Abstract

Title of dissertation	Robust and Efficient Inference of Scene and Object Motion in Multi-Camera Systems
	Aswin C. Sankaranarayanan, Doctor of Philosophy, 2009
Directed by	Professor Rama Chellappa Department of Electrical and Computer Engineering

Multi-camera systems have the ability to overcome some of the fundamental limitations of single camera based systems. Having multiple view points of a scene goes a long way in limiting the influence of field of view, occlusion, blur and poor resolution of an individual camera. This dissertation addresses robust and efficient inference of object motion and scene in multi-camera and multi-sensor systems.

The first part of the dissertation discusses the role of constraints introduced by projective imaging towards robust inference of multi-camera/sensor based object motion. We discuss the role of the homography and epipolar constraints for fusing object motion perceived by individual cameras. For planar scenes, the homography constraints provide a natural mechanism for data association. For scenes that are not planar, the epipolar constraint provides a weaker multi-view relationship. We use the epipolar constraint for tracking in multi-camera and multi-sensor networks. In particular, we show that the epipolar constraint reduces the dimensionality of the state space of the problem by introducing a "shared" state space for the joint tracking problem. This allows for robust tracking even when one of the sensors fail due to poor SNR or occlusion.

The second part of the dissertation deals with challenges in the computational aspects of tracking algorithms that are common to such systems. Much of the inference in the multi-camera and multi-sensor networks deal with complex non-linear models corrupted with non-Gaussian noise. Particle filters provide ap-

proximate Bayesian inference in such settings. We analyze the computational drawbacks of traditional particle filtering algorithms, and present a method for implementing the particle filter using the Independent Metropolis Hastings sampler, that is highly amenable to pipelined implementations and parallelization. We analyze the implementations of the proposed algorithm, and in particular concentrate on implementations that have minimum processing times.

The last part of the dissertation deals with the efficient sensing paradigm of compressing sensing (CS) applied to signals in imaging, such as natural images and reflectance fields. We propose a hybrid signal model on the assumption that most real-world signals exhibit subspace compressibility as well as sparse representations. We show that several real-world visual signals such as images, reflectance fields, videos etc., are better approximated by this hybrid of two models. We derive optimal hybrid linear projections of the signal and show that theoretical guarantees and algorithms designed for CS can be easily extended to hybrid subspace-compressive sensing. Such methods reduce the amount of information sensed by a camera, and help in reducing the so called data deluge problem in large multi-camera systems.

Robust and Efficient Inference of Scene and Object Motion in Multi-Camera Systems

by

Aswin C. Sankaranarayanan

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:

Professor Rama Chellappa, Chair
Professor P. S. Krishnaprasad
Professor Adrian Papamarcou
Professor Ankur Srivastava
Professor Amitabh Varshney

To my parents.

Acknowledgments

I would like to express my deepest gratitude to my dissertation advisor, Prof. Rama Chellappa for his wise guidance and mentoring over the years. His passion for research and teaching is very contagious, and has been and continues to be a source of inspiration.

I am very grateful to Prof. Ankur Srivastava, Dr. Ashok Veeraraghavan and Dr. Volkan Cevher for their help in the early parts of my graduate studies. Much of my perspective on research and how to go about it has been shaped from my interactions with them.

I am also grateful to Prof. P. S. Krishnaprasad, Prof. Adrian Papamarcou and Prof. Amitabh Vashney for serving on my dissertation committee, and the time they spent discussing the commenting on my thesis. In particular, I would like to thank Prof. Krishnaprasad for discussing various aspects of my research and instilling a sense of inquisitiveness in me.

Many thanks to the administrative and technical staff at the Center for Automation Research and UMIACS, in particular, Janice Perrone and Janet He.

I thank my friends and fellow graduate students for their support and camaraderie over the year. I, especially, thank Ashwin Swaminathan, Krishna "Jusu" Venkateswara and Mahesh Ramachandran for putting up with my silliness for over a decade; Wan-Yi Lin for making me appreciate the smaller things; Saurabh Srivastava, for providing company for many a lunch; and Chaitanya Deogade and Akshaye Kumar for their friendship.

I would like to express my sincerest appreciation to my family. My parents, Shyamala and Sankaranarayanan, for instilling the basic virtues that I rely on so heavily and being a constant source of support and encouragement. My brothers, Sriram and Jagan for always being there when I need them. To Anandhi and Kamala chittis, and Thyagarajan and Venkataramani chittappas for their guidance, especially during my undergraduate days. Kavita, Siva, Krishna, Chitra

and Indira and for all their companionship and encouragement. Finally, I owe a great debt to my grandparents, Valambal and Dr. Viswanathan, for their love and guiding me by example.

This dissertation owes its source in the aspirations of my parents and is a testament to their committment towards my education. I dedicate this dissertation to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Video cameras are fast becoming ubiquitous for a wide range of applications. Some of the challenges in single camera applications have been studied for over a decade and reliable algorithms for many tasks have been realized. However, in many applications, a single view point provides too little information to reliably solve the problem of interest. This could potentially be due to the limited field of view of the camera, poor resolution with depth, defocus and motion blur, and occlusion. Towards this end, having multiple cameras and additional sensors go a long way in overcoming the fundamental limitations of single camera systems. This dissertation addresses some of the challenges that are central to problems in multi-camera systems.

Much of the work in image understanding has focused on interpreting properties of a scene from images. The distribution of intensities and their spatial arrangement in an image contains information about the identity of objects, their reflectance properties, and the structure of objects in the scene. However, this information is buried in a form that makes it challenging to infer them from the images. One of the fundamental reasons for this difficulty is the fact that the mapping from the 3D scene to a 2D image in general is not invertible. It is in this context that reasonably accurate yet simple geometric models of scene structure (planar scene), scene illumination (point source), surface properties (Lambertian, Phong etc.) and imaging structure (camera models) serve critical roles in the design of inference algorithms. The first part of the dissertation focuses on geometric constraints that are specific to the multi-camera setting, and studies estimation techniques governed by these constraints. This finds use in multi-view tracking

and metrology.

Much of the work in multi-camera systems has been restricted to systems with very few cameras. As the number of cameras increase, data processing and acquisition becomes an overwhelming problem. This leads to the so called *data deluge problem* in systems with hundreds to thousands of cameras. The second part of the dissertation is devoted to the development of methods that scale with number of cameras. We discuss computationally efficient and parallel implementations of a general purpose estimation tool called particle filter. We also study efficient data acquisition schemes that sense minimally to alleviate the high data volumes that needs to be sensed and stored.

