill-climb the best 3 of 250 joint samples (using $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 200 & 0 \\ 0 & 200 \end{smallmatrix}\right)$ for each component sample of the joint sample); the best one of these three is used to update the state.

The Joint Likelihood Filter (JLF) also matches the ability of JPDAF to track multiple crossing objects, as illustrated in Figure 5.12. In this example, two airplanes flying in close formation are tracked using textured regions as they overlap and separate. The planes scale, translate, and rotate, so the state of each tracker is expressed as $\mathbf{X} = (x, y, \phi, s)$, making measurement space $\mathcal{Z} = X \times Y \times \Phi \times S$. In the PDAF example, each tracker selects the best 5 of 250 samples, where the state sampling covariance is $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 0.02 & 0 \\ 0 & 0 & 0 & 0.01 \end{smallmatrix}\right)$. Each of these samples is then improved using Powell's method for gradient ascent on the image likelihood function $p(\mathbf{I} \mid \mathbf{X})$. Finally, each tracker's set of hill-climbed samples is thinned independently by



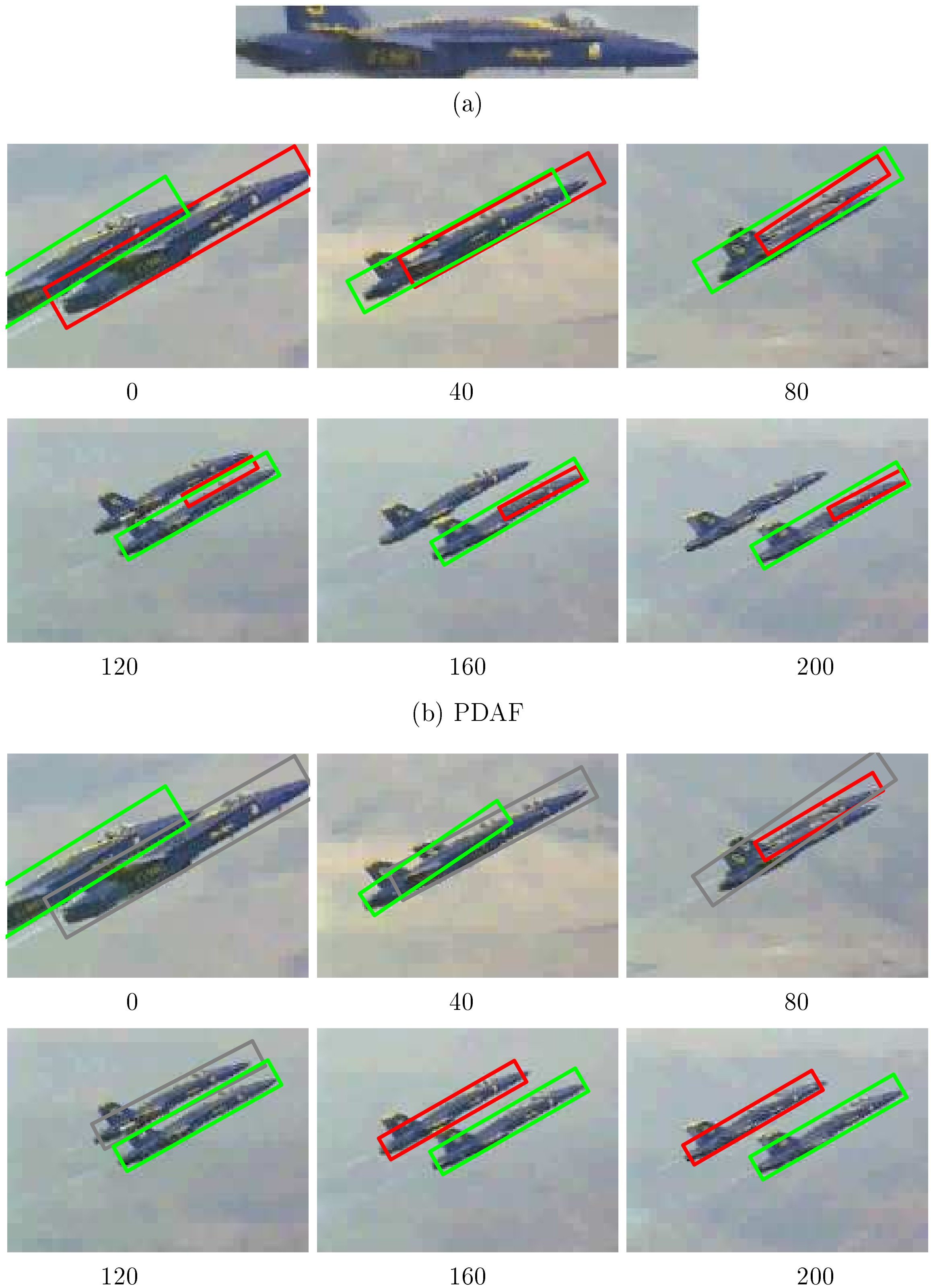
0 30 60

Figure 5.11: Joint Likelihood Filter: Deducing the occlusion relationship between two homogeneous regions (MPEG). (Sequence courtesy of J. MacCormick).

enforcing a minimum separation of 10 pixels horizontally and vertically, 0.1 radians, and 0.01 units of scale. In the Joint Likelihood Filter example, the best 5 of 250 joint samples (using $\mathbf{\Sigma}_{\mathcal{X}}$ for each component sample of the joint sample) are also improved using Powell's method on the joint image likelihood, and the best of these is used to update the state.

Calculating their image likelihoods independently, each plane tracker is attracted by the two nearby matching features in the image as they move together. When the two planes separate, both often follow the same feature, resulting in mistracking. By calculating their image likelihoods jointly, both trackers separate properly when their corresponding image features separate. This is because of the built-in preference, when the image supports it, for an interpretation that there are two visible objects over an interpretation that one visible object completely occludes the other. The random sampling technique for measurement generation is vital here because even using the joint image likelihood, a pure gradient ascent tracker can get stuck in a local minimum as the planes separate. The nonlocality of random sampling allows the trackers to jump out of suboptimal states as the planes separate unambiguously.

State estimation and the inferred depth ordering are somewhat noisy during the period of greatest overlap because of the identical coloration, shape, and markings of the two planes, and because of the poor resolution of the image.



(c) Joint Likelihood Filter

Figure 5.12: Joint Likelihood Filter: Handling crossing textured regions (MPEG). (a) Reference image; (b) Frame sequence showing states of two PDAF trackers; (c) Frame sequence with JLF tracker's state.

Chapter 6

Constrained Tracking

An important assumption of the PDAF, JPDAF, and Joint Likelihood Filter algorithms is that the incidence of occlusions and distractions caused by untracked objects or other visual phenomena is reasonably approximated by a uniform or Poisson noise process. When this approximation breaks down, as when such occurrences are actually due to persistent features of the visual environment, these tracking filters can yield biased results or mistrack. In the previous chapter our strategy for improving robustness for the target of interest was to try to track as many potential distractors as possible, allowing a principled prediction of their visual interactions. However, sometimes the characteristics of the background may change as the object moves or there may be too many distractors for this approach to be efficient. In this case the selection of what area of the object to focus on and what tracking modality to use become paramount in determining tracking accuracy.

With regard to making this selection, it is useful to distinguish between an object *attribute*, as defined in Chapter 4, and what we call a *part*. A part is a spatially distinct image feature physically linked to the larger object. Fundamentally, a part is *what* a tracker tracks, while an attribute is *how* the tracker identifies its target.

For example, a person's face may be found in an image by searching for a skin-colored region; or a pattern of textures matching the arrangement of the eyes, nose, and mouth; or the specific shape of the silhouette of their shoulders, neck, and head against a contrasting background. The face is a part (of the person's body), and the different ways of finding and tracking it are all attributes. Conversely, a person's hands, arms, and face, if bare, can all be identified by the same color attribute, but are separate image features and thus termed parts.

We have observed that when the noise approximation of the PDAF model is violated, tracking performance does not collapse all at once but rather by degrees. That is, the more an object is occluded or the better a distracting background feature matches an attribute used for tracking, the more severe the deterioration of accuracy and the greater the chance of outright failure. This means that larger objects or objects defined by many attributes may be less susceptible to distraction than smaller or singly-defined objects, suggesting a strategy for improving tracking robustness.

The approach we take here to the problem of persistent distractors is to try to reduce their incidence, and hence their influence, by defining a target as a conjunction of parts and/or attributes. As more image features are used to identify a target, the number of visual phenomena that may potentially distract the tracking process is reduced. Even if something of the same color is near the target, for example, it may be of the wrong shape, or another part of the object may not be distracted, allowing tracking to proceed without interference. An atomic tracker with temporarily weak discriminatory power can overcome difficult image conditions because of the *constraints* imposed by its linkage to other trackers. These force consideration of the entire ensemble of parts and attributes simultaneously when interpreting the image, helping to rule out incorrect alternatives. Constraints are only applicable, of course, when we are tracking a target complex enough that it has multiple resolvable parts and/or attributes. This is certainly true of many of the types of objects that we wish to track, such as people or cars, because they are amenable to description as a set of simpler, geometrically-connected features under most image conditions.

In this chapter we analyze the implications on the tracking process of the knowledge that two or more of the targets are physically connected. A linkage between targets means that they are parts of some larger object, and that their states are therefore not independent. This disallows the decomposition of the joint state prior $p(\mathbf{X}_1, \ldots, \mathbf{X}_T) = p(\mathbf{X}_1) \cdots p(\mathbf{X}_T)$ in Equation 5.3 that is a vital step in both the JPDAF and Joint Likelihood Filter multiple-object tracking algorithms.

As with the joint image likelihood $p^J(\mathbf{I} \mid \mathbf{X}^J)$ in the previous chapter, we need a more complex formulation of $p(\mathbf{X}_1, \ldots, \mathbf{X}_T)$ that takes into account the interactions between objects. Rather than predicting the visual results of bringing multiple, possibly independently-moving objects into a proximate or overlapping relationship, though, the joint state prior will be concerned with how multiple *linked* objects influence one another's states, even at a distance. Incorporating constraint information such as this can have a salutary effect on tracking performance by (a) shrinking search space, making processing faster for each frame, and (b) discounting or eliminating from consideration joint events/measurements that do not reflect expectations about the interrelationship of the linked parts.

In the next part of this chapter we introduce an extension to the Joint Likelihood Filter (JLF) called the *Constrained Joint Likelihood Filter*, or CJLF, that implements inter-part constraints efficiently and simply. We then present results demonstrating how the CJLF improves tracking performance in many visual situations over the previously described algorithms, and enables certain tracking tasks to be carried out for which those algorithms are not suited.

6.1 Constrained Joint Likelihood Filter

The JLF assumes that targets move independently of one another. An object such as a human body, though, violates this requirement when viewed as a group of parts: the connections between the arms, head, torso, etc. limit the possible range of their relative positions and motions. The expectation that parts or attributes of a complex tracked object will be in a particular configuration is extra information that may help distinguish the object from the background or other objects. In this section we describe modifications we have made to the JLF to encode these relationships. We call this method the Constrained Joint Likelihood Filter (CJLF), diagrammed in Figure 6.1.

The key idea behind the CJLF is an elaboration of one of the most basic kinds of constraints: limitation of the number of parameters in an object's state, which in turn reduces the size of its measurement space. We have been using this form of constraint for atomic trackers already when we analyze the object, the tracking task, and the visual environment in order to decide whether to allow the tracker to translate, scale, rotate, or even shear (for snakes). If the object to be tracked only slides back and forth horizontally, for example, or rotates in place, then there is no reason to give the tracker more than the minimal degrees of freedom required to follow that class of movement. To do otherwise only provides the tracker with an opportunity to mistrack along an extraneous state dimension.

For a multi-part or multi-attribute object, there are multiple trackers for which this kind of decision must be made. The CJLF simply formalizes the commonsense notion that a minimal state description of the entire object (or, more exactly, that portion which is being tracked) implies certain correlations between and limitations on the states of its constituent parts and attributes. As an example, suppose that we want to track the headlights of a truck driving directly toward the camera. A naive approach is to give each tracker a state with translation and scale parameters: $\mathbf{X}_{leftlight} = (x_l, y_l, s_l)$ and $\mathbf{X}_{rightlight} = (x_r, y_r, s_r)$. However, we know that the two lights are physically connected by the truck chassis and as such cannot move independently. Our state description is thus underconstrained, and a cursory analysis of the frontoparallel geometry of the truck grille indicates that $\mathbf{X}_{leftlight}$ can be immediately derived from $\mathbf{X}_{rightlight}$ (or vice versa). In other words, only 3 rather than 6 variables are necessary to fully describe the system, though image processing at both locations of course yields more robust estimation than at one alone.

Ordinarily, a special-purpose tracker with a customized image likelihood function $p(\mathbf{I} \,|\, \mathbf{X})$ is created for tracking a complicated object like the one in the example above. The CJLF is a framework for achieving the same performance as a customized approach, but in a flexible fashion that avoids the construction of a specialized image likelihood function for each new tracking task. Rather, the CJLF works by providing a small set of rules for composing the simple, atomic trackers that we have already developed into more complex assemblages for which the joint image likelihood remains a product of component likelihoods. The rationale for this decision is twofold: (1) to reduce the amount of time spent on analysis and code writing for novel tracking tasks by permitting code reuse, and (2) to provide a standard interface for new methods to easily be integrated with existing ones.

The compositional primitives used by the CJLF are based on intuitive physical relationships such as rigid links, hinges, and fixed depth orderings. Given a set of parts or attributes with unconstrained state spaces $\mathcal{X}_1, \ldots, \mathcal{X}_T$, these rules serve as a guide for paring them down to their minimal, constrained forms: $\mathcal{X}'_1, \ldots, \mathcal{X}'_T$. When the paring removes all degrees of freedom of a tracker, as would occur with one of the headlights from the example above, its state space becomes empty. It is still

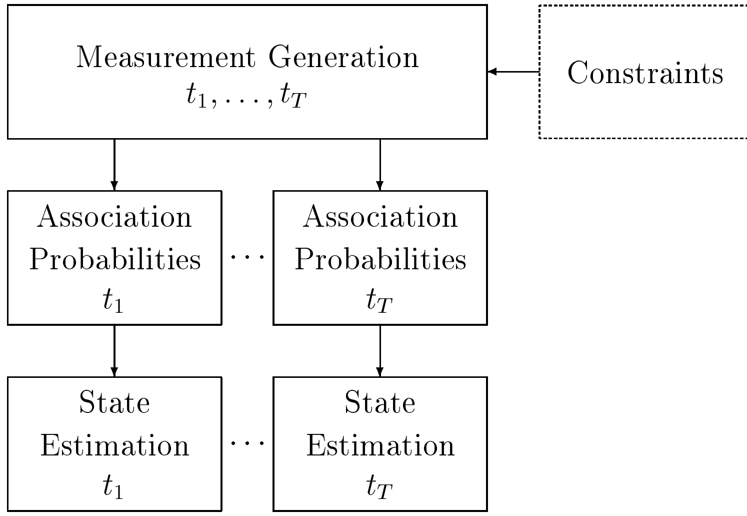


Figure 6.1: Constrained Joint Likelihood Filter pipeline

desirable to perform image processing for that tracker (such as looking for the second headlight), so as a matter of bookkeeping the notion of the tracker is retained. This process is the primary method by which constraints are introduced into the joint prior on states $p(\mathbf{X}_1, \ldots, \mathbf{X}_T)$.

In addition to reducing the degrees of freedom available to some of the trackers, the CJLF's compositional rules also indicate how to derive the image processing variables of linked parts from one another (e.g., where the left headlight is if we know the position and scale of the right headlight). The details of this derivation are explicated for each of the rules in the next section. Finally, though we do not use them for any of the examples in this chapter, the CJLF allows hard limits to be placed on state variables such as a range of permissible angles or scales. This allows further specificity in determining the form of $p(\mathbf{X}_1, \ldots, \mathbf{X}_T)$.

For purposes of implementation, the Constrained Joint Likelihood Filter approach alters the method of obtaining geometric image processing parameters detailed in Chapter 4. Let each target t_j have a measurement key $\mathbf{K}_j$. Previously the domain of each function in $\mathbf{K}_j$ was $\mathcal{X}_j$; we now extend it to the joint state space $\mathcal{X}^J$. This allows us to refer to the component measurement geometric parameters of *any* target t_i to define t_j's component measurement geometric parameters. A caveat is that care must be taken that there are no circularities in the definitions of the $\mathbf{K}_j$ for the various targets. Nonetheless, this constitutes a convenient and powerful mechanism for enforcing constraints.

The effect of this reduction in the joint state space is to alter the Joint Likelihood Filter so that it considers *only* those joint state samples which satisfy the constraints exactly, allowing their joint probabilities to be computed normally. Sampling and hill-climbing can then be used as in the previous chapter while still meeting the conditions on the interrelationship of the parts. However, as with the JPDAF and JLF, only one measurement is ultimately used to update the state. This is because the weighted combination of measurements carried out by the PDAF filters can give rise to states not satisfying the constraints.

6.1.1 Rigid link constraints

The simplest kind of constraint between measurements is a *rigid link*. By our definition, a rigid link between two objects t_1, t_2 implies that t_2's current geometric parameters are completely determined by their initial values and t_1's current values—it has no state or measurement space of its own to speak of. Its only function is to contribute to the calculation of the joint image likelihood $p(\mathbf{I} \mid \mathbf{X}_1, \mathbf{X}_2)$. Therefore t_2 does not use a Kalman filter to estimate its own state; its purpose is as an adjunct that makes t_1 a more complex visual object. As an example, suppose that two rigidly-linked objects are allowed to translate, scale, and rotate, and that the initial offset between them scales as they do. This joint object configuration is diagrammed in Figure 6.2(a). Then $\mathbf{X}_1 = (x_1, y_1, s_1, \phi_1)$ and $\mathbf{K}_1 = (x_1, y_1, s_1\bar{w}_1, s_1\bar{h}_1, \phi_1)^T$, while the geometric image processing parameters of t_2 are calculated directly from the

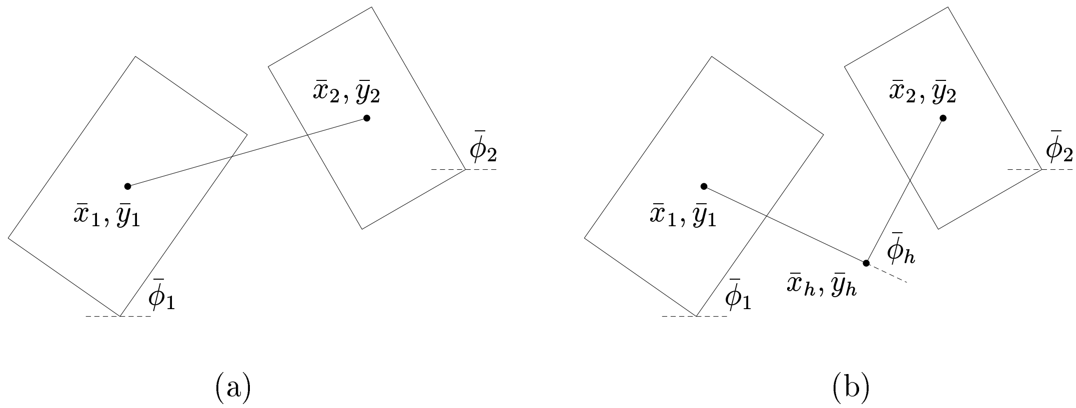


Figure 6.2: Constraint types. (a) Initial configuration of a rigid link; (b) Initial configuration of a hinge.

following:

$$\mathbf{K}_2 = \begin{pmatrix} x_1 + s_1(\delta_x \cos(\phi_1 - \bar{\phi}_1) - \delta_y \sin(\phi_1 - \bar{\phi}_1)) \\ y_1 + s_1(\delta_x \sin(\phi_1 - \bar{\phi}_1) + \delta_y \cos(\phi_1 - \bar{\phi}_1)) \\ s_1 \bar{w}_2 \\ s_1 \bar{h}_2 \\ \phi_1 + \delta_\phi \end{pmatrix} \tag{6.1}$$

where $\delta_x = \bar{x}_2 - \bar{x}_1$, $\delta_y = \bar{y}_2 - \bar{y}_1$, and $\delta_\phi = \bar{\phi}_2 - \bar{\phi}_1$. It is convenient to represent the rigid link transformation that takes the geometric parameters of object i to those of object j as a function $R_{i,j}$. Thus, $\mathbf{K}_2 = R_{1,2}(\mathbf{K}_1)$.

It is straightforward to generalize a two-part, rigidly-constrained joint object to a T-target system. T rigidly-linked parts can be modeled by treating them as $T - 1$ linked pairs, every one of which includes target t_1, which has a state, measurement space, and measurement key exactly like the first object in the example above. Then for all $i > 1$, target t_i has no state or measurement space, like the second object in the example, and its measurement key $\mathbf{K}_i$ is of the same form as that given in Equation 6.1, except that the appropriate initial parameters $\bar{x}_i, \bar{y}_i, \bar{\phi}_i, \bar{w}_i$, and $\bar{h}_i$ are substituted where $\bar{x}_2, \bar{y}_2, \bar{\phi}_2, \bar{w}_2$, and $\bar{h}_2$ appear, respectively. Using the functional notation, $\mathbf{K}_i = R_{1,i}(\mathbf{K}_1)$.

6.1.2 Hinge constraints

A somewhat more complex constraint is a *hinge* (we avoid the more common term "joint" because of its other connotations in this dissertation). A hinge is similar to a rigid link but with an angular degree of freedom granted to the second object; the axis of rotation is determined by the initial image location of the hinge: $\bar{x}_h, \bar{y}_h$ (see the diagram in Figure 6.2(b)). Using the two-part joint object from above as an example, if we allow the ensemble to translate, scale, and rotate freely and the second part to rotate independently about the hinge, then the state of the first part is again $\mathbf{X}_1 = (x_1, y_1, s_1, \phi_1)$ and $\mathbf{K}_1 = (x_1, y_1, s_1\bar{w}_1, s_1\bar{h}_1, \phi_1)^T$. However, the state of the second part is now $\mathbf{X}_2 = (\phi_2)$, where the angle represented by ϕ_2 is *relative* to the ray from (x_1, y_1) through the current hinge location (x_h, y_h). Following the derivation for a rigid link,

$$\begin{pmatrix} x_h \\ y_h \end{pmatrix} = \begin{pmatrix} x_1 + s_1(\delta^-_{h_{x_1}} \cos(\phi_1 - \bar{\phi}_1) - \delta^-_{h_{y_1}} \sin(\phi_1 - \bar{\phi}_1) \\ y_1 + s_1(\delta^-_{h_{x_1}} \sin(\phi_1 - \bar{\phi}_1) + \delta^-_{h_{y_1}} \cos(\phi_1 - \bar{\phi}_1)) \end{pmatrix} \tag{6.2}$$

where $\delta^-_{h_{x_1}} = \bar{x}_h - \bar{x}_1$ and $\delta^-_{h_{y_1}} = \bar{y}_h - \bar{y}_1$. If the initial value of the hinge angle is $\bar{\phi}_h$, the geometric parameters of the second object are:

$$\mathbf{K}_2 = \begin{pmatrix} x_h + s_1(\delta^+_{h_{x_2}} \cos(\phi_2 - \bar{\phi}_h) - \delta^+_{h_{y_2}} \sin(\phi_2 - \bar{\phi}_h)) \\ y_h + s_1(\delta^+_{h_{x_2}} \sin(\phi_2 - \bar{\phi}_h) + \delta^+_{h_{y_2}} \cos(\phi_2 - \bar{\phi}_h)) \\ s_1\bar{w}_2 \\ s_1\bar{h}_2 \\ \phi_1 + \phi_2 + \delta_\phi - \bar{\phi}_h \end{pmatrix} \tag{6.3}$$

where $\delta^+_{h_{x_2}} = \bar{x}_2 - \bar{x}_h$ and $\delta^+_{h_{y_2}} = \bar{y}_2 - \bar{y}_h$. The hinge transformation between objects i and j is denoted by $H_{i,j}$.

We can also extend the mathematics of a single hinge constraint to a system of multiple hinges. T parts connected in sequence by $T-1$ hinges form what is commonly called a *chain* [22]. Let C be a chain consisting of T hinge-connected parts: $C = (t_1, \ldots, t_T)$. We can specify the constraint on each part along C inductively: if the first and second links t_1, t_2 are defined by the two-part system introduced above, then the state of the ith part for $i > 1$ is $\mathbf{X}_i = (\phi_i)$ and its measurement space is $\mathcal{Z}_i = \Phi$. Given the measurement key $\mathbf{K}_1$ of the first part t_1, the measurement key of the ith part t_i is given by $\mathbf{K}_i = H_{i-1,i}(H_{i-2,i-1}(\ldots H_{1,2}(\mathbf{K}_1)\ldots))$. By writing $H_{i-1,i}(\mathbf{K}_{i-1})$, the calculations that lead to $\mathbf{K}_{i-1}$ are assumed.

Chains can also branch. Suppose the first part t_1 in a chain which carries its translational and scaling degrees of freedom is called the *head*, and the other parts which only have an angular degree of freedom are called *tails*. One chain $C_a = (t_{a_1}, \ldots, t_{a_{T_a}})$ can be attached to another $C_b = (t_{b_1}, \ldots, t_{b_{T_b}})$ at part t_{b_i} along C_b's length by converting t_{a_1} to a tail and redefining C_a as $C_a = (t_{b_1}, \ldots, t_{b_i}, t_{a_1}, \ldots, t_{T_a})$.

6.1.3 Depth constraints

Another useful kind of constraint is related to depth. When there is an expectation that some subset of the objects being tracked will not occlude one another, we can collect them into a *depth group*. Objects in the same depth group are not masked against one another during computation of the joint image likelihood. When justified, grouping objects in this way is more efficient because there are fewer depth orderings to consider for each joint measurement.

An obvious situation to which depth groups apply occurs when tracking an object with multiple attributes. Since attributes represent qualities of a physical object rather than the object itself, multiple instances can be “layered” onto a single object without affecting the visibility of any of them. When a person’s face, for example, is tracked by both a textured region tracker (to capture appearance) and a homogeneous region tracker (for skin color), the two trackers are members of the same depth group.

Depth groups are also appropriate for parts linked by constraints under certain viewing and motion conditions. Though these parts are spatially distinct, if they are physically prevented from overlapping they can also be placed in the same depth group. For example, consider a person’s arm viewed in profile as it moves parallel to the image plane. Considering the upper arm and forearm as two parts tracked using any modality, the joint limits of the elbow allow at most negligible overlap due to depth, and thus we can ignore this interaction. The depth-independent interactions between parts that are illustrated in Figure 5.7 still apply to parts in the same depth group, however. When two parts abut each other, even if neither is occluded there is still a change in the background along some portion of their perimeters and thus a change in each part’s expectations about color contrast and edge-finding.

6.2 Results

Though a rigid link is a fairly simple constraint, it can be used to good effect, as the first two examples demonstrate.

In Figure 6.3, we want to track a white pawn in a visual environment that contains a similarly-colored object (a white rook) and a similarly-shaped one (a black pawn). One obvious avenue is to try to track the pawn by color. We use a Joint Likelihood Filter tracker consisting solely of a homogeneous region initialized as shown in frame 0 of Figure 6.3(d).[1] There is negligible scaling or rotation and movement is slow,

[1] A single-object JLF is not the same as a standard PDAF tracker because of the way match values are used in the computation of the joint image likelihood $p^J(\mathbf{I} \mid \mathbf{X}^J)$, but we use the JLF

so we let the state be $\mathbf{X} = (x, y)$, making measurement space $\mathcal{Z} = X \times Y$. The homogeneous region tracker selects the single most likely of 50 samples using a state sampling covariance of $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 50 & 0 \\ 0 & 50 \end{smallmatrix}\right)$ and improves it with Powell's method.

This approach does not work, as illustrated in the frame sequence in Figure 6.3(d), because the untracked white knight fits the color model well and attracts the pawn strongly. The fundamental problem is the presence of a strong, persistent peak due to the knight in the homogeneous region's image likelihood, depicted at frame 0 in Figure 6.3(a), that is not expected by the Joint Likelihood Filter tracker. If the knight were also tracked, as was the case with the two-object example in Figure 5.10 from the previous chapter, then the Joint Likelihood Filter would prevent mistracking caused by multiple peaks in the likelihood.

Tracking the pawn in a similar fashion with a snake alone yields better results: the pawn is rarely mistracked, but there is some noise in the state estimation due to transient distractions caused by pawn-like arrangements of edges, such as between the fingers. This improved performance can be predicted from the image likelihood for the snake $p_{snake}(\mathbf{I} \,|\, \mathbf{X})$, represented as an image for $\mathbf{I}$ =frame 0 in Figure 6.3(b). We can conveniently draw the likelihood as an image for this example because the dimensions and limits of $\mathcal{Z}$ correspond exactly to the image width and height. The intensity $I(x, y)$ of each pixel of the likelihood image is drawn according to the function $I(x, y) = 255 * p_{snake}(\mathbf{I} \,|\, (x, y))$. The snake image likelihood has many more maxima than that of the homogeneous region, but none are nearly as high as the one corresponding to the snake. This quantifies our intuition that shape is a better cue for this task than color.

Without knowing ahead of time which modality, if any, is sufficiently distinctive for successful tracking, a prudent strategy is to use multiple attributes simultane-

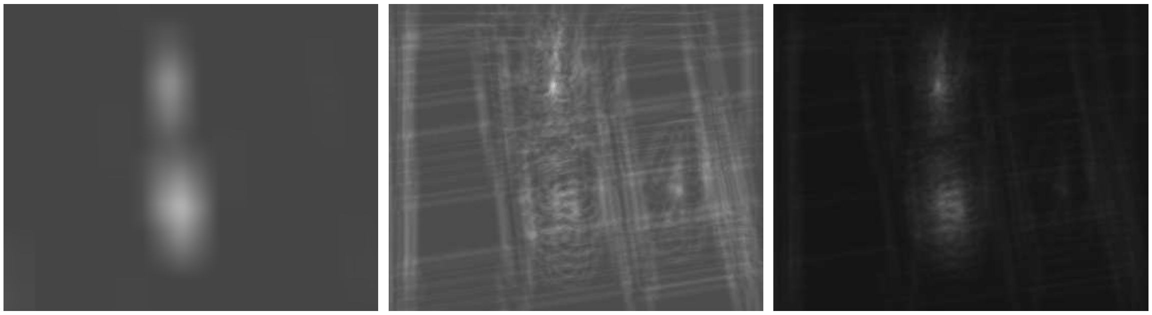
here to make comparisons with the CJLF clearer.

ously. The conjunction of color and shape results in a joint image likelihood $p^J(\mathbf{I} \,|\, \mathbf{X}^J)$ with peaks only where *both* likelihoods $p_{hregion}(\mathbf{I} \mid \mathbf{X})$ and $p_{snake}(\mathbf{I} \mid \mathbf{X})$ have peaks. This often reduces distractions, as can be seen in the representation of $p^J(\mathbf{I} \,|\, \mathbf{X}^J)$ in Figure 6.3(c).

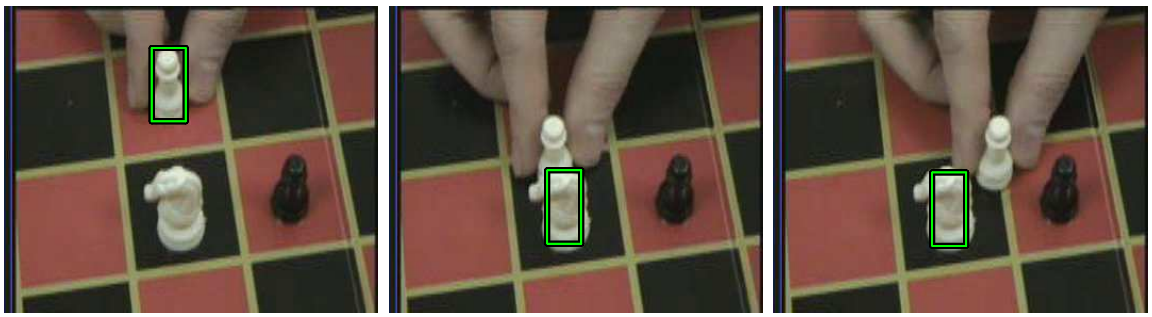
Formally, we can utilize the pawn's color and shape simultaneously by modeling the chess piece with two linked attributes—a homogeneous region and a snake—with the constraint that the centers of the region and snake be coincident. Arbitrarily, we let the homogeneous region tracker contain the state $\mathbf{X}_1 = (x_1, y_1)$, making its measurement space $\mathcal{Z} = X \times Y$. Its measurement key is the same as that of the first object in the rigid link example above, without scaling: $\mathbf{K}_1 = (x_1, y_1, \bar{w}_1, \bar{h}_1, \bar{\phi}_1)^T$. The snake tracker has no state and an empty measurement space; its measurement key is simple because the link has a length of 0: $\mathbf{K}_2 = (x_1 - \hat{\mathbf{\Lambda}}_x, y_1 - \hat{\mathbf{\Lambda}}_y, \bar{q}_2, \bar{q}_3, \bar{q}_4, \bar{q}_5)^T$. Both trackers are in the same depth group.

The pawn's joint region-snake tracker follows the same regime of hill-climbing on the single best of 50 samples as the single-attribute trackers above. As Figure 6.3 shows, this constrained formulation permits the pawn to be successfully tracked when the homogeneous region alone fails. We have also observed that the estimated state of the pawn is less erratic using conjoined trackers than is obtained by using the snake tracker alone.

Another example of tracking with the CJLF is given in Figure 6.4. In the input sequence, a person walks from the left side of the frame slightly toward the camera and then in profile to the right. Suppose we want to track the person's face as a homogeneous region with a single-part Joint Likelihood Filter tracker. Let the state be $\mathbf{X} = (x, y, \dot{x}, \dot{y}, s)$ because the face translates relatively quickly and scales gradually but significantly from the first frame to the last. Measurement space is $\mathcal{Z} = X \times Y \times S$; the best single sample of 50 is improved using Powell's method,

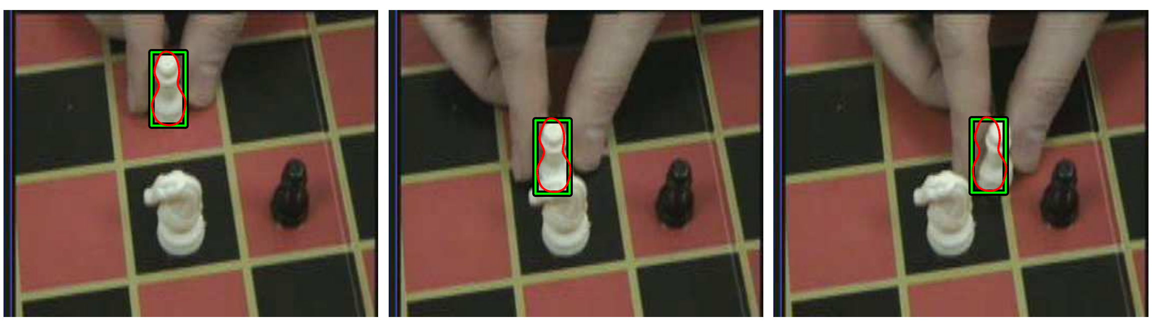


(a) (b) (c)



0 50 100

(d) One-attribute JLF tracker



0 50 100

(e) Two-attribute CJLF tracker

Figure 6.3: Multi-attribute Constrained Joint Likelihood Filter (MPEG). (a) $p_{hregion}(\mathbf{I} \mid \mathbf{X})$ for homogeneous region; (b) $p_{hregion}(\mathbf{I} \mid \mathbf{X})$ for snake; (c) $p^J(\mathbf{I} \mid \mathbf{X}^J)$ for both; (d) A homogeneous region JLF tracker is distracted by the white knight; (e) A CJLF homogeneous region and snake tracker overcomes the distraction.