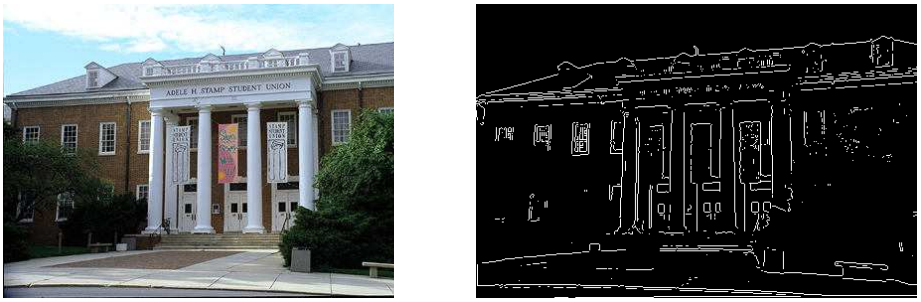
1.1 Motivation

1.1.1 Geometric Modeling and Inference

Image acquisition invariably introduces noise. Common sources of noise in the imaging system are due to shot noise, thermal noise etc. Inference in this noisy environment is further complicated by the inherent errors in physical modeling. Real surfaces are never truly Lambertian, cameras never truly perspective and illumination in a scene never a point light source. Nevertheless, inference algorithms make these assumptions in order to make the problem tractable.

To illustrate these sources of error, let us consider the following simple application. Suppose we are interested in designing a robot that can localize and identify the entrances to buildings (see Figure 1.1(a)). To start with, we first define a 'model' of an entrance. For computational tractability, we assume the edges of the entrance form a rectangle. Now, given an image containing the entrance, we might choose to use an edge-detector or corner-detector to obtain features. Due to image-noise, occlusions, and shadows, the features may not exactly correspond to edge locations. With these noisy feature locations, we proceed to fit two sets of parallel lines, where the lines from different sets are perpendicular to each other.

Consider the edge figure in Figure 1.1(b). Finding the set of points corresponding to the entrance and grouping them into a rectangle is a combinatorial optimization problem. Suppose we do obtain a solution to this optimization problem by some approximate method. The final error in fitting would be in part due to noisy measurements, the difficulty in solving the constrained optimization problem, and the error in the modeling itself since the entrance does not appear as a rectangle due to perspective effects. The error would become even worse when the viewing angle changes further away from frontal, or if there are shadows or occlusion.



(a) (b)

Figure 1.1: Fitting a rectangle to an entrance. Various sources of error are illustrated here – Feature points are noisy, grouping of the points into a rectangle is a challenge, and a rectangle is not an accurate model for the entrance.

As this example illustrates, computer vision algorithms involve the interplay between different geometric constraints that arise from models of the scene. Inference involves assumptions about the imaging apparatus and appropriate statistical estimation techniques that can contend with varying sources of error.

In multi-camera systems, there are two constraints which appear commonly. The first is that of the epipolar constraint, which is a general constraint linking position of objects across views. The epipolar constraint holds with no further assumption except that the cameras imaging the scene are pin-hole cameras. A planar scene provides a stronger constraint, namely, the homography constraint. Much of inference in multi-camera systems stem from an understanding and ju-

dicious use of these two constraints.

The first part of the dissertation deals with statistical estimation under the homography and epipolar constraints.

1.1.2 Computational Challenges in Large Systems

The homography and epipolar constraints are non-linear constraints. Further, the information (or state/parameter) that we seek to estimate is encoded non-linearly in the images that we obtain. Much of the estimation procedure in such a setting can be efficiently formulated as one of non-linear filtering of observations corrupted by noise. In its generality, filtering is the problem of estimation of an unknown quantity, usually referred to as the state, from a set of observations corrupted by noise. This problem has applications in a broad spectrum of real-life problems including GPS navigation, tracking etc. The specific nature of the estimation/filtering problem depends greatly on the state we need to estimate, the evolution of the state with time (if any) and the relation of this state to the observations and the sources of noise. Generally, solutions for estimation can be derived analytically under constrained and special scenarios. For example, Kalman filtering [71] is an optimal analytic filter when the models are linear and the corrupting noise processes are Gaussian. For non-linear systems driven by non-Gaussian processes, the extended Kalman filter or the iterated extended Kalman filter are used as approximations to the optimal filtering scheme. Another popular tool for solving the inference problems for non-linear systems is particle filtering [56, 42].

Particle filtering has been applied to a wide variety of problems such as tracking, navigation, detection [64, 107] and video-based object recognition [142]. This generality of particle filters comes from a sample (or particle) based approximation of the posterior density of the state vector. This allows the filter to handle both the non-linearity of the system as well as the non-Gaussian nature of noise processes. However, the resulting algorithm is computationally intensive and raises

the need for efficient implementations of the algorithm, tuned specifically toward hardware or multi-processor based implementations.

In the second part of the dissertation, we explore algorithmic and implementation schemes for particle filters for speeding up the basic computations, thereby making particle filtering-based solutions amenable to real time constraints. We identify the resampling part of the algorithm as being critical for pipelined and parallel implementations and demonstrate a methodology where there are no bottlenecks in pipelining. Further, this allows us to speedup the filter and reduce its latency through pipelining and parallelization.

1.1.3 Efficient Sensing

Sensing deals with the sampling of a continuous domain signal into a set of discrete samples while preserving critical properties of the signal. The Shannon-Nyquist sampling theorem states that if a signal is band-limited, then it lies in a subspace given by the Fourier basis functions. If a signal lies in a subspace then the signal can be accurately reconstructed by measuring the projections of the signal onto the subspace. The subspace can either be data-independent like the Fourier transform and Wavelet transforms or data dependent like the Principal component analysis (PCA). This basic idea can also be used in data compression, where one can compress the data by projecting the data to a subspace which contains significant energy content of the signal. Since most signals exhibit certain forms of redundancy it is possible to find a subspace such that most of the signal energy is contained within this subspace. More recently, sparse representations and estimation using sparse approximations have gained popularity motivated by recent results from Compressive Sensing (CS) [7, 19, 39]. The basic idea is that if a signal is sparse in some basis, then it may be efficiently recovered and reconstructed using few linear measurements of the signal. Compressive sensing has found applications in much the same domains as the subspace sampling. Visual data such as images, videos, light-fields and reflectance fields which were measured

and compressed using subspace sampling methods can potentially be measured using compressive sensing. In principle, if a signal has no structure in addition to being sparse, then it cannot be subspace limited. Similarly, if a signal has no further redundancy than being subspace limited, then sparsity provides no additional information. This leads to the hypothesis that real visual signals are neither truly sparse, nor just band-limited.

Towards this end, the third part of the dissertation deals with formulating models for visual signals that allow for efficient acquisition. This coupled with computational efficiency and geometric inference forms the three main contributions of this dissertation.

1.2 Contributions

This dissertation makes the following contributions:

1. **Projective transformation of random variables** [119, 115]: We study projective transformations of Gaussian random variables and highlight the influence of the Line at Infinity (of the projective transformation) to the form and statistical properties of the transformed random variable. We use this theory for multi-view tracking and metrology for planar scenes.

2. **Tracking in Joint State Spaces** [29]: We show how the epipolar constraint leads to systems that have overlapping state spaces. We derive a particle filtering framework (including appropriate importance sampling strategies) for such problems, and apply it to multi-sensor tracking problems.

3. **Computationally Efficient Particle Filtering** [118]: We present pipelineable and parallel architectures for implementing particle filters by using the Independent Metropolis Hastings algorithm for resampling. The use of auxiliary variables makes the filter even more flexible in terms of the choice of

importance sampling functions. Finally, we formulate a set of convex programs for obtaining the design specification of the fastest implementation of the algorithm.