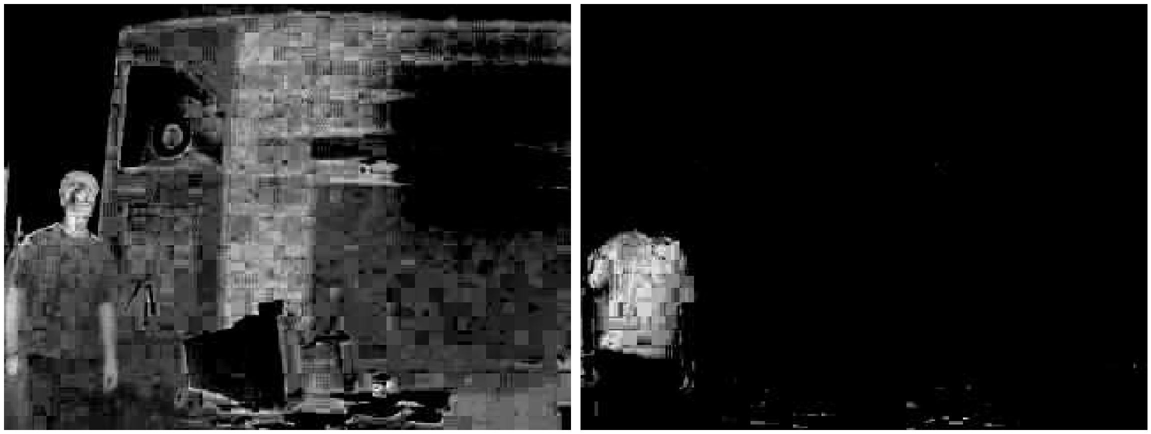
where $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 0.001 \end{smallmatrix}\right)$.

A sequence of frames from one run of this tracker is shown in Figure 6.4(c). Because of a somewhat skin-colored brick wall in the background, poor lighting, and a suboptimal skin color model, the discriminatory power of the face tracker is very marginal. This can be seen in Figure 6.4(a), which shows the color similarity γ_{face} of frame 0 to the face. The face tracker is distracted by a column of tan bricks in the center of the image; when the person emerges on the other side of the bricks, tracking has failed. This occurs for essentially the same reason as with the pawn tracking example above: the bricks are unmodeled, very similar to the target, and in close proximity to it for too many frames.

A tracker with the same filter parameters can track the red shirt through the same sequence without any problems, however. Figure 6.4(b), depicting γ_{shirt} for frame 0, shows why the color of the shirt is a much more distinctive cue than the face's color in this visual environment. The shirt is not a different attribute of the face like shape and color were for the pawn, but rather a different part of the person's upper body. This suggests that we can improve the face tracker's performance by exploiting its physical connection to the shirt with a two-part Constrained Joint Likelihood Filter tracker.

We impose the constraint between the face and shirt as a rigid link that scales with the two parts but does not rotate (since they do not). Letting the face tracker contain the state $\mathbf{X}_1 = (x_1, y_1, \dot{x}_1, \dot{y}_1, s_1))$ makes its measurement space $\mathcal{Z}_1 = X \times Y \times S$. The measurement key of the face tracker is $\mathbf{K}_1 = (x_1, y_1, s_1\bar{w}_1, s_1\bar{h}_1, \bar{\phi}_1)^T$. The shirt tracker has no state and an empty measurement space; its measurement key is $\mathbf{K}_2 = (x_1 + s_1\delta_x, y_1 + s_1\delta_y, s_1\bar{w}_2, s_1\bar{h}_2, \bar{\phi}_2)^T$, where $\delta_x = \bar{x}_2 - \bar{x}_1$ and $\delta_y = \bar{y}_2 - \bar{y}_1$. The trackers are put in the same depth group.

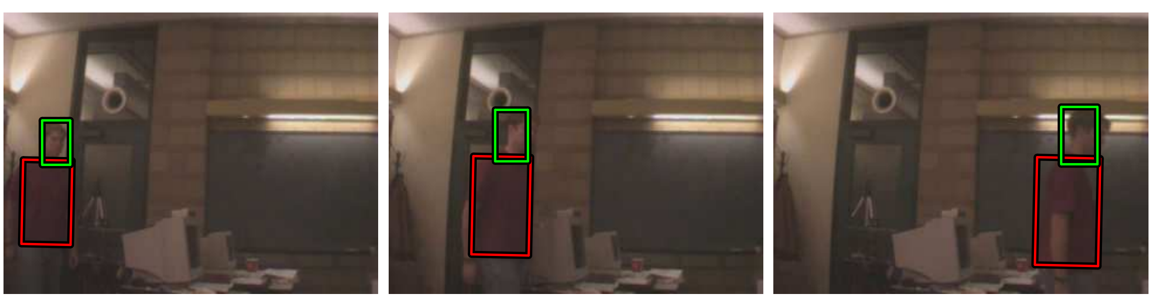
Figure 6.4(d) demonstrates that linking the shirt and face tracker together in



(a) (b)

0 60 120

(c) One-part JLF tracker



0 60 120

(d) Two-part CJLF tracker

Figure 6.4: Multi-part Constrained Joint Likelihood Filter: Resisting a distracting background (MPEG). (a) Face color match γ_{face}; (b) Shirt color match γ_{shirt}; (c) One-part Joint Likelihood Filter tracker on the face is distracted; (b) Two-part Constrained Joint Likelihood Filter tracker on the face and shirt succeeds.

this manner overcomes a distractingly face-colored background. The CJLF tracker sometimes bobbles slightly as the face passes in front of the brick column because the tracker briefly explores the possibility of not translating anymore and instead simply expanding to including the bricks, the face, and the shirt. This part of measurement space is quickly discarded, however, as the proportion of non-matches in the larger area dilutes its fitness compared to the correct interpretation. Phenomena such as these are another reason why pure gradient ascent tracking is not always workable: when used here, the tracker gets stuck in a local maximum of state space that corresponds to expanding ceaselessly and cannot jump to the better alternative as random sampling does.

We should also note that the face tracker is not just "along for the ride" as the shirt tracker smoothly proceeds. Though the shirt tracker may be the more valuable partner, the joint image likelihood ensures that both components contribute to the process equally (i.e., without regard to area, since mean match values are used). Indeed, because the rigid constraint without rotation is necessarily an approximation, the face often forces the joint state estimate to a compromise scale and position that best includes both regions, rather than fitting the shirt region with precision and deriving the face region estimate afterwards.

A more complicated situation which shows the advantage of the Constrained Joint Likelihood Filter over the Joint Likelihood Filter is shown in Figure 6.5. Here we want to track a person's hand and forearm as homogeneous regions while they shake hands with another person, who is not tracked. To account for quickly changing position and angle, each component ($i = 1, 2$) of the Joint Likelihood Filter tracker has a state of the same form: $\mathbf{X}_i = (x_i, y_i, \phi_i, \dot{x}_i, \dot{y}_i, \dot{\phi}_i)$, making their measurement spaces $\mathcal{Z}_1 = \mathcal{Z}_2 = X \times Y \times \Phi$. Accelerations during the handshake are too large for pure gradient tracking, so each component tracker selects the best 1 of 50 samples, where the state sampling covariance is $\mathbf{\Sigma}_{\mathcal{X}} = \left(\begin{smallmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 0.02 \end{smallmatrix}\right)$. This sample is then improved using Powell's method for gradient ascent on the joint image likelihood function. The match threshold for the homogeneous regions here is $\Upsilon_{hregion} = 2$. Despite these measures, the hand tracker mistracks when its target is in close proximity to the other person's hands (which are not being tracked) due to distraction. The forearm tracker wanders up and down the sleeve because there is no reason for it to remain fixed at one end.

These shortcomings can be eliminated by introducing the constraint that there is a hinge (i.e., the wrist) joining the hand and forearm trackers to one another at the midpoints of their abutting short sides. Formally, the state of the forearm tracker remains $\mathbf{X}_1 = (x_1, y_1, \phi_1, \dot{x}_1, \dot{y}_1, \dot{\phi}_1)$ and its measurement space is also $\mathcal{Z}_1 = X \times Y \times \Phi$. Its measurement key is the same as that of the first object in the hinge example above, without scaling: $\mathbf{K}_1 = (x_1, y_1, \bar{w}_1, \bar{h}_1, \phi_1)^T$, and the sampling covariance is also $\mathbf{\Sigma}_{\mathcal{X}_1} = \left(\begin{smallmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 0.02 \end{smallmatrix}\right)$. The hand tracker, however, has only one degree of freedom, and its state is just $\mathbf{X}_2 = (\phi_2, \dot{\phi}_2)$, reducing the measurement space to $\mathcal{Z}_2 = \Phi$. Thus, the sampling covariance for the hand is $\mathbf{\Sigma}_{\mathcal{X}_2} = \left(\begin{smallmatrix} 0.02 \end{smallmatrix}\right)$. The hinge angle is initially 0, the hand and forearm have the same initial orientation, and there is no scaling, simplifying the form of the hand's measurement key considerably compared to Equation 6.3 to yield:

$$\mathbf{K}_2 = \begin{pmatrix} x_1 - \frac{1}{2}\bar{w}_1 \cos(\phi_1) - \frac{1}{2}\bar{w}_2 \cos(\phi_1 + \phi_2) \\ y_1 - \frac{1}{2}\bar{w}_1 \sin(\phi_1) - \frac{1}{2}\bar{w}_2 \sin(\phi_1 + \phi_2) \\ \bar{w}_2 \\ \bar{h}_2 \\ \phi_1 + \phi_2 \end{pmatrix} \tag{6.4}$$

where the hand and forearm are also considered to be in the same depth group.

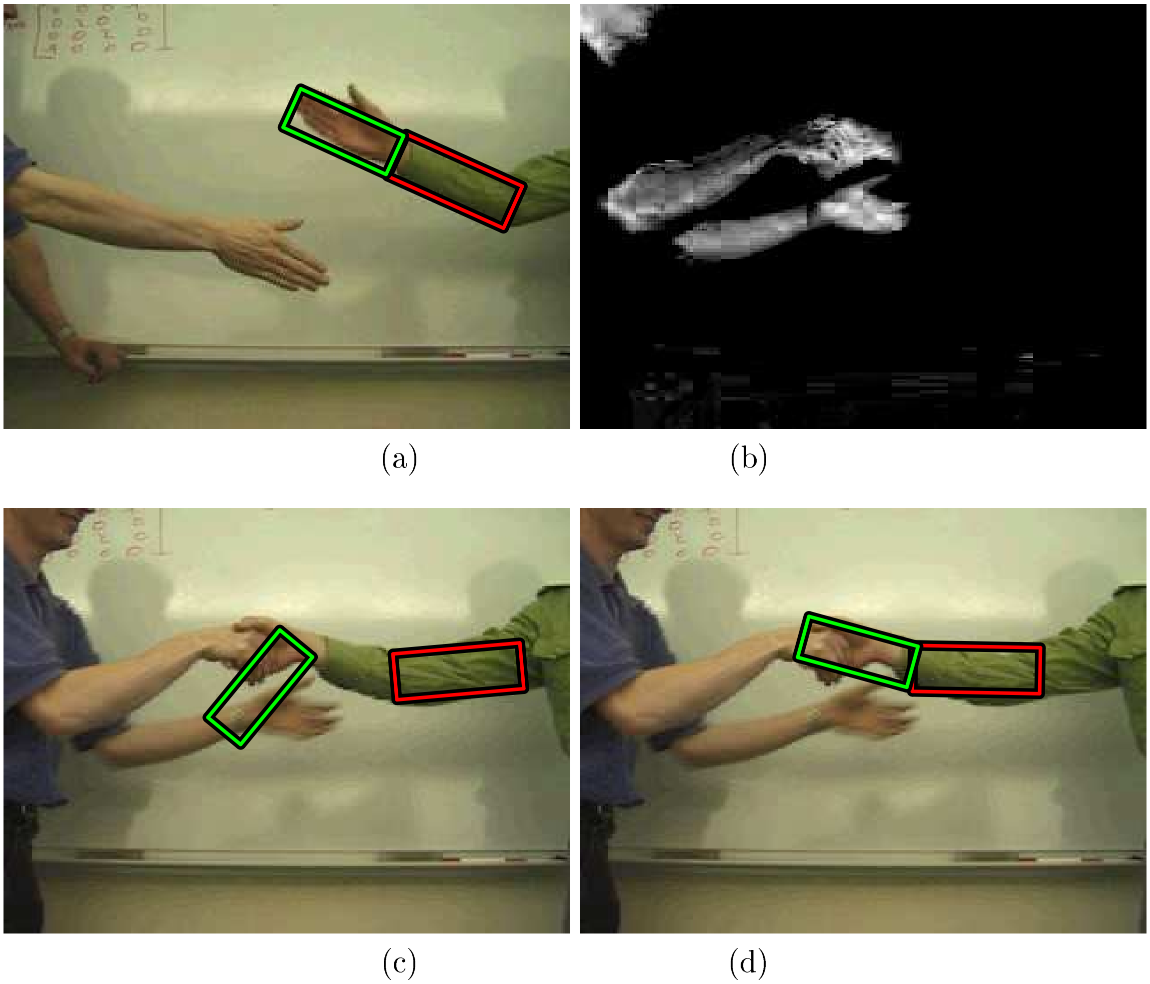


Figure 6.5: Constrained joint likelihood: Using a hinge constraint at the wrist to prevent mistracking during a handshake (MPEG). (a) Frame 0 of sequence of homogeneous region trackers on hand and forearm; (b) A distracting situation: hand color similarity γ at frame 260; (c) Running a Joint Likelihood Filter tracker for both parts, the hand tracker is distracted by other person's hand (frame 260); (c) The Constrained Joint Likelihood Filter formulation permits accurate tracking of the hand (frame 260).

Adopting this approach prevents the hand and forearm trackers from floating apart; relatively higher joint image likelihoods keep the hinge at the sleeve-hand border. The result is that during the period of ambiguity when the two hands are clasped together, a realistic interpretation of the situation is maintained and tracking proceeds correctly after the hands are fully separated.

Another task to which the CJLF is well suited is illustrated in Figure 6.6. In this case we want to track a person's whole arm, from shoulder to fingertips, as they raise their hand from a computer mouse to their face and lower it again. There is articulation at the elbow and wrist, and the person's face will not be tracked and is therefore an unmodeled distractor. Since the person is wearing a short-sleeved shirt, we divide the arm into four areas and assign each a tracker: the sleeve ($i = 1$), the exposed skin of the upper arm ($i = 2$), the forearm ($i = 3$), and the hand ($i = 4$). The wrinkles in the fabric of the sleeve provide good texture, so it will be tracked by a textured region; the other areas are tracked by homogeneous regions. Since the movement is nearly all in a plane parallel to the image plane, we neglect scaling and focus on translation and rotation.

Using the JLF, the state of each tracker is $\mathbf{X}_i = (x_i, y_i, \phi_i, \dot{x}_i, \dot{y}_i, \dot{\phi}_i)$, making each component's measurement space $\mathcal{Z}_i = X \times Y \times \Phi$. Powell's method is used for pure gradient ascent on the joint image likelihood, with the best of all visibility-affecting depth orderings serving as the starting point for hill-climbing for each frame. The initial arrangement of the trackers is shown in Figure 6.6(b), frame 0. This method is able to follow the movement of the arm in a gross sense, but the degrees of freedom of each tracker permit considerable shifting and rotational variability of the homogeneous region trackers along the arm. In particular, the upper arm tracker does not maintain the correct position and the hand tracker is severely distracted by the similarly-colored face as they separate (see Figure 6.6(a)). The textured region sleeve tracker performs well throughout the sequence.

The CJLF mitigates these problems to a large extent by exploiting the additional information available about the connectivity of the different areas of the arm. These constraints reduce the degrees of freedom of each tracker and effectively boost the performance of marginal trackers by deriving their state information from better performing trackers to which they are linked. Under the CJLF, only the state of the textured region sleeve tracker is unchanged: $\mathbf{X}_1 = (x_1, y_1, \phi_1, \dot{x}_1, \dot{y}_1, \dot{\phi}_1)$. Its measurement space is $\mathcal{Z}_1 = X \times Y \times \Phi$, and its measurement key is the same as that of the forearm in the handshaking example: $\mathbf{K}_1 = (x_1, y_1, \bar{w}_1, \bar{h}_1, \phi_1)^T$. The upper arm tracker is rigidly linked to the sleeve tracker, so it has no state or measurement space *per se*; its measurement key reflects the rigid constraint:

$$\mathbf{K}_2 = \begin{pmatrix} x_1 + b\cos(\phi_1) - \frac{1}{2}(\bar{h}_1 + \bar{h}_2)\sin(\phi_1) \\ y_1 + b\sin(\phi_1) + \frac{1}{2}(\bar{h}_1 + \bar{h}_2)\cos(\phi_1) \\ \bar{w}_2 \\ \bar{h}_2 \\ \phi_1 \end{pmatrix} \tag{6.5}$$

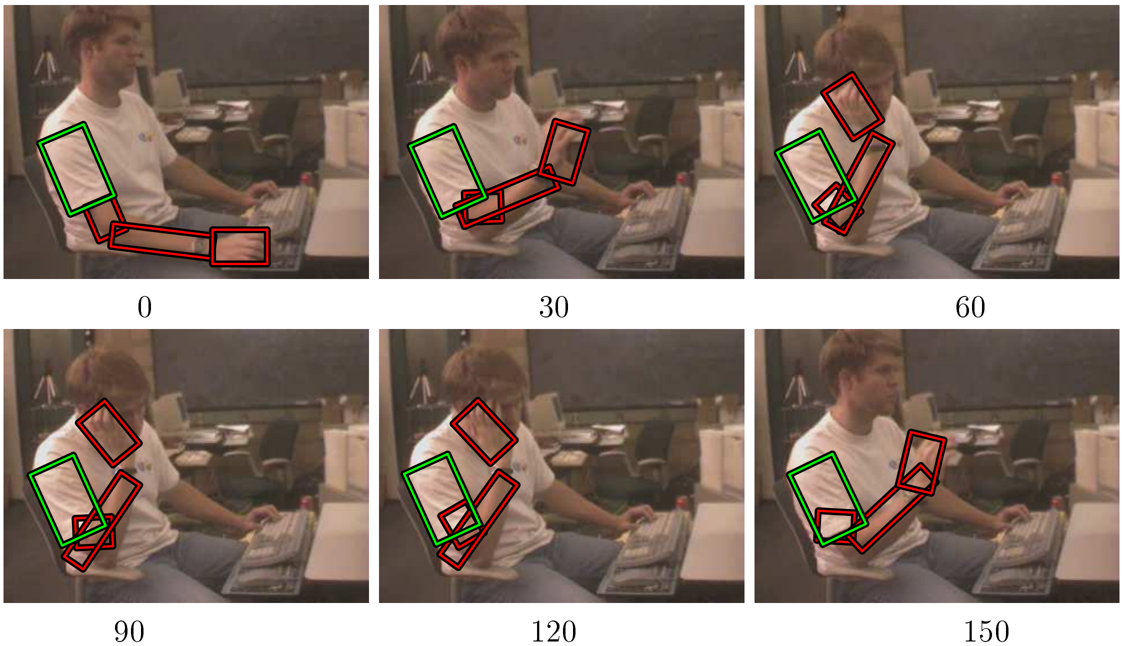
where $b = 6$ is an offset reflecting the fact that the sleeve hangs slightly below the arm. A compressed expression for the upper arm measurement key is given by $\mathbf{K}_2 = R_{1,2}(\mathbf{K}_1)$.

The forearm tracker is linked to the upper arm tracker via a hinge constraint at the elbow, so its only free parameter is a relative angle. This makes its state $\mathbf{X}_3 = (\phi_3, \dot{\phi}_3)$ and measurement space $\mathcal{Z}_3 = \Phi$. The exact location of the hinge is at the midpoints of the abutting ends of the upper arm and forearm rectangles; we forego the geometric derivation and define the forearm tracker's measurement key as $\mathbf{K}_3 = H_{2,3}(\mathbf{K}_2)$. Similarly, the hand is linked to the forearm by another hinge. Its state is $\mathbf{X}_4 = (\phi_4, \dot{\phi}_4)$ and its measurement space is $\mathcal{Z}_4 = \Phi$. The wrist hinge is located at the midpoints of the abutting ends of the forearm and hand rectangles, making its measurement key $\mathbf{K}_4 = H_{3,4}(\mathbf{K}_3)$

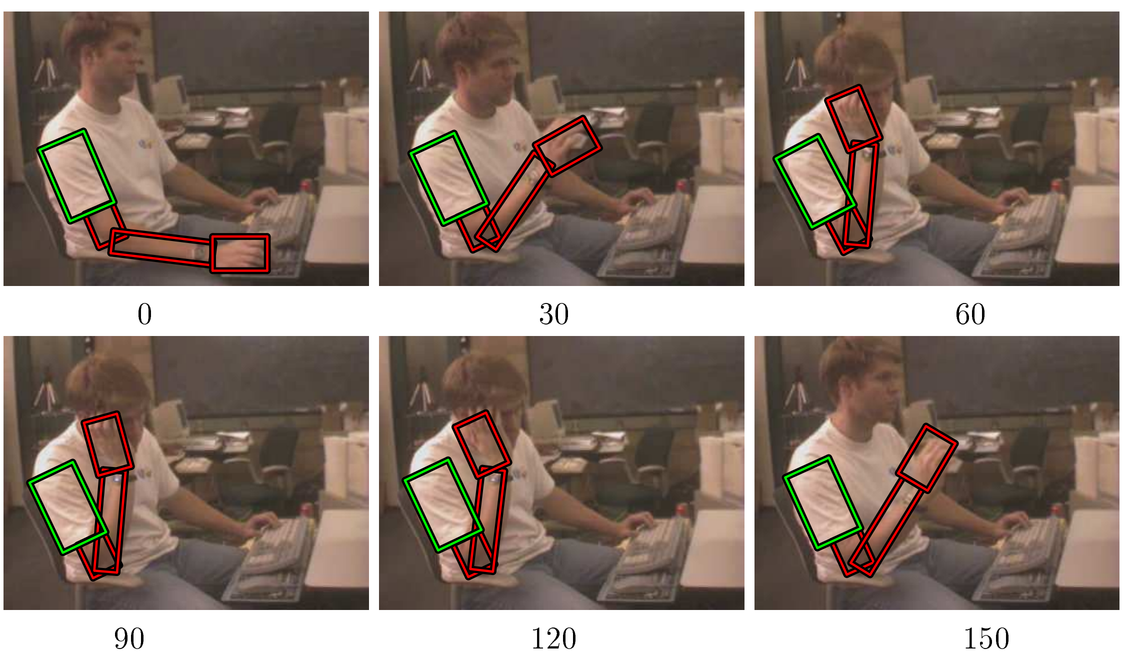
The match threshold of the sleeve tracker is $\Upsilon_{tregion} = 30$ and for the homogeneous regions it is $\Upsilon_{hregion} = 2$. All of the arm parts are in the same depth group because of the viewing angle.

As can be seen from Figure 6.6(c), the mixture of rigid and hinge constraints between the four parts of the arm considerably improves tracking performance. The parts maintain their relative positions and angles along the arm quite well, and the untracked face causes no appreciable problems throughout its overlap with the similarly-colored hand.

(a) Arm color similarity γ



(b) Joint Likelihood Filter



(c) Constrained Joint Likelihood Filter

Figure 6.6: Tracking an arm in four sections (MPEG). (a) Face is distracting to hand tracker (frame 130); (b) Frame sequence using JLF; (c) Frame sequence using CJLF.

Chapter 7

Related Work

In this chapter we discuss previous work on tracking and its relationship to the framework presented in this dissertation. In the first section we cover other approaches to tracking single objects, including the modalities used and the underlying state update algorithms. The second section surveys other work on jointly tracking multiple objects and how it differs from ours. Finally, we summarize previous research on combining different modalities and adding constraints between trackers in order to achieve more robust tracking.

7.1 Single Object Tracking

7.1.1 Modalities

The image cues—color, texture, motion, shape, depth, and so on—that have been used for tracking are quite varied. Typically, a single attribute is used to discriminate an object from the rest of the scene, making the selection of an identifying attribute quite important in determining the performance of the tracker. In this section we examine vision research related to color, texture, and shape as tracking modalities, as well as other currently popular and possible future cues.

Color

There have been numerous approaches to color representation for tracking and search.

Swain's color histogram [114] is one popular method. The color components of the pixels within an object are histogrammed; a *histogram intersection* can be computed between this histogram and an area of the image to test the similarity of the two. A useful feature of this algorithm is its ability to define an object as consisting of multiple colors in certain proportions (e.g., $\frac{3}{4}$ red and $\frac{1}{4}$ blue). However, there is no facility for specifying their relative geometric distributions (e.g., red above blue).

For example, Birchfield's elliptical head tracker uses color histograms [11]. Color space is a transformation of RGB space that encodes chrominance in two parameters $G - R$ and $B - G$ of eight bins each and luminance in the third, $R + G + B$, with four bins. The degree of match between the pixels in a postulated ellipse location and the histogram model is given by the histogram intersection formula in [114].

In the same vein, Bradski tracks a person's face in [19] with a window that is translated and scaled to maximize the fit of the pixels it contains to the face color model. This model is built by sampling skin-colored pixels and making a histogram of their hues (the first component in HSV space); image pixels whose hues fall into high-content bins are assigned proportionally high probabilities of being skin-colored. Pixels with low intensities or saturations have very noisy hues, so these are ignored.

Fleck *et al.* explicate a search procedure that relies on color and geometry to determine whether there are naked people in images in [42]. Skin-colored pixels are segmented with ranges in hue and saturation values after a log-opponent transformation of RGB space adapted from [46]. Darrell *et al.* use a similar log-opponent color representation for face tracking in [31].

The Pfinder system [122] models parts of a person's body such as their face, hands, and shirt with 5-D Gaussian ellipsoids, or *blobs*: a 2-D image spatial component plus a 3-D color component in YUV space. Each background pixel is modeled with a 3-D Gaussian in YUV space. A special step attempts to detect whether part of a blob is in shadow, and if so, the color components are normalized by intensity as follows: $U^* = U/Y$, $V^* = V/Y$.

Similarly, [20] tracks blobs as Gaussian ellipsoids with a 2-D spatial component, an optical flow component comprising either translation and rotation or affine motion, and optionally a 2-D color component in HSV space.

Stauffer and Grimson [113] adaptively model individual background pixels with a mixture of Gaussians in RGB or HSV space. They use a Gaussian not for modeling the variation in color over the extent of an object, but rather its variation over time at a single image location due to sampling noise. The Gaussian mixture accounts for such phenomena as trees swaying, light glinting, monitors flickering, etc. that can result in multiple contributions to a pixel's color.

SSD

Early work on image correlation for tracking was reported in [1, 80]. Shi and Tomasi refine this approach and describe a technique for finding textured image patches and tracking their affine motion in [109]. There is considerable overlap between this SSD work and the motion estimation research typified by [9].

Hager and Belhumeur exhibit a solution in [51] to the problem of SSD tracking when there are variations in object pose and in the contrast, intensity, and the direction of the light source. The method assumes that the surface of the object being tracked is rigid, Lambertian, and that there is no self-shadowing.

A larger set of influences on image appearance are considered in [12]. Image change is explained as a mixture of causes such as motion (including non-rigid), illumination variation, specularities, and iconic changes (object-specific occlusions and material changes such as teeth appearing and disappearing as a person's mouth opens and closes). The EM algorithm is used to determine the contributions of the various models. Results are only given for the three possible pairs of models that include motion.

The *active blob* method of [106] defines an object model in terms of a deformable 3-D mesh to capture shape and a color texture map for its appearance. The object may bend and twist as it moves, and global intensity and contrast variations are compensated for. A Lorentzian influence function [14] replaces the standard quadratic SSD error norm in order to achieve more robust tracking in the presence of outliers caused by specularities and small occlusions. Robust statistics are also discussed for SSD tracking in [51]. Though we have not adopted them here, robust statistical formulations of the image likelihoods $p(\mathbf{I} \mid \mathbf{X})$ for all three of our modalities would likely improve their tracking performance in certain circumstances.

Contours

An early formulation of the contour tracking problem is given by Kass *et al.* in [72] using energy functionals. Under their definition, a snake is a spline whose shape is determined by the sum of three forces: an internal force that governs the smoothness of the curve, an external force that guides the initial placement of the snake, and an image force that attracts the snake to edge and line features. Intensity and the gradient are two quantities that contribute to the image force. The method is applied primarily to fitting shapes in single images, but an example of lip tracking is also given.

A somewhat different approach to snakes is taken by Blake *et al.* in [16, 18]. Their

B-spline representation permits a detailed description of the curve's shape, which they call a *template*. However, only low-dimensional deformations of the template (e.g., an affine transformation) are allowed to represent the evolving state of the curve after initialization. The constraints on permissible shapes are derived for various assumptions about the curve and its motion, such as planarity, in-plane motion, etc. Reducing the size of the shape space avoids unconstrained movement of the B-spline control points, which can often result in mistracking. With tuned priors on object dynamics, this technique demonstrates robust, real-time tracking of fast-moving hands and lips [17].

In the same spirit, an ellipse with a fixed aspect ratio serves as a shape model for tracking a person's head in [11]. The ellipse is allowed to rotate, scale, and translate in order to maximize an edge match formula that favors a large gradient magnitude at each point along the oval's perimeter in conjunction with a gradient direction that is also normal to the curve at that point.

A learned subspace model for contour tracking is presented in [26] and [8]. They use principal components analysis (PCA) on a set of outlines of objects to compute their range of motion and use this to parametrize the possible curve shapes during tracking. This approach is applied to B-splines to track the silhouettes of walking people in [8].

Other modalities

There are many other potential modalities suitable for use in tracking besides color, appearance, and shape. In this section we review a few of the most frequently used.

Perhaps one of the most popular cues for tracking is motion. With a fixed camera, simple differencing between successive frames is sufficient to localize motion and hence serve as a basis for segmentation in the same way that we use γ for color.

The target may stop moving from time to time, though; one way around this is to compare the current frame to a reference image without the target in it (i.e., the background). Pfinder [122] computes a per-pixel background color model that is a vital component in localizing the tracked person. Background models are also used for person tracking in [103] and for person and car detection and tracking in [113].

Going further, one can also use optical flow to extract directional information. Cutler and Turk [30] compute optical flow and segment it into blobs to classify waving, clapping, flapping, and other motions of a person's hands. When the camera itself is in motion, more sophisticated techniques that compensate for egomotion must be used. A possible precursor to motion-based tracking is an algorithm for computing a *layered* representation of multiple motions in a scene, including that of the background, given in [105]. Irani and Anandan describe a system along these lines for detecting and tracking moving objects while moving in [58].

Stereo or other depth cues are another obvious candidate for use in tracking. A stereo version of Pfinder correlates color blobs from two cameras for depth estimation [3]. Darrell *et al.* segment faces by intensity from a disparity map computed by a hardware stereo solution [31]. Specialized hardware such as Kanade's machine for computing a dense depth map at frame rate [70] and Konolige's low-cost board for obtaining a somewhat cruder depth estimate [77] have recently made this modality much more practical.

Indeed, a frequent obstacle to adopting otherwise promising cues as tracking modalities is the computational expense associated with them. Virtually any method that segments an image into regions sharing some common visual property or searches for matches in an image database could potentially be the basis of a tracking algorithm. Though there are a number of diverse segmentation and edge detection algorithms that yield excellent results, many take on the order of hours or more for each image, rendering them unusable for real-time tasks. Nonetheless, as these algorithms are coded more efficiently or fast approximations are found for them, and as computing power continues to increase geometrically, they could soon find their way into a tracking framework such as ours.

7.1.2 Tracking methods

There are a few standard ways that modalities have typically been used for tracking. Historically, a common method has been to define a modality-specific error function that measures the degree of match between a hypothesized object state and the image evidence and do gradient ascent or descent on it. A related but distinct approach is to frame the tracked object parameters as the solution to a set of equations, leading to another set of iterative methods for zero-finding. When the system of equations is overconstrained, least-squares techniques are employed [94]. Many researchers have found it advantageous to extend this basic approach with a Kalman or similar filter. A Kalman filter smooths the state estimate in the face of noisy data and allows the addition of dynamics (velocity, acceleration, etc.) to the state for better predictive power. More recently, there has been considerable interest in the Condensation algorithm [59], a randomized estimation procedure that exhibits good performance when there is visual clutter that may distract a standard Kalman filter. In this section we discuss these three tracking paradigms and their relationship to the framework presented in this dissertation.

The interested reader is also referred to a number of other randomized optimization methods such as simulated annealing [87], RANSAC [41], and the genetic algorithm [47]. They might also prove adaptable to tracking, though thus far there has been little work along these lines.

Iterative algorithms

Kass *et al.*'s energy minimization method for contour tracking [72] is an example of gradient ascent (or rather, descent) for tracking. For each new image, the energy terms give rise to a system of equations that is solved using LU decomposition to yield a new parametrization of the snake's shape.

SSD [51] is an iterative method that assumes small motions between frames. This allows a Taylor series approximation of the error norm $\frac{1}{|R|}\sum_{x,y \in R}(\hat{\mathbf{I}}(x,y) - \mathbf{I}(x,y))^2$ that leads to a system of linear equations. Solving this system yields the motion parameters that take the pixels in region R from the reference image to the image $\hat{\mathbf{I}}$ that is best registered with the current image $\mathbf{I}$. Other research on motion estimation methods relying on image correlation (such as [9]) often incorporates a coarse-to-fine image pyramid to handle larger motions and allows multiple iterations of gradient ascent before convergence.

A robust error norm quantifies the degree of fit between the image and a warped template for the active blob tracker in [106]. The Levenberg-Marquardt method [94] is employed to compute the values of the deformation and photometric variables that minimize this error. These new values become the tracker's state in the next frame.