4. **Signal Models for Efficient Sensing:** We develop a new signal model called the Hybrid-Subspace-Sparse (HSS) signals which we show approximate several real-world visual signals much better than competing signal models such as subspace and sparse signals. We show that most of the theoretical guarantees obtained for CS can be extended to the HSS signal model but with tighter bounds in terms of the number of measurements needed. We derive the optimal projections for the HSS signals and develop a reconstruction algorithm that amounts to solving a constrained convex optimization, for which fast and stable algorithms exist.

1.3 Organization

Chapter 2 introduces some of the concepts that is the core of this dissertation. Chapters 3 and 4 study statistical estimation for planar scenes. In Chapter 5, we discuss tracking in joint state spaces using acoustic and video sensors as the main example. Computational aspects of particle filters are studied in Chapter 6. We discuss signal models and efficient sensing strategies in Chapter 7. Finally, we highlight future directions of research in Chapter 8.

Chapter 2

Basics

In this chapter, we cover some of the concepts that are central to this dissertation. The chapter is divided mainly into two parts. We first discuss imaging with a pinhole camera and the geometric properties it induces in the context of multiple cameras. The second part of the chapter discusses Bayesian inference in dynamical systems using particle filtering.

2.1 Imaging with a Camera: Projective Geometry

In this section, we introduce the basics of projective geometry and discuss some of the concepts that are extensively used for object detection and tracking. The projective nature of imaging introduces unique challenges in multi-camera systems. It becomes important to understand the nature of such constraints and their impact on these problems. We do note that this is not an exhaustive coverage of this topic. An in-depth discussion of projective geometry can be found in [60] [86].

2.1.1 A Note on Notation and Homogeneous coordinates

In the rest of the dissertation, we use **bold-font** to denote vectors, and CAPS to denote matrices. Further, we use x, y, z alphabets to denote quantities in world coordinates, and u, v alphabets for image plane coordinates. In addition to this, the concept of homogeneous coordinates is important. We use the *tilde* notation to represent entities in homogeneous coordinates. Given a $d-$dimensional vector $\mathbf{u} \in \mathbb{R}^{\mathbf{d}}$, its homogeneous representation is given as a $(d+1)-$dimensional vector $\tilde{\mathbf{u}} \backsim [\mathbf{u}, 1]^T$, where the operator $\backsim$ denotes equality up to scale. In other words,

$\tilde{\mathbf{u}} \backsim \tilde{\mathbf{x}} \Rightarrow \tilde{\mathbf{u}} = \lambda\tilde{\mathbf{x}}, \lambda \neq 0$. In simpler terms, when we deal with homogeneous quantities we allow for a scale ambiguity in our representation. The representation mainly allows for elegant representations of the basic imaging equations, that we discuss next.

2.1.2 Central Projection

Central projection is the fundamental principle behind imaging with a pinhole camera, and serves as a good approximation for lens-based imaging for the applications considered here. In the pinhole camera model, rays (or photons) from the scene are projected onto a planar screen after passing through a pinhole. The screen is typically called the image plane of the camera. Consider a camera with its pinhole at the origin and the image plane aligned with the plane $z = f$. Under this setup, a 3D point $\mathbf{x} = (x, y, z)^T$ projects onto the image plane point $\mathbf{u} = (u, v)^T$, such that

$$u = f\frac{x}{z}, v = f\frac{y}{z}. \tag{2.1}$$

This can be elegantly written in homogeneous terms as,

$$\tilde{\mathbf{u}} \backsim \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \\ 1 \end{pmatrix} \backsim \begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}. \tag{2.2}$$

A more general model of the pinhole camera allows for the pinhole to be at an arbitrary position and the image plane oriented arbitrarily. However, we can use a simple Euclidean coordinate transformation to map this as an instance of the previous one. Finally, the camera might have non-square pixels with image plane skew. This leads us to a general camera model whose basic imaging equation is

given as:

$$\tilde{\mathbf{u}} \backsim K[R\ \mathbf{t}]\tilde{\mathbf{x}} = P\tilde{\mathbf{x}} \tag{2.3}$$

where P is the 3×4 matrix encoding both the internal parameters of the camera K (its focal length, principal point etc.) and the external parameters (its orientation R and position $\mathbf{t}$ in a world coordinate system). $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{x}}$ are the homogeneous coordinate representations of the pixel in the image plane and the point being imaged in the real world respectively. Although central projection is inherently non-linear, it can be written as a linear transformation of the homogeneous coordinates. Finally, (2.2) can be obtained from (2.3) with $R = \mathbb{I}_3$ (the identity matrix), $\mathbf{t} = \mathbf{0}$ and $K = \text{diag(f,f,1)}$.

It is noteworthy that the projection equation of (2.3) is not invertible in general. Intuitively, the pinhole camera maps a 3D world onto a 2D plane, and hence, the mapping is many-to-one and non-invertible. All points that lie on a line passing through the pinhole map onto the same image plane point. This can also be independently verified by the scale ambiguity in (2.3). Given a point on the image plane $\mathbf{u}$, its *pre-image* is defined as the set of all scene points that map onto $\mathbf{u}$ under central projection. It is easily seen, that the pre-image of a point is a line in the real world. Without knowledge of the scene and/or additional constraints, it is not possible to identify the scene point which projects onto $\mathbf{u}$. This lack of invertibility leads to some of the classical problems in computer vision, the most fundamental being establishing correspondence across views.

2.1.3 Epipolar Geometry

Consider two images (or central projections) of a 3D world. Given a point $\mathbf{u}_A$ on the first image of a world point $\mathbf{x}$, we know that its pre-image is a line passing through the point $\mathbf{u}_A$ and C_A, the pinhole of the camera (see Figure 2.1). Hence, given information about $\mathbf{u}_A$ on the first image, all we can establish is that the

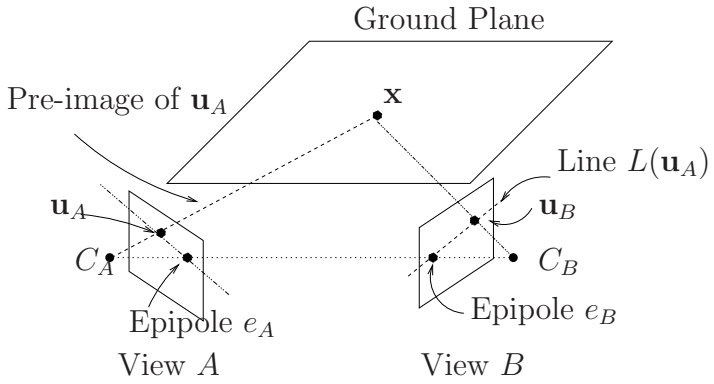


Figure 2.1: Consider views A and B (camera centers C_A and C_B) of a scene with a point $\mathbf{x}$ imaged as $\mathbf{u}_A$ and $\mathbf{u}_B$ on the two views. Without any additional assumptions, given $\mathbf{u}_A$, we can only constrain $\mathbf{u}_B$ to lie along the image of the pre-image of $\mathbf{u}_A$ (a line). However, if world were planar (and we knew the relevant calibration information) then we can uniquely invert $\mathbf{u}_A$ to obtain $\mathbf{x}$, and re-project $\mathbf{x}$ to obtain $\mathbf{u}_B$

corresponding projection of the point $\mathbf{x}$ on the second image plane $\mathbf{u}_B$ lies on the projection of the pre-image of $\mathbf{u}_A$ onto the second image plane. Since the pre-image of $\mathbf{u}_A$ is a line, the projection of this line onto view B gives the line $L(\mathbf{u}_A)$, the *epipolar line* associated with $\mathbf{u}_A$. Thus, the epipolar geometry constrains corresponding points to lie on conjugate pairs of epipolar lines.