Bradski's face tracker [19] and an early version of our color tracking work [95] are based on pure gradient ascent. The similarity to a particular color of the pixels within a rectangular tracking window centered at $\mathbf{p}_i$ is measured; let the center of mass of these pixels (weighting more similar pixels more heavily) be $\hat{\mathbf{p}}_i$. If the image distance $|\mathbf{p}_i - \hat{\mathbf{p}}_i| > \epsilon$ for some threshold ϵ, $\mathbf{p}_{i+1}$ is set to $\hat{\mathbf{p}}_i$ and the process is repeated until convergence on the same image. Within the tracking window the moments of the color similarity distribution are computed to estimate the area and angle of the tracked object, optionally guiding the size of the tracking window.

Birchfield's head tracker [11] elaborates somewhat on this approach. The tracker uses an ad-hoc scheme for velocity estimation, and within some constant size neighborhood of the predicted head configuration (including location, scale, and orientation) hypothesizes and tests possible head configurations. The highest-scoring configuration in this neighborhood becomes the new state of the tracker.

For generality, we have chosen to use conjugate gradient or Powell's method for gradient ascent on the image likelihood and joint image likelihood. A modality-specific analysis like that carried out in [51, 106] may yield a faster technique for a particular objective function, but employing a common method for all modalities makes adding new ones much easier, and we have found that only a modest number of iterations of gradient ascent are usually sufficient to significantly improve a state sample.

Kalman filtering

Pfinder [122] does not use a Kalman filter, but it updates each blob's parameters by blending the second-order statistics of the pixels matched to it with prior knowledge and an approximate dynamical model. The procedure for doing this is not detailed.

Snakes are placed in a Kalman filtering framework in [117]. This allows dynamics to be part of the state estimation process, as opposed to the purely gradient-driven approach of [72]. However, the dynamics are chosen somewhat arbitrarily. The snake-tracking systems described in [17, 18, 100], on the other hand, attempt to learn sophisticated dynamical models of their targets' motions from example sequences. With such training the hand trackers in [17, 18] and lip tracker in [17] are able to follow certain agile motions than untuned trackers cannot, as well as less susceptible to mistracking due to background distractors.

In this dissertation we have not focused on building elaborate dynamical models because they are often quite target-specific—the dynamics of a hand waving do not apply to a knee bending, etc. We assert, and the authors of [18] concede, that this can restrict the ease of reuse and range of applicability of a tracking algorithm. Rather, we have strived to create a framework that can handle multiple classes of objects (cars, people, chess pieces, airplanes, etc.) undergoing a variety of motions. Notwithstanding the simplicity of the dynamics we do use, we claim that the random sampling method for measurement generation outlined in Chapter 4 yields robust performance in the presence of quick accelerations and direction changes. Dealing successfully with such phenomena is a large part of what tuned dynamical models are designed to handle.

Another reason why we have not used detailed dynamical models is because we wish to emphasize the data association aspect of the tracking problem. Though the papers about them rarely say so explicitly, nearly all of the visual trackers based on the Kalman filter use what is called the nearest-neighbor (NN) method for measurement generation. This follows naturally from the use of gradient ascent to compute a single measurement for each new image. As the discussion in Chapter 4 indicates, this approach is only guaranteed to work when the objective function is unimodal. A multimodal objective function, or image likelihood as we call it, violates the assumptions of such algorithms. The PDAF, JPDAF, and Joint Likelihood Filter (JLF), conversely, acknowledge and deal with multimodality explicitly. Doing so obviates much of the need for highly specific dynamical models which serve to define the target more precisely and thus reduce the chance of distraction. Relying on well-tuned dynamic models is actually similar in spirit to our strategy in Chapter 6 of using conjunctions of attributes and constraints to define the tracked object more distinctively.

Condensation

The Condensation algorithm is a stochastic tracking technique that was introduced in [59] (a more detailed exposition and further extensions are reported in [62, 60, 61, 63]). Its purpose is to facilitate tracking in situations where the image likelihood is non-Gaussian or multimodal. Condensation was originally developed for snake tracking; examples of objects tracked include a person's head and shoulders, the spread fingers of a hand against a cluttered desktop, and a leaf on a bush [60]. Recently, it has been applied to other problem areas such as mobile robot navigation [34] and gesture recognition [13].

The way that the Condensation algorithm works is by approximating the posterior probability of the state $p(\mathbf{X}\,|\,\mathbf{I})$ with a set of weighted samples using the factored sampling algorithm of [49] (henceforth we subscript by t to indicate time). At time step t, let there be N samples $\mathbf{s}_t^{(i)}$ with weights $\pi_t^{(i)}$ such that $\sum_{i=1}^{N} \pi_t^{(i)} = 1$. This set of samples and their weights $\{(\mathbf{s}_t^{(i)}, \pi_t^{(i)})\}$ is obtained from those of the previous time step $\{(\mathbf{s}_{t-1}^{(i)}, \pi_{t-1}^{(i)})\}$ via a stochastic dynamical model which makes a prediction while increasing uncertainty, plus a measurement process which tends to reduce uncertainty. The first step is one of selection: N samples are drawn with replacement from $\{\mathbf{s}_{t-1}^{(i)}\}$, choosing a particular $\mathbf{s}_{t-1}^{(i)}$ with probability $\pi_{t-1}^{(i)}$. This means that some samples may be chosen more than once, and some not at all. Secondly, the predictive step, or *drift*, moves all chosen samples deterministically according to the current estimate of velocity, acceleration, and the like. An increase in uncertainty in the absence of information is simulated by *diffusion*, which moves each sample randomly and independently to its new position $\mathbf{s}_t^{(i)}$. Finally, the measurement step assigns a new weight to each sample by measuring all of their image likelihoods and normalizing: $\pi_t^{(i)} = p(\mathbf{I}_t\,|\,\mathbf{X}_t = \mathbf{s}_t^{(i)}) / \sum_{j=1}^{N} p(\mathbf{I}_t\,|\,\mathbf{X}_t = \mathbf{s}_t^{(j)})$. The samples are initialized before tracking begins with a uniform distribution and equal weights.

If we think of the N samples as computational resources, the function of the selection step becomes clearer. Its tendency is to redistribute these assets so that they become sparser in less promising areas of state space, facilitating a more efficient search. Over time, the normalization of the weights serves a winner-take-all function by concentrating samples in the vicinity of the most likely state, while still occasionally exploring less-likely regions of state space. If the target is becomes completely occluded or leaves the image, the samples diffuse through state space since there is no measurement reinforcement causing them to congregate. Widening the search in this manner is a good strategy for looking for the lost target to reinitialize tracking. Condensation's steady-state tracking and lost-target reacquisition make it a robust algorithm for searching for the global maximum, or MAP state estimate, of the time-varying distribution $p(\mathbf{X} \mid \mathbf{I})$.

In [59], the authors assert that a method such as Condensation is necessary because Kalman filtering is inadequate for tracking in cluttered or distracting visual environments. Though they do not say so explicitly, they seem to be referring to the NN approach to measurement generation for snakes given in [117], which certainly is vulnerable to clutter. An NN Kalman filter considers only one alternative to update the state for each image. As we discussed in Chapter 4, this means that there is a non-zero probability of choosing incorrectly, and with increasing background clutter the chance of making a mistake only goes up. Mistakes can be recovered from, but a succession of wrong choices usually causes mistracking. The Condensation algorithm, on the other hand, effectively maintains multiple hypotheses over several frames, letting the best one prove itself through continued reinforcement while the probabilities of poorer hypotheses dwindle.

Because the sample set completely represents the tracker's parameters at any given time step, Condensation does not explicitly maintain the state of the tracked object as the Kalman filter does. Rather, the set of samples must be queried. The suggested method is to define the weighted mean of the samples at time t as the current object "state" in the Kalman filter sense: $\mathbf{X}_t \approx \sum_{i=1}^{N} \pi_t^{(i)} \mathbf{s}_t^{(i)}$. This works well when the posterior density is roughly unimodal, but when there are multiple strong peaks the samples can be divided fairly evenly between them, rendering this "state" a meaningless compromise. The argument that Condensation is preferable to straightforward Kalman filtering is predicated on the presence of multimodality, leaving the would-be tracker's ability to obtain a precise state estimate in doubt. What is necessary, as the authors themselves point out in [59, 60], is a "mode finder" that only averages within clusters of samples. They do not give a solution to the problem, however.

The measurement generation algorithm we presented in Chapter 4, which combines random sampling, gradient ascent, and enforcement of minimum separation, is a kind of mode finder or clusterer (this is discussed in more detail below). By solving the mode-finding problem at the front end of the filter cycle, we can use a Kalman filter to achieve a similar level of tracking robustness. That the Condensation algorithm glosses over this important step is somewhat surprising: "tracking" a distribution is not very useful by itself.

As might be expected from the Kalman filtering framework that we use, our approach to sample generation also differs from the Condensation version. We sample only in the neighborhood of the target's current predicted measurement, rather than generating samples directly from previous good samples as Condensation does in the selection step. This prevents the "cloud" of measurements associated with a target from separating into multiple groups, thus enforcing the notion that a single target is being tracked. Condensation's observation step is its mechanism for preventing runaway dispersion of samples.

In sum, Condensation is an effective approach, but it does not discredit the Kalman paradigm. The standard PDAF, by considering multiple measurements in each frame, is considerably more robust than an NN filter in many situations; this has been demonstrated in [5] and borne out in our own experiments. Furthermore, the improvements we have made to the PDAF—especially the combination of different modalities for additional distinctiveness—result in a level of tracking performance that matches that of Condensation for many difficult sequences.

7.2 Joint Tracking

The history of efforts to track multiple objects simultaneously has been relatively brief, as multiple objects require proportionally more computational effort and the difficulties of tracking single objects have been considerable enough to keep researchers well-occupied. However, as processing power has increased and problems in single-object tracking have become more thoroughly mapped, issues stemming from the interactions between multiple objects are beginning to attract attention. In this section we explore the foundations of the methods we use for JPDAF and JLF tracking as well as related efforts with similar objectives.

7.2.1 JPDAF

The Joint Probabilistic Data Association Filter (JPDAF) was originally introduced in [4] to deal with problems arising in the domain of tracking radar- and sonar-based targets. Since then, the algorithm has been corrected and extended; we use the version summarized in [5, 28] for our discussion and results. As we noted in Chapter 5, the JPDAF is typically applied to non-visual sensors with point-like returns that obviate the measurement generation problem. There has been some work on adapting the PDAF to measurements with areas, such as bright regions in low-resolution infrared sequences, by computing connected components after thresholding in [78]. An extension of the JPDAF to handle the problem of overlapping and hence merged measurements in infrared images is given in [108]. However, this work assumes that the intensity of the overlapping area is additive because the underlying targets are exhaust plumes; we cannot make this assumption because we are tracking opaque objects. Other applications of the JPDAF include tracking a fixed number of visual features for structure-from-motion [24] and autonomous navigation from a set of landmarks parametrized by range and bearing [37].

When the validation gates of a set of targets do not overlap, the JPDAF reduces to several PDAFs. Since many standard tracking methods use gradient ascent or some other iterative technique that assumes small motions, their effective validation gates are relatively small and overlaps occur infrequently. Neglecting the general superiority of PDA filters to NN filters in the presence of greater noise [5], this makes the behavior of standard NN Kalman filters (such as [117] or Kalman filtering on the output of an SSD matcher [51]) and the JPDAF similar in many visual situations. However, overlaps tend to happen more frequently when validation gates are expanded to handle higher tracking speeds, the number of objects being tracked increases, or constraints on the targets (such as with parts of nonrigid or articulated objects) encourage their proximity to one another. If these situations occur frequently enough or last long enough, the performance of the NN approximation to the JPDAF breaks down. By neglecting joint calculation of association probabilities, two targets can "claim" the same image feature and their states may converge inappropriately.

Two other data association filters that elaborate on the PDAF and JPDAF are the interacting multiple model filter (IMM) [6] and multiple hypothesis tracking

(MHT) [99], respectively. IMM maintains multiple dynamical models for a single tracked object and attempts to employ the one that best describes its behavior at all times. MHT is similar to the JPDAF but provides a rigorous approach to deciding when there is a new, trackable object and when to halt tracking of objects that have left the image or been occluded for a long time. The basic idea behind the initialization procedure is that measurements not associated with any current targets (i.e., candidates for unknown targets) are correlated with each other from frame to frame to see if any are continually found in the same vicinity. This strategy can be effective, but the computational requirements of the MHT grow exponentially over time. An efficient approximation of MHT is applied to the management of a large set of appearing and disappearing features for motion estimation in [29].

7.2.2 Joint Likelihood Filter

When targets' extents actually overlap or they are in close proximity, the JPDAF's combinatorial method of assortment becomes insufficient to avoid state convergence. We introduced the Joint Likelihood Filter, a joint tracking algorithm that addresses many of the JPDAF's shortcomings, in Chapter 5. A few other researchers have investigated similar approaches to tracking multiple interacting objects.

Rosales and Sclaroff, for example, track multiple crossing humans in [103]. They assume a fixed camera for background modeling, allowing individual foreground objects to be segmented as connected blobs. Occlusions are predicted and the system goes into a different mode during them, using pre-occlusion velocity estimates to correctly label blobs after separation. If the targets reverse direction while they are overlapping, the system can mistrack.

Stauffer and Grimson perform tracking of people, cars, and other moving objects from cameras mounted high above campus plazas and street intersections as part of a surveillance and activity categorization project [113]. They also use a per-pixel background model to identify large, connected foreground regions as putative objects. An ad-hoc, JPDAF-like method associates foreground objects with a pool of Kalman filters that are currently tracking them; further heuristics are used to decide when to spawn or kill a tracker in the manner of MHT (though neither JPDAF nor MHT is cited). According to the authors, the tracker has the most trouble in situations where objects overlap one another. One reason for this may be the low-resolution of most targets, as the camera has a fairly wide-angle lens and is mounted hundreds of feet away.

Beymer and Konolige track multiple people with stereo and SSD in [10]. They expect a person to be standing and viewed frontally or from behind, so a scalable, narrow rectangle is used to represent their size and location. To track, a foreground disparity map is first created by subtracting the background (a fixed camera is assumed). For the tracked person closest to the camera, the foreground disparity map is thresholded to isolate the layer around their predicted depth. An intensity template is correlated with the image and adjusted by matching a binary person template to the disparity map layer. A Kalman filter updates the tracker's state and the area of the foreground disparity map corresponding to the person's binary template is removed. The process is repeated for the next farthest-away person, and so on. When the number of pixels are left in the disparity map layer that overlap the binary template get too low, a person is considered completely occluded and its tracker is removed. New people are searched for by correlating scaled binary templates with the depth layers extracted from the decimated foreground disparity map. One problem with the tracking system is that it tends to delete and create person trackers instead of maintaining continuity when there are temporary full occlusions.

A somewhat more sophisticated approach is introduced in [76]. Their system tracks passing cars from a camera mounted on a highway overpass by fitting contours to contiguous regions of motion. Occasionally, one car follows another closely enough or changes lanes in such a way that a partial occlusion occurs. Without special logic to deal with this eventuality, the contour fitter becomes confused. The relative geometry of the fixed camera and the road, however, allows the system to deduce that the car whose bottom edge is lowest in the image is closer to the camera. The occluded part of the more distant car is "masked" out of the contour-fitting operation, yielding better tracking accuracy during the period of overlap.

Another system which is described in [98] tracks the fingers of a human hand with SSD templates as they bend and block one another. By matching a detailed 3-D model to the image, the depth ordering of the palm, thumb, and digits is derived and occlusions between them are predicted. Those portions of the templates predicted to be invisible are "windowed" out of the SSD calculations, a method similar to what we do in the computation of the component image likelihood for textured regions in Equation 5.5.

The essential idea of the three preceding approaches is to mask out the occluded part of an object in order to prevent the state estimator assigned to it from claiming image information generated by the occluding object. In each case, predictions are made about which targets are occluded and which should be visible. This type of information sharing about visibility between trackers is characteristic of the Joint Likelihood Filter. However, all of the other techniques that do this use a three-dimensional state configuration to inform the visibility analysis, whereas the JLF infers a depth ordering strictly from the image information. This gives the JLF tracker the flexibility to handle more visual situations and target types because camera movement is not restricted and separate objects that do not constrain one another (as opposed to connected finger joints) can be tracked.

Most similar to the Joint Likelihood Filter is the work of MacCormick and Blake in [81]. Their system tracks multiple wireframe and opaque contours using a variant of Condensation. They append the object depths to the joint state and impose transition rules that make changes in the visibility ordering improbable while the objects overlap. This is more efficient than considering all permutations of overlapping targets as we do, but it may slow the transition from erroneous initial orderings to correct ones.

Among these joint trackers, only [10] uses multiple modalities (stereo and SSD) to help discriminate between objects. We examine other research on tracking objects with more than one attribute in a later section.

7.2.3 Measurement generation

The measurement generation steps of both the single-object and joint tracking cycles, as detailed in Chapters 4 and 5, bear a resemblance to clustering methods described elsewhere. Recall that for the ith of T independently-tracked targets, the PDAF procedure is to sample N_i locations in the measurement space $\mathcal{Z}_i$ of each target, hill-climb some fraction of them with the conjugate gradient algorithm, and eliminate less-fit samples within a small neighborhood of more-fit ones. This winnowing process leaves n_i measurements; each represents one peak in the image likelihood $p_i(\mathbf{I} \mid \mathbf{X})$. The measurements can be regarded as exemplars of clusters of samples in the same basins of attraction of $p_i(\mathbf{I} \mid \mathbf{X})$ (the rest of which were removed in the minimum separation phase).