In the context of multi-view localization problems, the epipolar constraint can be used to associate objects across multiple views [108]. Once we obtain reliable correspondence across multiple views, we can triangulate to localize objects in the real world. However, correspondences based on epipolar constraint alone tends to be insufficient as the constraint does not map points uniquely across views. In general, all points lying on the epipolar line are potential candidates for correspondence.

2.1.4 Triangulation

In many detection and tracking applications, once we have correspondences between object locations across views, we are interested in localization of these ob-

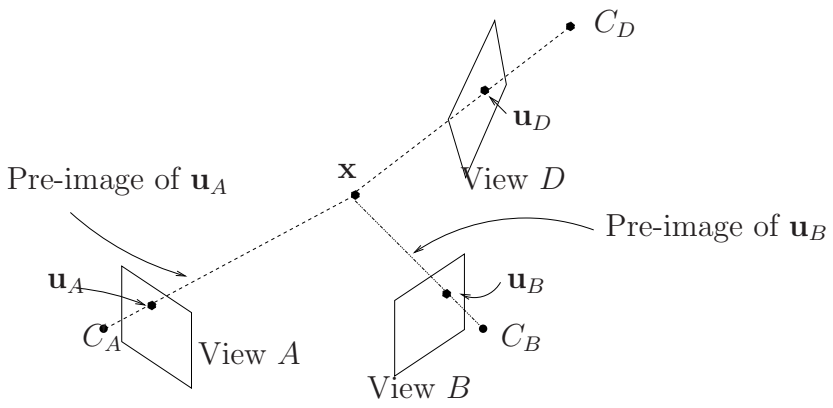


Figure 2.2: Consider views A, B and D of a scene with a point $\mathbf{x}$ imaged as $\mathbf{u}_A$, $\mathbf{u}_B$ and $\mathbf{u}_D$ on the views. We can estimate the location of $\mathbf{x}$ by *triangulating* the image plane points as shown in the figure. At each view, we draw the pre-image of the point, which is the line joining the image plane point and the associated camera center. The properties of projective imaging ensure that the world point $\mathbf{x}$ lies on this pre-image. Hence, when we have multiple pre-images (one from each view) the intersection of these lines gives the point $\mathbf{x}$.

jects in scene coordinates. Let us assume that the same object has been detected in two views (A and B) with camera center C_A and C_B at image place locations $\mathbf{u}_A$ and $\mathbf{u}_B$ as shown in Figure 2.2. In this case, the basics of projective imaging constrains the object to lie on the pre-image of the point $\mathbf{u}_A$ (the line connecting C_A and $\mathbf{u}_A$). Similarly the object must also lie on the pre-image of $\mathbf{u}_B$ in view B. Therefore, estimating the true location of the object amounts to estimating the point of intersection of these two lines. In a general scenario with several cameras, each camera gives rise to a line and estimating the object's location involves computing the point of intersection of these lines. In the presence of noisy measurements, these lines do not intersect at a single point and error measures such as sum of squares are used to obtain a robust estimate of the location of the object. This is called *triangulation* [61]. The drawback of the triangulation approach is that it requires correspondence information across cameras which is difficult to obtain.

2.1.5 Using the Epipolar Constraint

Consider tracking a point from two projective views. The problem can be formed as one of tracking the 3D location of the point. However, to do this the intrinsic as well as the extrinsic camera parameters need to be known. Estimating this from correspondence information results in a projective ambiguity in the estimated parameters [60]. This invalidates many of the noise models used popularly for location tracking.

An alternate way to tracking is in tracking the point in the image plane itself. The epipolar constraint restricts the multi-view projection of a point to lie on corresponding epipolar lines. Hence, we can formulate the tracking problem in a 3 dimensional space $\mathscr{S} = \{\lambda_A, \lambda_B, \alpha\}$. The α parameter chooses the epipolar lines on which the point lies, and λ_A and λ_B determine the location on the epipolar line at each camera with the epipolar marked as origin. Note that α is a parameter that is common to both views, while λ_A and λ_B are completely independent of each other. In a sense, the state space $\mathscr{S}$ can be partitioned into two subsets, $\mathscr{S}_A = \{\lambda_A, \alpha\}$ and $\mathscr{S}_B = \{\lambda_B, \alpha\}$ each of which deal with location in different cameras, and share a common parameter. When tracking extended object characteristics, such as appearance and shape, these add on as independent parameters to the state spaces $\mathscr{S}_A$ and $\mathscr{S}_B$. A study of tracking in state spaces that share a common parameters is extremely useful for using epipolar constraint. In chapter 5, we explore this further and present examples using acoustic video sensors, with the acoustic sensor playing the role of the second projective device.

2.1.6 Planar scenes and Homography

There is one special scenario when the imaging equation becomes invertible, and that is when the world is planar. Most urban scenarios form a good fit as majority of the actions in the world occurs over the ground plane. This makes it a valid assumption for many visual sensing applications. The invertibility can

also be efficiently exploited by algorithms for various purposes. As an example, consider the correspondence problem that we mentioned earlier. Under a planar world assumption, the pre-image of a point becomes a point (in most cases), being the intersection of the world plane and the pre-image line. This implies that by projecting this world point back onto the second image plane, we can almost trivially find correspondence between points on the two image planes. This property induced by the world plane, that allows for finding correspondences across image planes is referred to as the *homography* induced by the plane.