For T identical, jointly-tracked targets, the JPDAF procedure is the same except that there is only one common group of N samples narrowed to one pool of n measurements based on a single image likelihood $p(\mathbf{I} \mid \mathbf{X})$. The Joint Likelihood Filter is in some sense a higher-dimensional version of PDAF: it samples N locations in joint state space $\mathcal{X}^J$ and clusters them into n joint measurements by hill-climbing. (We treat the use of only the best joint event or joint measurement for JPDAF and JLF, respectively, as a step that comes after measurement generation rather than being integral to it).

At this point an obvious question is: Why use our method instead of running a standard grouping algorithm on the samples before doing gradient ascent? For example, one common clustering method for a known number of clusters is the K-means algorithm [53, 82]. Given n data points $\mathcal{D} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$, the algorithm is initialized by choosing k initial cluster centroids $\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$, often as random members of $\mathcal{D}$. An iteration of the algorithm consists of two parts: first, the ith data point is assigned to the nearest cluster j such that the Euclidean distance between the point and the cluster centroid $\| \mathbf{p}_i, \mathbf{c}_j \|$ is minimal. Second, the location of each cluster centroid is updated as the mean of all data points belonging to that cluster. These two steps are repeated until no data points switch memberships between clusters. The K-means algorithm is not optimal and sensitive to the initial cluster centroid locations. Therefore, a criterion for the "goodness-of-fit" of a clustering that favors low within-cluster variance and high between-cluster variance is often used to select the best classification after several repetitions of the algorithm with different random seeds. Examples of the K-means algorithm's use in vision include [30, 113].

We do not use K-means in the measurement generation process for a number of reasons. First, our gradient ascent step is still necessary as its primary purpose is to improve the samples and provide consistency in their locations to balance the randomness of their initial locations. Without it, state estimates would have an element of noise related to the sampling covariance $\mathbf{\Sigma}_{\mathcal{X}}$ and the number of samples N. Granting the utility of gradient ascent, we do not need a general clustering algorithm because of the assumption that hill-climbing brings the members of each cluster sufficiently close together. This is what makes the enforcement of minimum separation a valid grouping technique. Furthermore, K-means assumes a known number of clusters, but under our assumptions we only know that the *expected* number of clusters/measurements is T—some variation from image to image must be permitted. Finally, K-means forms clusters exclusively on the basis of proximity in the data space, which is measurement space or joint measurement space. The clusters should be based on the underlying structure of the image likelihood function, though, in the sense that samples in the same basin of attraction are grouped together. There is generally a correlation between samples that are close in measurement space and samples in the same basin of attraction, but these basins are not necessarily spherical as K-means would have them be.

Another clustering algorithm often used in vision is Expectation-Maximization (EM) [36, 84]. EM is a general algorithm for computing ML and MAP estimates with incomplete data. A special case of estimation with incomplete data is mixture models, or estimation with data that is generated by multiple processes. Assuming that there are k models and n pieces of data, EM simultaneously solves a classification problem and a parameter estimation problem. It does this by alternating between assigning each piece of data to the processes that best explain it (the expectation step) and changing the parameters of the models to reflect the data membership (the maximization step). Membership may be soft, so that a piece of data can belong to multiple models and exert weight on their parameters proportional to the degree of its membership, or it may be hard, meaning that a data point can only belong to one model at a time. K-means can be regarded as an approximation to the EM approach to mixture models in which the models are the means and covariances of spherical ellipsoids and label assignment is hard.

Bregler uses EM to label pixels as belonging to one of k coherent motion-color blobs in [20]; the output of EM becomes a measurement in a Kalman tracking filter. EM is also used to learn a dynamical model for hand tracking in [90] and for motion estimation in [2, 64, 121]. An extension is made to EM in [121] to try to estimate the number of models; the minimum description length (MDL) principle from information theory is used in [2] to select an appropriate number of models.

EM could probably be adapted for use in the measurement generation generation process, but the fit is not perfect. Typically, the data points that are clustered in other vision applications are image pixels rather than the measurements that we work with. Without completely reworking the tracking framework, a more appropriate use of EM might be to run separate instances of it to obtain a few measurements or joint measurements and use these as inputs to PDAF or the JLF. As far as we know, EM has not been applied to snake tracking; doing so would likely require a different formulation than the region-based applications described above. One advantage of the approach that we have taken to measurement generation is its generality: once a image likelihood and minimum separation distances $\mathbf{\Delta}$ for novel parameters have been defined for a new modality, our clustering method works without further modification. Lastly, as with K-means, EM cannot estimate the number of clusters by itself.

7.3 Constraints

Constraints have been used in tracking in many forms and by many researchers to improve performance. The notion of eliminating unnecessary degrees of freedom and taking advantage of relationships between nearby pixels, edges, and higher-level visual units is so basic that SSD and snake methods, for example, are often viewed as fundamental trackers rather than constrained systems of individual pixels and short edge segments, respectively. Nonetheless, constraint methods have continued to develop for targets beyond simple, rigid objects, and a major motivation for this push seems to have been the complexity of the human body. In this section we review previous work on incorporating constraints into motion-based, edge-based, and other kinds of trackers of complicated, articulated objects such as people and their faces, hands, and other body parts. Another, more recent, branch of research on constraints has focused on exploiting multiple target attributes simultaneously for additional robustness. We also survey these efforts. Finally, we briefly discuss earlier work of ours on extending the JPDAF to implement constraints probabilistically.

7.3.1 Multi-part tracking

Much of the previous work on tracking complex objects has not explicitly tackled the data association issue. One line of primarily motion-based tracking work, which is discussed in more detail below, has avoided the association or correspondence problem entirely through a differential approach. Many of these efforts have more of a flavor of pure estimation, rather than the simultaneous problem of estimation and label assignment with which this dissertation has been concerned. Here the data association problem is subsumed into the well-known correspondence problem in optical flow [54]. For example, Yamamoto and Koshikawa [124] track in-plane articulated movements of a human arm by relating arm motion to image change via the Jacobian and solving the brightness equation using least-squares. Basu et al. [7] use a similar technique to recover 3-D head motion parameters by regularizing optical flow.

Another line of tracking research has dealt with data association, but somewhat implicitly by using some form of nearest-neighbor association. Examples include Kalman snakes [117], [18], the edge-based arm tracking of Goncalves *et al.* [48], and many feature trackers used as input to motion estimation or structure from motion algorithms (examples are given in [29]). Typically, such tracking systems must manage a number of small image processing windows or validation gates (e.g., snake segments) within which multiple candidate features (e.g., edge fragments) may be detected. Correspondences are established by selecting the nearest neighbor to each predicted measurement. By stacking many measurements into a large vector and keeping the state relatively small, the measurement equation of the Kalman filter becomes overconstrained. The redundancy of multiple measurements tends to outweigh the influence on the overall object state of any individual part-measurement misassociations due to the NN method.

The many researchers who have investigated human body tracking, either in whole or in part, have encountered a few common design choices stemming from constraints. One major decision is whether to model the target *kinematically* or *dynamically* [123, 102]. A kinematic model simply specifies the connectivity between and range of motion of an object's parts, while a dynamic model also considers the influence of external forces such as gravity and the ground as well as internal forces operating at joints and points of connection. As mentioned previously, our focus on data association has led us to deemphasize dynamics, which a number of other researchers have also neglected [65, 98, 102]. Moreover, we have found a simple kinematic constraint model sufficient for a broad range of tracking tasks. Efficient mathematical representations for kinematic chain constraints, which are essentially what we use in Chapter 6, are given in [22, 65].

Regarding the actual enforcement of constraints, vision researchers have typically chosen from two methods: first, allowing separate parts to have full, possibly conflicting states and using Lagrange multipliers or a similar technique to reconcile between them [68, 123]; and second, giving the parts the minimal degrees of freedom and deriving their parameters recursively along a chain [22, 65].

Finally, articulated-object trackers can be roughly divided into those that attempt to use or recover full 3-D information, those that are only interested in 2-D information (and therefore might assume no occlusions or out-of-plane motions), and those that compute what we call "2.5-D" information. A 2.5-D representation is something like a 3-D representation, but without exact depth estimates. Instead, the scene is approximated as a set of 2-D layers for which the depth order is known. This has been our approach in Chapters 5 and 6.

One example of a 3-D tracker is the work of Bregler and Malik on tracking articulated human motions using a kinematic chain for a linear representation of joint constraints between coherent motion-color blobs [21, 22]. A constrained form of the brightness equation is solved using Newton-Raphson minimization. To aid in segmentation, a background model is used. Results show that the torso, arms, legs, feet, and head can be tracked during walking from frontal, side, and three-quarters views under normal imaging conditions. Because the method is differential, it can have difficulties with quick movements.

Pentland and Horowitz demonstrate full-body tracking in [92] using a method similar to [124]. Optical flow calculation is constrained by a 3-D model consisting of rigid cylinder-like shapes joined by springs. The Kalman-filtered state estimate is the result of cumulative optical flow and thus is subject to a build-up of error, limiting the length of accurate tracking.

Jojic *et al.* track the torso, arms, and hands of a person with a 3-D kinematic chain using a dense stereo disparity map in [65]. In addition to the body part trackers, the system maintains a background depth model. The tracker can handle prolonged self-occlusion. Wren and Pentland [123] track a person's face and hands with color blobs (similarly to [122]) using a *virtual work* formulation to enforce constraints on an articulated upper-body model. Temporary occlusions are handled, and the constraints prevent mistracking when another person's hand is interposed.

Gavrila and Davis [45] do whole-body tracking using four calibrated cameras mounted on a geodesic dome surrounding the subject. A 3-D, 22 degree of freedom (DOF) model of a person consisting of linked superquadrics corresponding to the head, torso, arms, and legs is used to predict the location of edges in the four images; pose space is searched for the best overall match. Subjects must wear skintight, contrasting clothing to assist edge detection, and edges with no history of motion are considered part of the background and removed before matching. Up to two people can move and interact fairly freely inside the dome without difficulty. Processing is done offline; no data on processing speed is given.

Rehg and Kanade present the DigitEyes system for 3-D hand and finger tracking in [98]. A 28 DOF kinematic chain model represents the palm and all five digits. A visibility ordering of the digits is generated from the state in order to predict occlusions, guiding SSD matching. Tracking is based on gradient descent on the error function. A distinction between this approach and the Joint Likelihood Filter, as we noted in the previous section, is that DigitEyes infers occlusion from the state rather than the image. This works well for predicting self-occlusions when tracking a single, articulated object, but it is not applicable to the occlusions that may occur when tracking multiple objects that are not connected to one another, such as two people. Occlusions in these kinds of situations can, however, sometimes be derived from stereo or camera calibration information as [10] and [76], respectively, demonstrate. It is likely that a combination of state-based occlusion prediction and image-based occlusion deduction would yield still better results.

Morris and Rehg [88] do whole-body tracking with a 2-D *scaled prismatic model* (SPD) that largely eliminates the singularity problems of 3-D kinematic models. The authors assert that the articulated tracking problem can be divided into two steps: a registration phase in which the model is fitted to the image, and a reconstruction phase in which 3-D parameters are recovered. They argue that the latter phase is not always necessary, such as for gesture recognition, and can always use the output of the first phase in batch form. In the SPD, joint angles are all in the image plane and links change length for out-of-plane rotations. There is no occlusion handling.

A *cardboard person* model for tracking 2-D articulated human motions with connected quadrilateral patches is presented in [66]. The brightness equation for optical flow is augmented to incorporate the articulation constraints and solved directly using a robust estimation technique. A three-level coarse-to-fine pyramid is used to cope with relatively large motions. Examples given are of tracking the thigh and calf of the foreground leg during walking motions parallel, orthogonal, and at 45 degrees to the image plane. Large motions from frame to frame can cause the gradient ascent method to mistrack, and occlusions are not handled.

Pfinder [122] maintains a background model and classifies foreground pixels as belonging to one of seven body part blobs (face, hands, feet, shirt, and pants) based on a combination of color similarity and spatial proximity to the blob center. A morphological growing operation is used to ensure connectivity within blobs. Occlusions are handled as all-or-nothing blob "disappearances," and the system can bootstrap itself onto a person entering the image. Multiple people in the scene can confuse tracking.

Kakadiaris *et al.* describe an algorithm for tracking in-plane, articulated arm and finger motions as it deduces the number of rigid parts in [68]. The edges of the arm or finger are found against a contrasting background, and initially a single deformable, physics-based model is fitted to them. When the model bends far enough, the fitting error and discontinuity in the edge curvature derivatives trigger a decision to split the model into two parts connected by a joint. Fuzzy clustering is used to share points in the border zone between parts. A Kalman filtering framework helps mitigate the effect of spurious edges introduced by partial occlusions.

Rohr models a person as a set of connected cylinders with elliptic cross sections in [102]. Medical data was used to obtain a one-variable phase angle parametrization of the internal configuration (joint angles and limb positions) of the model for a walking motion parallel to the image; the horizontal image position of the body centroid is estimated separately. Assuming a fixed, calibrated camera, a Kalman filter predicts the location of straight edges in the image and computes the degree of match. Analogous to our argument against a too-specific dynamical model, the kinematic model here is overspecialized. This tracker cannot handle anything but one person walking in profile at a constant velocity.

Reynard *et al.* investigate coupling frontal mouth and head snake trackers in [100] as a method for preventing mistracking of the mouth during lateral head movements. Without the coupling, the mouth tracker depends primarily on vertical edges at the top and bottom of the lips and therefore suffers from the aperture problem during horizontal motions. When linked with the mouth in a combined Kalman tracker, however, the strong horizontal edges on the sides of the head silhouette compensate for this vulnerability and allow a full range of motion.

7.3.2 Multi-attribute tracking

Investigations into exploiting multiple different modalities for tracking have become more common in the past few years. This can be ascribed in part to the additional computational overhead entailed by combining methods, and partially to the tendency of some researchers to continue refining the single-modality techniques with which they are most familiar. Even those systems that do rely on different modalities rarely use them at the same time. Most often, a set of heuristics arbitrates between which modality to employ or favor at any given time.

Darrell *et al.* combine stereo, color, and a face detection module to track frontal views of multiple people's faces and upper torsos in [31]. No Kalman filter is used; the state of each person is updated using a nearest neighbor approach after identifying candidate person locations. Candidate locations are found by running each of the three modules separately to find good range profiles, skin color, and face matches, respectively. These separate candidates are ranked, with preference given in decreasing order to face matches, overlapping range and color candidates, range candidates, and color candidates. This is in contrast to the Constrained Joint Likelihood Filter outlined in the previous chapter, which always looks for conjunctions of the attributes it is using and does not favor one over another. The system does not recognize the occurrence of occlusions *per se*, but rather waits for people to separate before using the face detector to reclassify them.

Birchfield uses color and intensity gradients for head tracking [11]. The color and edge match criteria are described above; to compare these two sources of information at a common scale, the two independent match formulae are converted into percentages by dividing by their range of possible values. The effect of this step is similar to the function of $\sigma_{tregion}, \sigma_{hregion}$, and σ_{snake} in the component image likelihoods given by Equations 5.4, 5.6, and 5.8, respectively.

Other examples of multi-attribute tracking include Bregler's inclusion of both color and motion parameters in a multivariable Gaussian representing a blob [20]. This implicitly handles the differences in scale between the two quantities. Mae *et al.* use optical flow plus contours found by edge detection to improve tracking in ambiguous visual situations [83], and Pfinder [122] combines the output of the color blob segmenter with a contour analysis step in a heuristic way.

Perseus, a vision system mounted on a mobile robot, computes multiple feature maps (color, motion, and disparity) to figure out where a person is pointing [67]. When the robot is not moving, background subtraction is used to finding the person; when it is moving, stereo disparity is used (after eliminating pixels with of known floor color or bright enough to be ceiling lights). A geometric analysis is then used to determine where the arm is; arm-finding fails if the hand is not away from the body. The different cues are not used simultaneously, but rather in sequence depending on certain conditions.

The Incremental Focus of Attention (IFA) algorithm [118, 119] is a system of decision criteria for switching between coarse-to-fine tracking methods (motion, color, SSD) in order to maximize accuracy and recover quickly and robustly from mistracking. Again, the different methods are used sequentially rather than simultaneously.

7.3.3 Constrained Joint Probabilistic Data Association Filter

In previous research [96, 97], we augmented the JPDAF to include a method of enforcing constraints probabilistically, which we called the Constrained Joint Probabilistic Data Association Filter (CJPDAF). The CJPDAF works by quantifying how well the relationships between measurements fit the desired constraints between their associated targets. Introducing constraints into the state update process at this point affects the computation of the association probabilities, which are determined in Equation 5.1. By encoding a probabilistic preference for certain part arrangements, the effect is to favor those interpretations of the data that best fit the model rather than force the target state into fitting it (which is what the Constrained Joint Likelihood Filter (CJLF) does). The only examples of constraints used in [96, 97] are rigid links with no rotation or scaling, but a functional definition of the inter-part constraints in [97] allows for more complicated relationships. The CJPDAF is often a useful, flexible way to introduce constraints into a multi-part tracking system, but by not imposing them until after the measurement generation step, there is a possibility that the independently-generated measurements cannot be arranged in a joint event that closely satisfies the constraints. By jointly generating measurements that meet the constraints from the very beginning of the process, however, the CJLF proves to be much more robust.