Consider two views of a planar scene labeled View A and View B. We can define a local coordinate system at each view. The same scene point denoted as $\mathbf{x}_A$ and $\mathbf{x}_B$ on the two coordinate systems is related by an Euclidean transformation,

$$\mathbf{x}_B = R\mathbf{x}_A + \mathbf{t}. \tag{2.4}$$

Here, R (a rotation matrix) and $\mathbf{t}$ (a 3D translation vector) define the coordinate transformation from A to B. Lets us assume that the world plane has an equation $\mathbf{n}^T\mathbf{x}_A = d$ with $d \neq 0$[1]. For points that lie on the plane, we can rewrite (2.4) as,

$$\begin{aligned} \mathbf{x}_B &= R\mathbf{x}_A + \mathbf{t}\tfrac{\mathbf{n}^T\mathbf{x}_A}{d} \\ &= \left(R + \tfrac{1}{d}\mathbf{t}\mathbf{n}^T\right)\mathbf{x}_A. \end{aligned} \tag{2.5}$$

In each local camera coordinate system, we know that $\tilde{\mathbf{u}} \backsim K[R\ \mathbf{t}]\tilde{\mathbf{x}}$ (see (2.3)) with $R = \mathbb{I}_3$ and $\mathbf{t} = \mathbf{0}$. Therefore, $\tilde{\mathbf{u}}_B \backsim K_B\mathbf{x}_B$ and $\tilde{\mathbf{u}}_A \backsim K_A\mathbf{x}_A$, which gives us,

$$\begin{aligned} K_B^{-1}\tilde{\mathbf{u}}_B &\backsim \left(R + \tfrac{1}{d}\mathbf{t}\mathbf{n}^T\right)K_A^{-1}\tilde{\mathbf{u}}_A \\ \tilde{\mathbf{u}}_B &\backsim H\tilde{\mathbf{u}}_A, \text{ where } H = K_B\left(R + \tfrac{1}{d}\mathbf{t}\mathbf{n}^T\right)K_A^{-1}. \end{aligned} \tag{2.6}$$

This implies that a point in View A, $\mathbf{u}_A$ maps to the point $\mathbf{u}_B$ on View B as defined by the relationship in (2.6). The 3×3 matrix H (in (2.6)) is called

[1]When $d = 0$, the plane passes through the pinhole at A, thereby making the imaging non-invertible

the homography matrix or just the homography. Also, note that H (like P) is a homogeneous matrix, and the transformation defined by it is unchanged when H is scaled. Further, H is invertible when the world plane does not pass through pinholes at either of the two views. This is easily verified as our derivation is symmetric in its assumptions regarding the two views.

Finally, the premise of the induced homography critically depends on the fact that the pre-image of a point on the image plane is a unique point on the world plane. Suppose we use a local 2D coordinate system over the world plane, the image plane to world plane transformation (from their respectively 2D coordinate systems) can be shown to be a projective transformation, which like before can be encoded as a 3×3 homogeneous matrix, say H_π. This transformation is useful when we want to estimate metric quantities, or quantities in a Euclidean setting. The most common example of this is when we need to localize the target in the scene coordinates.

Computing the image plane to world plane transformation H_π is a challenging problem, that is typically done by exploiting properties of parallel and perpendicular lines on the planes. Typically, this requires manual inputs such as identifying straight lines segments that are parallel. While this not always possible, many urban scenes (such as parking lots, roads, buildings) contain such lines which makes it easier to estimate the transformation H_π, at least in a semi-supervised way. Computing H_π, as it turns out, is identical to a metric rectification of the image plane. Many such techniques are illustrated in [60].

2.1.7 Projective Transformation

A 2D projective transformation $P : \mathbb{P}^2 \mapsto \mathbb{P}^2$ is an invertible mapping such that three points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{P}^2$ are collinear if and only if $P(\mathbf{x}_1), P(\mathbf{x}_2)$ and $P(\mathbf{x}_3)$ are collinear [60]. It has been shown that the definition of projective transformation as given above results in $P(\mathbf{x}) = P\tilde{\mathbf{x}}$ where P is a 3×3 invertible matrix. That is, a projective transformation can be defined in terms of a linear invertible trans-

formation on the homogeneous coordinate representation of points. As mentioned earlier, the homography constraint induced by a plane is a classical example of a 2D projective transformations occurring in practice. Higher dimensional projective transformations can be written as invertible linear transformations as well, the size of the matrix determined by the dimensionality of the space.

While projective transformations are linear in the homogeneous coordinates, they inherently define a non-linear class of transformation given the scale ambiguity that underlies the homogeneous coordinate representation. When the projective transformation is restricted to the real Euclidean space, the transformation can be written as a ratio of affine transformations.

$$P(\mathbf{x}) = \frac{1}{\mathbf{p}_3^T \tilde{\mathbf{x}}} \begin{bmatrix} \mathbf{p}_1^T \tilde{\mathbf{x}} \\ \mathbf{p}_2^T \tilde{\mathbf{x}} \end{bmatrix} \tag{2.7}$$

where $\mathbf{p}_i$ are the rows of the matrix P. The restriction to Euclidean space is important as it ensures $\mathbf{p}_3^T \tilde{\mathbf{x}} \neq 0$. Equation (2.7) is called the *direct linear transformation*, and is identical to the projective transformation when both the domain and range of the transformation exclude ideal points of the projective spaces defining the domain and co-domain.

Finally, the set of point $\{\mathbf{x} \mid \mathbf{p}_3^T \tilde{\mathbf{x}} = 0\}$ is called the Line at Infinity of the transformation P. Physically, the Line at infinity corresponds to the vanishing line or the horizon line defined by planar structures.

2.1.8 Cross Ratios

Consider three collinear points A, B and C in a reference frame Π, and a projective transformation of them labeled A', B' and C' on the line Π' (see Figure 2.3). Given these three point correspondences $(A \leftrightarrow A', B \leftrightarrow B', C \leftrightarrow C')$ there is a unique 1D projective transformation linking the two coordinate frames. One dimensional projective transformations are defined entirely by three unique correspondences.

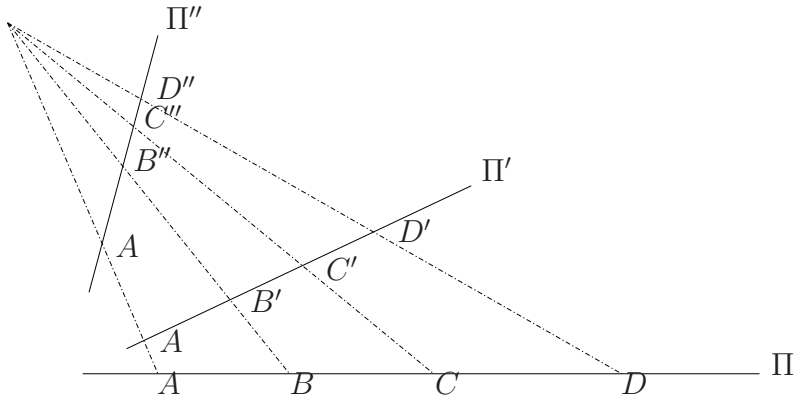


Figure 2.3: The three lines Π, Π′ and Π″ are under central projection, hence related by a 1D homography. Given any three point correspondence, this homography can be uniquely determined. The fourth point can no longer be independently chosen on the lines.

Now given a fourth point D' its location can be computed in the Euclidean frame of reference by inverting the transformation to obtain D. It can be shown that if points A and C are chosen to be origin and the point at infinity respectively, and B a point at unit distance from A, then the value of D forms the cross ratio of the set A', B', C' and D' (or any other projective transformation of the set of points).

Formally, the cross ratio of four collinear points A', B', C' and D' is defined as

$$\frac{|D'A'|}{|D'C'|} / \frac{|B'A'|}{|B'C'|}. \tag{2.8}$$

Here, $|XY|$ denotes the distance between points X and Y. It can be shown that cross ratio of four collinear points is an invariant under any projective transformation of the points.

Cross ratios are extremely useful in various applications in computer vision, given the projective nature of cameras. In particular, it has been extensively used for measuring lengths of objects from images or the metrology [37] problem.

2.1.9 A Note on Calibration

In order to effectively use the epipolar constraint or the homography constraint for images obtained from arbitrary camera views, it is essential to calibrate the various cameras. Calibration, is the process of quantitatively relating the internal pixel based coordinate system inside each camera to a world co-ordinate system. In the case of epipolar geometry, calibration involves estimating the parameters of the camera projection matrix P for each camera. The camera projection matrix P encodes the information about the internal camera parameters like sensor size, focal length, lens distortions etc., and about the external parameters such as location and orientation of the camera with respect to the world co-ordinates. In the case of using the plane-based homography constraint, calibration involves estimating the 3×3 homography matrix, H, which encodes the relation between the image location and the corresponding location on the scene-plane. In either case, calibration is an extremely well-studied problem and we refer the readers to [60] [86] for an in-depth analysis of the methods and issues involved in calibration.

2.2 Inference in Dynamical Systems

Dynamical systems provide a structured representation for both the nature of the temporal variations, and the relationship between observations and time varying parameter. The elegance of dynamical systems come in the form of generic inference algorithms, that can be suitably chosen and tailored to the specifics of the problem domain.

In particular, we formulate the problem of Bayesian inference for dynamical systems. Let $\mathcal{X} \subseteq \mathcal{X}$ and $\mathcal{Y} \subseteq \mathcal{Y}$ denote the state space and the observation space of the system respectively. Let $x_t \in \mathcal{X}$ denote the state at time t, and $y_t \in \mathcal{Y}$ the noisy observation at time t. We model the state sequence $\{x_t\}$ as a Markovian random process. Further we assume that the observations $\{y_t\}$ to be conditionally independent given the state sequence. Under these assumptions, the

system is completely characterized by the following:

- $p(x_t|x_{t-1})$: The state transition density, describing the evolution of the system from time $t-1$ to t. Alternatively, the same could be described with a *state transition model* of the form $x_t = h(x_{t-1}, n_t)$, where n_t is a noise process.

- $p(y_t|x_t)$: Observation likelihood density, describing the conditional likelihood of observation given state. As before, this relationship could be in the form of an *observation model* $y_t = f(x_t, \omega_t)$ where ω_t is a noise process independent of n_t.

- $p(x_0)$: The prior state probability at $t = 0$.

Given statistical descriptions of the models and noisy observations, we are interested in making inferences about the state of the system at current time. Specifically, given the observations till time $t, y_{1:t} = \{y_1, \ldots, y_t\}$, we would like to estimate the posterior density function $\pi_t = p(x_t|y_{1:t})$. With the posterior, we aim to make inferences $I(f_t)$ of the form,

$$I(f_t) = \mathbf{E}_{\pi_t}[f_t(x_t)] = \int f_t(x_t)p(x_t|y_{1:t})dx_t \tag{2.9}$$

where f_t is some function of interest. An example of such an inference could be the conditional mean, where $f_t(x_t) = x_t$.

Under the Markovian assumption on the state space dynamics and the conditional independence assumption on the observation model, the posterior probability $\pi(\mathbf{x}_t)$ is recursively estimated using the *Bayes Theorem*

$$\pi_t(x_t) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})} = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_{t-1}}. \tag{2.10}$$

The computation of $p(x_t|y_{1:t-1})$ sets up the premise for recursion and is called

the *prediction* step,

$$p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}. \tag{2.11}$$

Note that

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)\int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}}{p(y_t|y_{1:t-1})} \tag{2.12}$$

has no unknowns since all terms are either specified or computable from the posterior at the previous time step. The problem is that this computation (including the integrations) need not have an analytical representation.

Analytical solutions exist when the state transition model and the observation model are both linear, and we are interested in tracking only the first two moments (equivalently, the mean and the covariance matrix) of the posterior density function. In this setting, the optimal estimator is the *Kalman filter* [71]. For non-Gaussian models, the Kalman filter is optimal among the class of linear filters.

In many applications, we are interested in higher order statistics, that capture key properties of the posterior density, such as multi-modality and non-linear evolution. There are two cases when the general inference problem can be efficiently solved. When the state space $\mathcal{X}$ is a finite set, the dynamical system can be compactly modeled as a finite state machine, and the inference problem can be solved efficiently as well. The other scenario is when we allow for approximate inference, leading to a class of Sequential Monte-Carlo techniques also known as particle filters [84, 43, 56]. We discuss these next.

2.2.1 Finite State Machines: Hidden Markov Models

When the state space $\mathcal{X}$ is a finite set $\{1, \ldots, M\}$, we can compactly represent the state transition model in terms of a $M \times M$ matrix. Suppose we define $\Pr(x_t = j|x_{t-1} = i) = p_{ij}^t$, such that $0 < p_{ij}^t < 1$. The matrix $P(t) = [p_{ij}^t]$ is the

$M \times M$ state transition matrix must satisfy

$$\sum_j p_{ij}^t = 1 \iff P(t)\mathbf{1} = \mathbf{1} \tag{2.13}$$

implying that the all one vector is one of its eigenvector with unit eigenvalue.

The posterior probability mass function[2] $\pi_t \in \mathbb{R}^M$ such that $\pi_t^T \mathbf{1} = 1$ can be recursively estimated. The vector denoting the probability mass of $Pr(x_t|y_{1:t-1})$ is given as $P(t)^T \pi_{t-1}$. Let $L_y \in \mathbb{R}^M$ be the vector characterizing the likelihoods, such that its i-th component is $Pr(y_t|x_t = i)$. Using simple algebra and Bayes' theorem it is possible to show that the posterior π_t is given as,

$$\pi_t = \tau \text{diag}(L_y) P(t)^T \pi_{t-1} \tag{2.14}$$

where τ is a normalizing constant that ensures that π_t is a valid mass function, its components summing up to unity.

While estimating the posterior mass function is straight-forward in the case of a finite state space, in many cases, it is time consuming to evaluate the posterior mass completely. In such cases, we are interested in inferring only the state sequence with the highest likelihood or the Maximum A Posteriori (MAP) state sequence. The *Viterbi* algorithm [110, 49] allows the computation of the MAP state sequence very efficiently. However, we restrict ourself to Bayesian inference of the entire posterior density. We direct the interested reader to an excellent review of hidden Markov models by Rabiner and Juang [109].