Chapter 8

Conclusion

This dissertation's primary contribution is its demonstration of the importance of reasoning about correspondences between trackers and image data in order to achieve robust vision-based tracking. In arguing that standard estimation techniques are often inadequate for real-world tracking tasks, we have catalogued a number of disruptive visual phenomena such as agile motions, occlusions, and distractions that make tracking difficult and presented a series of new methods to counteract them.

One innovation of this work is the analogy we have drawn between visual occlusions and distractions and the problem of no or multiple measurements that algorithms such as the Probabilistic Data Association Filter (PDAF) and Joint Probabilistic Data Association Filter (JPDAF) [5] were intended to solve. Though these filters were originally developed for discrete radar and sonar tracking applications, we were able to successfully adapt them to visual tasks by defining measurements suitably and devising a preprocessing step to extract them. Run head-to-head on the same image sequences, the vision-based tracking algorithms thus created exhibited markedly better performance in the presence of clutter and when tracking multiple identical objects than many current commonly-used methods.

Our technique for generating the measurements used as input to these data association filters is also notable. We have found that a combination of random sampling and gradient ascent for extracting a discrete set of high-likelihood hypotheses about characteristics of the target's image projection consistently yielded accurate results. The identification and consideration of a group of alternative hypotheses in the neighborhood of the tracking filter's prediction allowed ambiguities to be resolved over multiple frames and helped to quickly relocate the target when it moved with agility. This procedure is also general enough that it can be applied to any new tracking modality defined in the manner of the examples in Chapter 3.

We have also explicated shortcomings in the JPDAF and remedied them with a more efficient and sophisticated method, the Joint Likelihood Filter (JLF). By relating the exclusion principle at the heart of the JPDAF to the method of masking out image data [76, 98], the JLF handles occlusions between tracked objects. Our extension of this method to collections of objects of different modalities such as color, shape, and appearance is original. The approach we take to color representation and region geometry for homogeneous regions is our own. Moreover, though others have used three-dimensional state parameters to assist with occlusion reasoning, the JLF's inference of the depth ordering of tracked objects during overlaps from image data alone is novel.

Finally, we augmented the JLF method to allow low-level trackers to be composed via part and attribute constraints in order to specify more complex targets. This algorithm, the Constrained Joint Likelihood Filter (CJLF), reduces the vulnerability of a vision-based tracker to unmodeled distractions and occlusions by effectively defining its target more distinctively. Although geometric constraints are a well-established method for increasing robustness, exploiting multiple modalities simultaneously to track a single object—especially three, as we do—is fairly new, and the union of these two approaches is clearly an advance. The way that the CJLF framework does so is made more useful by its flexibility and extensibility: target models can be easily specified and new modalities can be added straightforwardly.

The main claim that we make about these new algorithms is that they offer qualitative improvements in tracking performance over existing methods when the visual disruptions we enumerated are present in an image sequence. That is, by modeling distractions and occlusions as we do, we add a novel capability that standard techniques lack and allow tracking to proceed successfully where it otherwise it is prone to fail. Such a claim is distinct from one of simple quantitative superiority, which, for example, might assert that the expected error between state estimates and the ground truth is smaller for one algorithm than another. The latter claim follows from the first, in a sense, but here we have focused on the inability of standard algorithms to cope with certain phenomena as an obstacle to their more widespread applicability.

We took several approaches in this dissertation to demonstrating the efficacy of our tracking methods. First and foremost, we have argued through hypothetical examples and derivations that, for example, not carrying out the measurement process jointly and/or exploiting constraints between trackers when they apply leads to an incorrect formulation of the posterior probability on the state given the images observed. Correspondingly, we have sought to show how the JPDAF and JLF address the first problem through the image likelihood and how the CJLF addresses the second through the prior on the state. These probabilistic arguments demonstrate why the visual disruptions classified constitute significant violations of the assumptions of standard tracking techniques.

We also buttressed this theoretical approach with empirical evidence obtained from tracking experiments on real image sequences. By attempting to track in various situations with standard techniques, we confirmed the detrimental consequences of visual disruptions. Noise, interactions between multiple tracked objects, and insufficient constraints between linked parts frequently led to mistracking as predicted. Conversely, when an appropriate new algorithm was employed, the incidence of mistracking was greatly reduced. The figures that make up the bulk of the data discussed in each chapter's results section were selected as representative of many head-to-head runs. Repetitions were performed both to gauge the consistency of the results for minor parameter variations and to give a better statistical picture when the algorithms had a stochastic component. In several cases, we have summarized the results of a batch of runs by the fraction in which the object was successfully tracked. The wide disparities in performance quantified in those examples are characteristic of what we observed over the entire corpus of experiments, but regrettably we did not do the same precise tabulation for all of them.

The way that we chose the image sequences in this dissertation points up a general difficulty with comparing tracking methods. While we strived to include sequences that had different kinds of objects, motions (both of the object and the camera), and backgrounds in order to put a range of stresses on the various algorithms and modalities, there is still a question of how representative the sequences are. Addressing this issue is more straightforward when the problem domain of the tracker is known and limited, but we have tried to keep our framework general. What is really called for is in some sense a basis set of sequences. The theory behind what would go into such a set is still embryonic, however, and the tracking community has yet to even achieve consensus on any common set of sequences suitable for comparison. Moreover, when using real-world video clips it is hard to control enough of the visual variables to support a claim that the success or failure of tracking is due to one particular factor. This difficulty is compounded by the fact that where good sequences are available, it is often laborious if not impossible to extract ground truth from them. Synthetic sequences along the lines of what we used in Figure 4.8 would seem to remedy this problem, but whether photorealism is necessary and how applicable results obtained on synthetic sequences are to real-world situations are issues that demand more study.

In the following section, we will discuss some other questions that have arisen in the course of this research which we plan to investigate further.

8.1 Future work

There are a few major directions which we see as promising for improving and extending the work presented here.

Additional modalities An obvious next step in our research is to implement more tracking modalities within the framework described for single objects in Chapter 3 and joint objects in Chapter 5. Additional descriptive attributes would further increase the distinctiveness of any tracked object, boosting the reliability of tracking in more difficult visual situations. Moreover, when limited processing power does not allow the use of all available modalities, careful selection of the most useful ones at hand is necessary. A larger set of methods with complementary strengths and weaknesses would afford better coverage in situations where the object's color, appearance, or shape alone are difficult to discern due to lighting, resolution, or other limitations.

A number of well-studied candidate modalities such as motion and stereo are reviewed in Section 7.1.1. Some other novel tracking cues also seem worthy of investigation. For example, one interesting cue is texture in a statistical sense. Suppose that we would like to recognize and track an object covered with vertical stripes, wavy lines, spots, or some other repetitive pattern. Defining a texture procedurally or parametrically as opposed to the static reference image of a textured region would make for a more concise, view-insensitive object description. Moreover, classes of objects (e.g, zebras) instead of only single instances (e.g., *this* zebra) could be characterized, and single-instance tracking would likely be more robust. Zhu *et al.* do texture modeling by picking a basis set of filters using information theory [126] and Efros and Leung [39] synthesize textures from a sample using a non-parametric technique. A corollary of such results which might lead to a quite versatile and powerful tracker is a more complex texture comparison function than simply subtracting the intensities of corresponding pixels. However, these texture modeling methods are currently too computationally intensive to be feasible for tracking.

Modality comparison A natural question to ask about cues concerns their efficacy. Thus far we (and many other researchers, it seems) have relied solely on intuition and empirical observation to guide the choice of which tracking modality works best for a given image feature. Furthermore, our use of MPEGs for input has given us the luxury of concluding (as have others—e.g., [31]) that more modalities are always more helpful. With a large group of cues available and a real-time task that places hard limits on computational resources, however, it becomes paramount to have a rigorous technique for selecting one best cue or a minimal subset of cues that satisfy the task's requirements efficiently. What is necessary is a decision criterion that accounts for the characteristics of the target being tracked and the image conditions at any given time and makes a reasoned choice about the superiority of a textured region vs. a homogeneous region vs. a snake vs. any other modality. As a corollary, we would like to be able to predict when tracking human faces, for example, how much of a performance improvement would be gained by augmenting a homogeneous region with a textured region instead of, say, adding a snake. Such inquiries are essentially concerned with quantifying distinctiveness.

One possible basis for a theory of comparing tracking modalities is some sort of measure for our confidence that the estimate returned by the tracker is a good one. Confidence in an estimate may stem from a number of factors, but a very important one is how much better (in a probabilistic sense) it is than the alternatives. How much does the estimate stand out as *the* solution rather than *a* solution? If we view the visual cues as akin to channels carrying messages of varying helpfulness about the value of the state $\mathbf{X}$, information theory provides a principled approach to quantifying these intuitive qualities. In particular, Fisher information [27] is promising as a bridge between information theory and estimation. The Fisher information $\mathbf{J}$ is a lower bound on the variance of an unbiased estimator of a set of parameters of a probability density. Here the parameters are contained in $\mathbf{X}$, the probability density $p(\mathbf{X} \mid \mathbf{I})$ is conditioned on the current image $\mathbf{I}$, and the estimator is one of the candidate tracking methods. As a yardstick for an estimate's repeatability, $\mathbf{J}_A$ for a particular modality A can be regarded as a maximum confidence measure in that modality's tracker for a given set of state parameters and image conditions. This makes it relevant for comparing the relative effectiveness of various tracking methods.

There has been some previous work on the theory of visual cue integration using Fisher information and other information theoretic approaches. Yuille and Bülthoff [125] develop a theory about fusing binocular stereo and monocular depth cues for motion parallax, as well as shape from shading and texture. Blake *et al.* [15] use the compression and density of texture elements on a surface as cues for estimating its tilt and slant in computer-generated images, analyzing the usefulness of each according to its Fisher information. However, there has been no research to date on employing

Fisher information to select and integrate cues specifically for tracking, so this might be a fruitful avenue.

Tracker initialization and termination Another area which deserves more attention is the initiation and termination of tracking. Currently, a user points and clicks with a mouse to set a tracker's initial state parameters—e.g., by choosing locations along a snake's contour or indicating a region's image position, angle, width, and height. The user then presses a button to start tracking, which continues until the end of the MPEG is reached or the program is killed. There are several reasons why it would be beneficial to have tracking begin and end without user intervention. The first is convenience. The commencement of tracking can be easily automated with script files for MPEGs, but repeatedly picking the face to be tracked in a live image sequence, for example, is tiresome. A higher-level process that looks everywhere for faces and starts tracking in locations with good matches would obviate this chore. A second reason is that tracked objects sometimes leave the frame or are completely occluded for long stretches of time. Recognizing such absences would prevent erroneous state estimates from being promulgated, and refinding lost objects to resume tracking would allow a tracker to run for longer periods and recover from clutter for which this dissertation's methods are insufficient. Such skills have been termed *post-failure robustness* [118]. Lastly, an ability to discover image features that match the target model but have not been pointed out by the user could also identify persistent distractors. As a result, distractors could be treated as trackable objects in their own right and dealt with using the JPDAF or Joint Likelihood instead of approximating them as noise or minimizing their occurrence with additional modalities.

The high-level search process posited for initiating a tracker assumes the existence of a detailed model of the sought object's color, appearance, or shape in order to test hypotheses. A major difference between such a search and the tracking measurement process detailed in Chapter 4 is the markedly greater breadth and hence computational load required for initialization. The prior on the state $p(\mathbf{X})$ is considerably more diffuse because there is no prediction from the previous frame to narrow the focus. Close fits to the object model must be subjected to a thresholding step to decide whether to actually start a new tracker. As for termination, a separate module would monitor the health of existing trackers in order to determine when they no longer have sufficient image corroboration to continue.

The Condensation algorithm [59] starts tracking with a spread-out prior and continually shifts resources away from less likely areas of state space to more promising ones, so in a sense automatic initialization and termination are built in. Nevertheless, the same failure to make a discrete decision about how many targets are being tracked which we referred to in Section 7.1.2 also means that there is no definite decision as to whether a new target has appeared or an old one is gone. In contrast, multiple hypothesis filtering [5, 28], which we covered in Section 7.2.1, is a data association technique similar to the JPDAF that is very explicit about these events. It automatically detects new targets and eliminates obsolete targets with a sophisticated methodology that takes several frames to make a decision. However, it is quite expensive computationally and seems to have only been applied in vision to fixed-size, SSD-type features [29].

The expense of the broad search necessary for automatic initialization is only compounded by the high-dimensional states resulting from the kinematic constraints introduced in Chapter 6. The strategy of Gavrila and Davis in [45] is one interesting approach to improving efficiency. They search pose space hierarchically for body parts: first the torso, then the arms, then the legs. Their intuition is that the torso is easier to locate because it varies less in position and orientation, and can then "anchor" the search for attached limbs. This is somewhat similar to the *partitioned sampling* technique used by MacCormick and Blake in [81] to reduce the number of samples needed for Condensation tracking of objects or groups of objects with high-dimensional states. The basic idea of searching for object parts in descending order of distinctiveness would likely boost the performance of ordinary tracking as well.

A problem closely related to automatic initialization is how to find features worth modeling and tracking in the first place. This version has arisen in structure-from-motion problems [29] and surveillance tasks [103, 113]. In this case there is no specific representation of the target *per se* (like a reference image of a face or a skin color model), but rather only an idea of an ideal target's properties. The chief such property used by researchers is sometimes called "trackability," a virtual synonym for distinctiveness. Shi and Tomasi establish strong vertical and horizontal gradients as a trackability criterion for SSD features in [109]; motion-sensitive trackers usually look for a compact, connected group of strongly-changing pixels [103, 113]. Fisher information is used in [115] to derive a goodness measure similar to that of [109] by searching for fixed-size regions that maximize $\mathbf{J}$. A procedure such as this might be generalizable to other modalities.

Model learning It would also be advantageous to ultimately incorporate learning into our tracking framework. Learning affords an opportunity to accurately tune object dynamics and build more refined models of appearance, color, kinematics, and so on. The trained snake trackers in [18, 100], for example, obviously react better than their untrained counterparts to agile motions. Improving a tracker's skills continuously through an online version of their learning algorithm would be ideal.

Another benefit of learning is that rough models entered by hand can be augmented or corrected over time, and long-term modifications to the tracked object such as beard growth or changes in clothing can be accommodated adaptively. The system in [68] initially models a human arm as a single, flexible object until it observes a large enough bend to deduce that there are actually two parts (the forearm and upper arm) joined at the elbow. An elaboration of this kind of approach might be able to first approximate a whole human body as a single rectangle and gradually parse out the structure of the torso, limbs, and head, avoiding the explicit spelling-out of the kinematic chain that must currently be done.

Code optimization A final consideration for future work is a technological one: the implementation of the tracking algorithms described in this dissertation could likely be sped up greatly with a concerted effort to optimize code. We have not been concerned as much with maximizing speed as with improving general tracking performance, and this is reflected in the running times of our algorithms, which range from near real-time (defined as tracking calculations keeping pace with a 30 frames-per-second stream of 640×480 images) to seconds per frame on a 650 MHz Pentium III. Nonetheless, besides facilitating more efficient testing and comparison of methods, a reliably real-time implementation is a prerequisite to any real-world application of the full spectrum of our methods.

Speed depends on a number of factors, most importantly the number and type of atomic trackers, iterations of gradient ascent allowed, and number of state samples explored. Profiling the code shows, unsurprisingly, that a major portion of its time is spent doing image processing in repeated evaluations of $p(\mathbf{I} \mid \mathbf{X})$ for different state samples. Using the SIMD capabilities offered by the MMX component of the Pentium III chip, which we do not, would cut down per-frame computation time a great deal.

For example, benchmarks for the Intel Image Processing Library (IPL) [57] indicate as much as a 500% improvement going from non-SIMD to SIMD versions of common image operations such as convolution, zoom, and addition.

Another intriguing possibility is suggested by the work on active blob tracking in [106]. They use the specialized 3-D hardware of an OpenGL graphics card to quickly synthesize expected images of a deformable tracked object from its state. This ability would be a great help to us in performing, for example, the affine warp with bilinear interpolation on the reference image $\mathbf{I}_R$ that is part of the image likelihood for textured regions. The recent OpenGL 1.2 specification [110] includes some image processing functions akin to those in the Intel IPL, making it likely that 3-D graphics cards will soon provide even more hardware that can be exploited for tracking. Moreover, our use of MPEGs instead of live input in order to facilitate exact comparisons causes some performance degradation due to the extra load on the CPU of software MPEG decoding. Most current graphics cards can do this decoding in hardware.

8.2 Coda

A guiding motivation throughout this thesis has been that distractions, occlusions, and sudden movements pose a major challenge to the reliability of vision-based tracking for both people and machines.

How do people cope? The basic neurobiology of tracking consists of eye (and ultimately head) motions aimed at keeping the image projection of the object of interest on the fovea, the high-acuity center of the retina that subtends about $1°$ of the visual field [71]. Smooth pursuit movements of both eyes, which follow targets at moderate speeds, are punctuated by quick saccadic movements. Saccades "catch up" when target motion is too fast or discontinuous and also serve to shift attention to interesting stimuli in the periphery of the visual field. Perceptions of large objects such as human faces that do not fit within the fovea seem to be constructed from repeated saccades between smaller areas of interest [71].

A comprehensive theory of how the visual system makes attentional choices and acts upon them is, of course, still nascent. Nonetheless, these findings agree with the intuition that as we track a complex object such as another human being, a constant series of adjustments must be made to where we are looking and what we are looking for in order to compensate for unpredictable motions, disappearances, and ambiguities. Consider the friend-in-a-crowd scenario alluded to in the first chapter. To follow our friend we can try to fixate on their face, but the similarity of the surrounding faces may prompt quick, continual scans of the vicinity to insure that there have been no mix-ups. If this is too confusing, we might attend to our friend's hair color as long as it is distinctive enough. When the friend's head is blocked, our attention shifts to their still-visible shirt. If they are entirely obscured, we anticipate their reemergence at some predicted spot or simply search that area of the image until they are refound. The sum total of this patchwork of strategies is the human ability to maintain visual contact with a target despite many severe disruptions.