2.2.2 Particle Filters

Particle filters address Bayesian inference for general non-linear non-Gaussian dynamical systems. As mentioned earlier, inference for such systems are in general analytically intractable. However, foregoing the requirement for an analytic solu-

[2]The posterior probability density in this setting becomes a PMF as the state space is discrete

tion, particle filtering approximates the posterior π_t with a discrete set of particles or samples $\{x_t^{(i)}\}_{i=1}^N$ with associated weights $\{w_t^{(i)}\}_{i=1}^N$ suitably normalized so that $\sum_{i=1}^N w_t^{(i)} = 1$. The approximation for the posterior density is given by

$$\hat{\pi}_t(x_t) = \sum_{i=1}^{N} w_t^{(i)} \delta_{x_t}\left(x_t^{(i)}\right) \tag{2.15}$$

where $\delta_{x_t}(\cdot)$ is the Dirac Delta function centered at x_t. The set $S_t = \{x_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ is the weighted particle set that represents the posterior density at time t, and is estimated recursively from S_{t-1}. The initial particle set S_0 is obtained from sampling the prior density $\pi_0 = p(x_0)$.

We first discuss the so called *importance function* $g(x_t|x_{t-1}, y_t)$, an easy-to-sample function whose support encompasses that of π_t. The estimation of $I(f_t)$, as defined in (2.9) can be recast as follows,

$$\begin{aligned} I(f_t) &= \int f_t(x_t) \tfrac{p(x_t|y_{1:t})}{g(x_t|x_{t-1},y_t)} g(x_t|x_{t-1}, y_t) dx_t \\ &= \int f_t(x_t) w(x_t) g(x_t|x_{t-1}, y_t) dx_t \end{aligned} \tag{2.16}$$

where $w(x_t)$ is referred to as *importance weight*,

$$w_t = \frac{p(x_t|y_{1:t})}{g(x_t|x_{t-1}, y_t)}. \tag{2.17}$$

Particle filters sequentially generate S_t from S_{t-1} using the following steps,

1. **Importance Sampling:** Sample $x_t^{(i)} \sim g(x_t|x_{t-1}^{(i)} y_t), i = 1, \ldots, N$. This step is also called the *proposal step* and $g(\cdot)$ is sometimes called the *proposal density*.

2. **Computing Importance Weights:** Compute the unnormalized impor-

tance weights $\tilde{w}_t^{(i)}$,

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} \frac{p(y_t|x_t^{(i)})}{g(x_t^{(i)}|x_{t-1}^{(i)}, y_t)}, \ i = 1, \ldots, N. \tag{2.18}$$

3. **Normalize Weights:** Obtain the normalized weights $w_t^{(i)}$,

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}, \ i = 1, \ldots, N. \tag{2.19}$$

4. **Inference Estimation:** An estimate of the inference $I(f_t)$ is given by

$$\hat{I}_N(f_t) = \sum_{i=1}^{N} f_t(x_t^{(i)}) w_t^{(i)} \tag{2.20}$$

This sequence is performed for each time iteration to get the posterior at each time step. A basic problem that the above algorithm suffers from is that, after a few time steps, all importance weights except a few go to zero. These weights will remain at zero for all future time instants (as a result of (2.18)), and do not contribute to the estimation of $\hat{I}_N(f_t)$. Practically, this degeneracy is undesirable and is a waste of computational resource. This is avoided with the introduction of a *resampling* step. Resampling essentially replicates particles with higher weights and eliminates those with low weights. This can be done in many ways. [85,43,56] list many resampling algorithms. The most popular one, originally proposed in [56], samples N particles from the set $\{x_t^{(i)}\}$ (samples generated after proposal) according to the multi-nomial distribution with parameters $w_t^{(i)}$ to get a new set of N particles $\tilde{S}_t$. The next iteration uses this new set $\tilde{S}_t$ for sequential estimation. We discuss some additional sampling algorithms in Section 2.2.4.

2.2.3 Choice of Importance Function

Crucial to the performance of the filter, is the choice of the importance function $g(x_t|x_{t-1}, y_t)$. Ideally, the importance function should be close to the posterior. If we choose $g(x_t|x_{t-1}, y_t) \propto p(y_t|x_t)p(x_t|x_{t-1})$, then we would obtain the importance weights w_t identically equal to $1/N$ and the variance of the weights would be zero. For most applications, this density function is not easy to sample from. This is largely due to the non-linearities in the state transition and observation models. One popular choice is to use the state transition density $p(x_t|x_{t-1})$ as the importance function. In this case, the importance weights are given by

$$w_t \propto w_{t-1}p(y_t|x_t) \tag{2.21}$$

Other choices include using cleverly constructed approximations to the posterior density [41].

2.2.4 Resampling Algorithms

In the particle filtering algorithm, the resampling step was introduced to address degeneracies resulting due to the importance weights getting skewed. Among resampling algorithms, the SR technique is popularly used. The basic steps of SR [85] are recounted below.

- For $j = 1, \ldots, N$
  1. Sample $J \sim \{1, \ldots, N\}$, such that $Pr[J = i] = a^{(i)}$, for some choice of $\{a^{(i)}\}$.
  2. The new particle $\tilde{x}_t^{(j)} = x_t^{(J)}$ and the associated weight is $\tilde{w}_t^{(j)} = w_t^{(J)}/a_t^{(J)}$.
- The resampled particle set is $\tilde{S}_t = \{\tilde{x}_t^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^N$.

If $a^{(i)} = w_t^{(i)}$ the resampling scheme is the one used in [56]. Other choices are discussed in [85].

Particle filtering algorithms that use Sequential Importance Sampling (SIS) and SR are collectively called SISR algorithms. Computationally, SR is a tricky step, as it requires the knowledge of the normalized weights. Resampling based on SR cannot start until all the particles are generated and the value of the cumulative sum is known. This is the basic limitation that we overcome by proposing alternative techniques. We address the computational aspects of particle filters in detail in Chapter 6.

Chapter 3

Projective Transformation of Random Variables

Many problems in computer vision involve estimation of real world metrics. For an ideal pinhole camera, the scene to image plane transformation is a projective transformation. Inverting this transformation, in the presence of additional constraints forms the core of many vision tasks. For example, in structure from motion, constraints such as the rigid body motion of the scene/camera are employed to obtain the scene structure. In calibration problems, the assumption of a dominant plane allows for the recovery of intrinsic and extrinsic camera parameters.

The geometric properties of the projective transformation have been well studied [60, 86]. In particular, the projective transforms of simple objects like points, lines, planes and conics are well understood. Calibration, tracking, detection, structure from motion and metrology represent few applications where the properties of projective transformations of simple objects are studied. However, the geometric theory analyzing the properties of the projective transformation is usually under the assumption of noise-free measurements, or employs heuristics to account for noise. A rigorous and formal characterization of statistical estimation and projective transformations has not been adequately developed.

In this chapter, we study how properties of random variables change under projective transformations. As an example, consider the problem of location estimation on a plane using inputs from multiple cameras. To simplify the problem, let us assume that the object under consideration is a point object (say the corner of a square). At each camera, we use a feature point tracker to get an estimate of where the corner lies. We are interested in estimating the location of this point in

the world coordinate system. It is well known that planar scenes induce a homography or a 2D invertible projective transformation between the image planes of cameras and the ground plane. Given this homography, we can project the image plane location to arrive at individual estimates of the ground plane location of the tracked point. In an ideal noise-free scenario, the estimates arising from each of the cameras would be identical. However, in the presence of noise - corrupting the image plane observations leading to errors in calibration and inaccuracies in modeling, the ground plane location estimates will no longer be identical. We need a rigorous method to fuse these estimates. The same problem arises in a host of other problems, mainly dealing with the estimation of real world metrics like lengths and location from multi-view inputs (see Figure 3.1). Common to these problems is that they involve a projective transformation of noisy image plane measurements.