Although we make no claims that our methods are in any way consonant with the mechanisms of the human visual system, we have drawn inspiration from them. Responding to the problems detailed above, this dissertation has taken the view that unless traditional estimation techniques are bolstered with explicit reasoning about these phenomena, tracking performance in many real-world situations inevitably suffers. The philosophical touchstone of our work has been that robustness can be increased by exploiting multiple sources of information simultaneously. This approach is manifested in many innovative aspects of our framework: e.g., individual trackers avoid commitment to a single attractive feature to defend against noise, groups of trackers share information about correspondence choices to avoid interfering with one another, and diverse geometric and qualitative cues are integrated to increase distinctiveness. The value of these methods is demonstrated by their superior tracking performance on many objects in the presence of difficult clutter and partial occlusions.

Bibliography

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *Int. J. Computer Vision*, 2(3):283–310, 1989.

[2] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum likelihood estimation of mixture models and MDL encoding. In *Proc. Int. Conf. Computer Vision*, pages 777–784, 1995.

[3] A. Azarbayejani and A. Pentland. Real-time self-calibrating stereo person tracking using 3-d shape estimation from blob features. In *Proc. Int. Conf. Pattern Recognition*, 1996.

[4] Y. Bar-Shalom. Extension of the probabilistic data association filter to multi-target environments. In *Proc. 5th Symposium on Nonlinear Estimation*, 1974.

[5] Y. Bar-Shalom and T. Fortmann. *Tracking and Data Association.* Academic Press, 1988.

[6] Y. Bar-Shalom and X. Li. *Multitarget-Multisensor Tracking: Principles and Techniques.* YBS Publishing, 1995.

[7] S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. In *Proc. Int. Conf. Pattern Recognition*, 1996.

[8] A. Baumberg and D. Hogg. An efficient method for contour tracking using active shape models. In *Proc. IEEE Workshop on Motion of Nonrigid and Articulated Objects*, 1994.

[9] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proc. European Conf. Computer Vision*, pages 237–252, 1992.

[10] D. Beymer and K. Konolige. Real-time tracking of multiple people using stereo. In *IEEE Workshop on Frame Rate Applications, Methods and Experiences with Regularly Available Technology and Equipment*, 1999.

[11] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proc. Computer Vision and Pattern Recognition*, pages 232–237, 1998.

[12] M. Black, D. Fleet, and Y. Yacoob. A framework for modeling appearance change in image sequences. In *Proc. Int. Conf. Computer Vision*, pages 660–667, 1998.

[13] M. Black and A. Jepson. Recognizing temporal trajectories using the Condensation algorithm. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1998.

[14] M. Black and A. Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *Int. J. Computer Vision*, 19(1):57–91, 1996.

[15] A. Blake, H. Bülthoff, and D. Sheinberg. Shape from texture: Ideal observers and human psychophysics. In D. Knill and W. Richards, editors, *Perception as Bayesian Inference*, pages 287–321. Cambridge University Press, 1996.

[16] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *Int. J. Computer Vision*, 11(2):127–145, 1993.

[17] A. Blake and M. Isard. 3-d position, attitude and shape input using video tracking of hands and lips. In *SIGGRAPH*, pages 185–192, 1994.

[18] A. Blake, M. Isard, and D. Reynard. Learning to track the visual motion of contours. *Artificial Intelligence*, (78):101–133, 1995.

[19] G. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, April 1998.

[20] C. Bregler. Learning and recognizing human dynamics in video sequences. In *Proc. Computer Vision and Pattern Recognition*, pages 568–574, 1997.

[21] C. Bregler and J. Malik. Video motion capture. Technical Report UCB-CSD-97-973, University of California, Berkeley, 1997.

[22] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. Computer Vision and Pattern Recognition*, pages 8–15, 1998.

[23] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[24] Y. Chang and J. Aggarwal. 3-d structure reconstruction from an egomotion sequence using statistical estimation and detection theory. In *Proc. IEEE Workshop on Visual Motion*, pages 268–273, 1991.

[25] D. Coombs, M. Herman, T. Hong, and M. Nashman. Real-time obstacle avoidance using central flow divergence and peripheral flow. In *Proc. Int. Conf. Computer Vision*, pages 276–283, 1995.

[26] T. Cootes, C. Taylor, A. Lanitis, and D. Cooper. Building and using flexible models incorporating grey-level information. In *Proc. Int. Conf. Computer Vision*, pages 242–246, 1993.

[27] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley and Sons, 1991.

[28] I. Cox. A review of statistical data association techniques for motion correspondence. *Int. J. Computer Vision*, 10(1):53–65, 1993.

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

[30] R. Cutler and M. Turk. View-based interpretation of real-time optical flow for gesture recognition. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1998.

[31] T. Darrell, G. Gordon, M. Harville, and J. Woodfill. Integrated person tracking using stereo, color, and pattern detection. In *Proc. Computer Vision and Pattern Recognition*, pages 601–608, 1998.

[32] J. Davis and A. Bobick. The representation and recognition of action using temporal templates. In *Proc. Computer Vision and Pattern Recognition*, 1997.

[33] A. Davison and D. Murray. Mobile robot localisation using active vision. In *Proc. European Conf. Computer Vision*, 1998.

[34] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the Condensation algorithm for robust, vision-based mobile robot localization. In *Proc. Computer Vision and Pattern Recognition*, pages 588–594, 1999.

[35] F. Dellaert, D. Pomerleau, and C. Thorpe. Model-based car tracking integrated with a road-follower. In *Proc. Int. Conf. Robotics and Automation*, 1998.

[36] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society B*, 39:1–38, 1977.

[37] J. Dezert and Y. Bar-Shalom. Joint probabilistic data association for autonomous navigation. *IEEE Trans. Aerospace and Electronic Systems*, 29(4):1275–1285, 1993.

[38] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[39] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proc. Int. Conf. Computer Vision*, 1999.

[40] I. Essa, T. Darrell, and A. Pentland. Tracking facial motion. In *Proc. IEEE Workshop on Motion of Nonrigid and Articulated Objects*, 1994.

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

[42] M. Fleck, D. Forsyth, and C. Bregler. Finding naked people. In *Proc. European Conf. Computer Vision*, pages 592–602, 1996.

[43] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics - Principles and Practice*. Addison-Wesley, 1989.

[44] W. Freeman, K. Tanaka, J. Ohta, and K. Kyuma. Computer vision for computer games. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1996.

[45] D. Gavrila and L. Davis. 3-d model-based tracking of humans in action: A multi-view approach. In *Proc. Computer Vision and Pattern Recognition*, 1996.

[46] R. Gershon, A. Jepson, and J. Tsotsos. Ambient illumination and the determination of material changes. *J. Opt. Soc. America A*, 3(10):1700–1707, 1986.

[47] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, 1989.

[48] L. Goncalves, E. Di Bernardo, E. Ursella, and P. Perona. Monocular tracking of the human arm in 3-d. In *Proc. Int. Conf. Computer Vision*, pages 764–770, 1995.

[49] U. Grenander, Y. Chow, and D. Keenan. *HANDS: A Pattern Theoretical Study of Biological Shapes.* Springer-Verlag, 1991.

[50] S. Grossman. *Multivariable Calculus, Linear Algebra, and Differential Equations: Second Edition.* Academic Press, 1986.

[51] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[52] G. Hager and K. Toyama. The "XVision" system: A general purpose substrate for real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–27, January 1998.

[53] J. Hartigan. *Clustering Algorithms.* John Wiley and Sons, 1975.

[54] B. Horn. *Robot Vision.* MIT Press, 1986.

[55] I. Horswill. *Specialization of Perceptual Processes.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1993.

[56] J. Hoschek and D. Lasser. *Fundamentals of Computer-Aided Geometric Design.* A.K. Peters, 1993.

[57] Intel Corporation. Intel(R) Image Processing Library. Available at `http://developer.intel.com/vtune/perflibst/ipl/index.htm`. Accessed April 9, 2000.

[58] M. Irani and P. Anandan. A unified approach to moving object detection in 2-d and 3-d scenes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(6):577–589, 1998.

[59] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. European Conf. Computer Vision*, pages 343–356, 1996.

[60] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29:5–28, 1998.

[61] M. Isard and A. Blake. ICondensation: Unifying low-level and high-level tracking in a stochastic framework. In *Proc. European Conf. Computer Vision*, pages 893–908, 1998.

[62] M. Isard and A. Blake. A mixed-state Condensation tracker with automatic model-switching. In *Proc. Int. Conf. Computer Vision*, pages 107–112, 1998.

[63] M. Isard and A. Blake. A smoothing filter for Condensation. In *Proc. European Conf. Computer Vision*, pages 767–781, 1998.

[64] A. Jepson and M. Black. Mixture models for optical flow computation. In *Proc. Computer Vision and Pattern Recognition*, pages 760–761, 1993.

[65] N. Jojic, M. Turk, and T. Huang. Tracking self-occluding articulated objects in dense disparity maps. In *Proc. Int. Conf. Computer Vision*, pages 123–130, 1999.

[66] S. Ju, M. Black, and Y. Yacoob. Cardboard people: A parametrized model of articulated image motion. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, pages 38–44, 1996.

[67] R. Kahn, M. Swain, P. Prokopowicz, and R. Firby. Gesture recognition using the Perseus architecture. In *Proc. Computer Vision and Pattern Recognition*, 1996.

[68] I. Kakadiaris, D. Metaxas, and R. Bajcsy. Active part-decomposition, shape, and motion estimation of articulated objects: A physics-based approach. In *Proc. Computer Vision and Pattern Recognition*, pages 980–984, 1994.

[69] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[70] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *Proc. Computer Vision and Pattern Recognition*, pages 196–202, 1996.

[71] E. Kandel, J. Schwartz, and T. Jessell. *Principles of Neural Science*. Elsevier, 1991.

[72] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Computer Vision*, pages 321–331, 1988.

[73] C. Kidd, G. Abowd, C. Atkeson, I. Essa, B. MacIntyre, E. Mynatt, and T. Starner. The aware home: A living laboratory for ubiquitous computing research. In *Proc. of the Second Int. Workshop on Cooperative Buildings*, 1999.

[74] G. Klinker, S. Shafer, and T. Kanade. A physical approach to color image understanding. *Int. J. Computer Vision*, 4:7–38, 1990.

[75] D. Knill, D. Kersten, and A. Yuille. Introduction: A Bayesian formulation of visual perception. In D. Knill and W. Richards, editors, *Perception as Bayesian Inference*, pages 1–21. Cambridge University Press, 1996.

[76] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *Proc. Computer Vision and Pattern Recognition*, pages 189–196, 1994.

[77] K. Konolige. Small vision systems: Hardware and implementation. In *Eighth Int. Symp. on Robotics Research*, 1997.

[78] A. Kumar, Y. Bar-Shalom, and E. Oron. Precision tracking based on segmentation with optimal layering for imaging sensors. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(2):182–188, 1995.

[79] A. Lipton, H. Fujiyoshi, and R. Patil. Moving target classification and tracking from real-time video. In *Workshop on Applications of Computer Vision*, 1998.

[80] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[81] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proc. Int. Conf. Computer Vision*, 1999.

[82] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability: Vol. 1*, pages 281–297, 1967.

[83] Y. Mae, Y. Shirai, J. Miura, and Y. Kuno. Object tracking in cluttered background based on optical flow and edges. In *Proc. Int. Conf. Pattern Recognition*, 1996.

[84] G. McLachlan and K. Basford. *Mixture Models: Inference and Applications to Clustering.* Dekker, 1988.

[85] P. Meer, D. Mintz, D. Kim, and A. Rosenfeld. Robust regression methods for computer vision: A review. *Int. J. Computer Vision*, 6(1):59–70, 1991.

[86] J. Mendel. *Lessons in Digital Estimation Theory.* Prentice-Hall, 1987.

[87] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[88] D. Morris and J. Rehg. Singularity analysis for articulated object tracking. In *Proc. Computer Vision and Pattern Recognition*, pages 289–296, 1998.

[89] D. Mumford. Pattern theory: A unifying perspective. In D. Knill and W. Richards, editors, *Perception as Bayesian Inference*, pages 25–62. Cambridge University Press, 1996.

[90] B. North and A. Blake. Learning dynamical models using Expectation-Maximization. In *Proc. Int. Conf. Computer Vision*, pages 384–389, 1998.

[91] E. Oja. *Subspace Methods of Pattern Recognition.* Research Studies Press, 1983.

[92] A. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):730–742, July 1991.

[93] C. Poynton. ColorFAQ: Frequently Asked Questions about Color. Available at `http://www.inforamp.net/ poynton/ColorFAQ.html`. Accessed Sept. 30, 1999.

[94] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1993.

[95] C. Rasmussen and G. Hager. Tracking objects by color alone. Technical Report DCS-RR-1114, Yale University, 1996.

[96] C. Rasmussen and G. Hager. Joint probabilistic techniques for tracking multi-part objects. In *Proc. Computer Vision and Pattern Recognition*, pages 16–21, 1998.

[97] C. Rasmussen and G. Hager. Joint probabilistic techniques for tracking objects using multiple visual cues. In *Proc. Int. Conf. Intelligent Robots and Systems*, pages 191–196, 1998.

[98] J. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc. Int. Conf. Computer Vision*, pages 612–617, 1995.

[99] D. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automatic Control*, 24(6):843–854, 1979.

[100] D. Reynard, A. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *Proc. European Conf. Computer Vision*, pages 357–368, 1996.

[101] B. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, 1996.

[102] K. Rohr. Incremental recognition of pedestrians from image sequences. In *Proc. Computer Vision and Pattern Recognition*, pages 8–13, 1993.

[103] R. Rosales and S. Sclaroff. 3-d trajectory recovery for tracking multiple objects and trajectory guided recognition of actions. In *Proc. Computer Vision and Pattern Recognition*, 1999.

[104] K. Russell, T. Starner, and A. Pentland. Unencumbered virtual environments. In *IJCAI-95 Workshop on Entertainment and AI/Alife*, 1995.

[105] H. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(8):814–830, 1996.

[106] S. Sclaroff and J. Isidoro. Active blobs. In *Proc. Int. Conf. Computer Vision*, 1998.

[107] S. Shafer, B. Brumitt, B. Meyers, M. Czerwinski, and D. Robbins. The new EasyLiving project at Microsoft Research. In *Joint DARPA/NIST Smart Spaces Workshop*, 1998.

[108] H. Shertukde and Y. Bar-Shalom. Tracking of crossing targets with imaging sensors. *IEEE Trans. Aerospace and Electronic Systems*, 27(4):582–592, 1991.

[109] J. Shi and C. Tomasi. Good features to track. In *Proc. Computer Vision and Pattern Recognition*, 1994.

[110] Silicon Graphics Inc. OpenGL Graphic Capabilities & Ver 1.2 Enhancements. Available at `http://www.opengl.org/About/Capabilities.html`. Accessed April 9, 2000.

[111] I. Sobel. An isotropic 3×3 image gradient operator. In H. Freeman, editor, *Machine Vision for Three-Dimensional Scenes*, pages 376–379. Academic Press, 1990.

[112] T. Starner and A. Pentland. Real-time American sign language recognition from video using hidden Markov models. Technical Report TR-375, M.I.T. Media Laboratory, 1996.

[113] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. Computer Vision and Pattern Recognition*, 1999.

[114] M. Swain and D. Ballard. Color indexing. *Int. J. Computer Vision*, 7(1):11–32, 1991.

[115] Y. Tan, S. Kulkarni, and P. Ramadge. Extracting good features for motion estimation. In *Proc. IEEE Int. Conf. on Image Processing*, pages 117–120, 1996.

[116] C. Taylor, J. Malik, and J. Weber. A real-time approach to stereopsis and lane-finding. In *Proc. IEEE Intelligent Vehicles Symposium*, 1996.

[117] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake and A. Yuille, editors, *Active Vision*, pages 3–20. MIT Press, 1992.

[118] K. Toyama. *Robust Vision-Based Object Tracking*. PhD thesis, Yale University, New Haven, Connecticut, 1997.

[119] K. Toyama and G. Hager. Incremental focus of attention for robust visual tracking. In *Proc. Computer Vision and Pattern Recognition*, pages 189–195, 1996.

[120] A. Watt and M. Watt. *Advanced Animation and Rendering Techniques.* Addison-Wesley, 1992.

[121] Y. Weiss and E. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *Proc. Computer Vision and Pattern Recognition*, pages 321–326, 1996.

[122] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. In *SPIE*, volume 2615, 1995.

[123] C. Wren and A. Pentland. Dynamic modeling of human motion. In *Proc. Int. Conf. Automatic Face and Gesture Recognition*, 1998.

[124] M. Yamamoto and K. Koshikawa. Human motion analysis based on a robot arm model. In *Proc. Computer Vision and Pattern Recognition*, 1991.

[125] A. Yuille and H. Bülthoff. Bayesian decision theory and psychophysics. In D. Knill and W. Richards, editors, *Perception as Bayesian Inference*, pages 123–161. Cambridge University Press, 1996.

[126] S. Zhu, Y. Wu, and D. Mumford. Filters, random fields, and maximum entropy (FRAME): Towards a unified theory for texture modeling. *Int. J. Computer Vision*, 27(2):107–126, 1998.