Before we devise a strategy for fusion, we need to study the properties of the estimates produced by each camera. Clearly, the statistical properties of the estimate from a given camera are influenced by the error in tracking of the object on its image plane. However, a stronger influence is played by the homography transformation between the image plane of the camera and the ground plane. The reason for this is as follows. The homography transformations are different for the different views, given that each camera is guaranteed to have different external parameters with respect to the world coordinate system. Hence, each image plane estimate is generated by a different homography. Therefore, even if the properties of the noise on the image plane are identical across camera views, the statistical properties of the transformed estimate are in general different. A clever fusion scheme must necessarily account for the different statistical properties of the individual estimates.

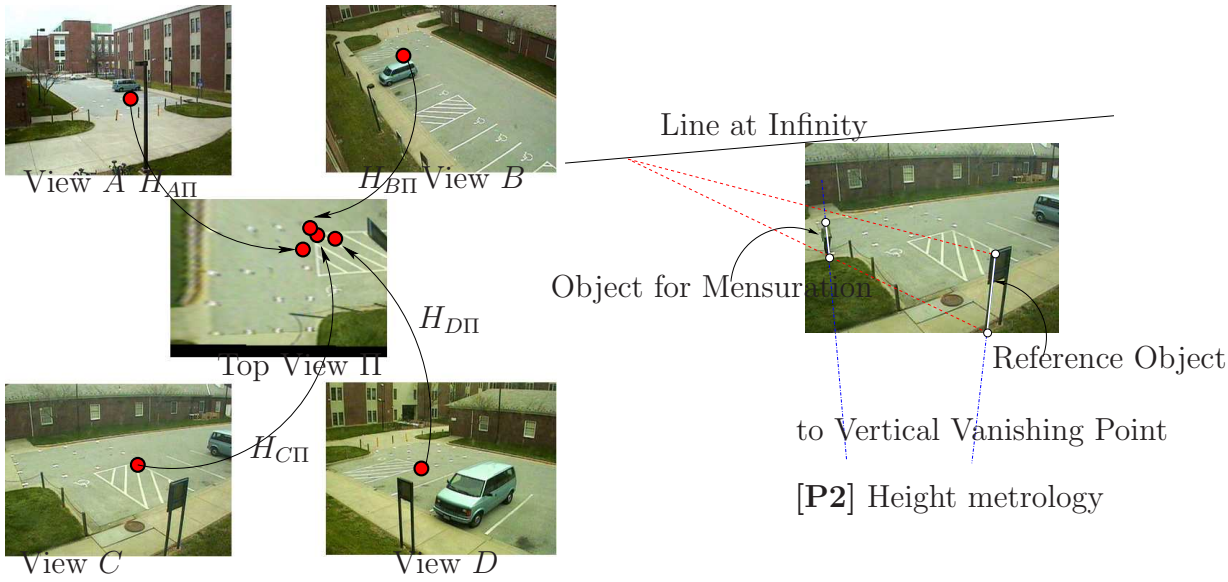


Figure 3.1: Graphical depiction of some problems that critically involve projective transformations. (Left) [P1] Multi-view fusion of object locations on a plane can possibly involve image plane location estimates projected through different transformations. Fusion of these ground plane estimates needs knowledge of the statistics of the transformed estimates. (Right) [P2] Metrology of object heights using cross-ratios involves a one-dimensional projective transformation.

3.1 Prior Work

The ideas presented in this chapter come under the paradigm of statistical estimation under geometric constraints. Kanatani [72] pioneered the research of statistical estimation using the structure induced by geometry for various estimation problems in 2D and 3D. [72] discusses the role of geometric constraints in solving line, plane and conic fitting problems using covariance matrices for error characterization. In [36], Criminisi uses similar error characterization (in terms of covariance matrices) for metrology in multi-view settings. In [50], the choice of representation for projective entities is considered in light of the earlier works by Kanatani and others. Linearization approaches for propagating covariance matrices through projective transforms is used in [98] for localization and mosaicing. The use of covariance matrices for error characterization and the propagation of these covariances using first order approximations (linearization) of the transformation are discussed in these papers. However, the effect of projective transformations on random variables (characterized in terms of probability density functions) has not been studied. Interestingly, we also show that projectively transformed random variables can have moments that are undefined/infinite, making error characterization using covariances highly inaccurate.

Hartley [61] addresses the problem of two-view triangulation for the case of Gaussian noise corrupting image plane measurements. Traditionally, the 3D location of a point is obtained by minimizing a cost function over the world coordinates. The authors argue that such a cost function might not be appropriate if the scene reconstruction deviates from Euclidean by a projective transformation and suggest optimization over image plane coordinates where the probability distributions are more meaningful.

A closely related topic is how projective invariants vary under noise. The most commonly used projective invariant is the *cross-ratio*. The properties of the cross ratio have been studied by Astrom [3] for uniform noise corrupting the image

plane observations and by Maybank [92] for the case of Normal distribution.

[10], [93] describe methods to define measures and densities on the space of homography matrices. This, however is different from the study of how random variables transform under a projectivity. To the best our knowledge, such a study has not been considered.

The derivation of the necessary analytical results for our applications relies heavily upon prior work in the area of statistics on ratio distributions. The earliest work (to the best of our knowledge) was by Geary [52] in which it is shown that if the support of the denominator for negative values is small, a suitably transformed version of the ratio can be assumed to be Normal. Marsaglia's work [89] [90] on this topic is also noteworthy for many reasons, foremost as it was the first to derive the distribution of the Normal ratio. Further, Marsaglia also shows that the ratio of correlated Normals can be easily obtained from ratio of independent Normals. We borrow extensively the notations and derivations used in Marsaglia's work. More recent work on the same topic can be found in [103] [23] [24].

Many tracking algorithms have been proposed that use a planar scene assumption to track objects in the real world. The single-camera multiple-human tracking algorithm in [140] uses the ground plane motion constraint to estimate foot location on the ground plane to decide depth ordering. The algorithm in [139] uses a Kalman filter to track the location and velocity on the plane with the observation noise model obtained by linearization of the homography between the image and the ground plane. An alternate single camera tracker using the homography to project head and feet positions to the ground plane is described in [47]. Multi-camera tracking algorithms [48, 73, 75] also use homographies to project inputs from background subtraction onto the ground plane. Typically, data association and targets localization are achieved through consensus among projections from the cameras.

As an example, in [75], the vertical axis from each segmented human is extracted in each view and the intersection of their projections on the ground plane

is used to localize the person. The intersection points are then filtered using a particle filter. In contrast, the algorithm proposed in [73] achieves consensus by projecting foreground likelihood images from each view to a reference view. The likelihood images achieve consensus for foreground objects that lie on the plane, while those in parallax with respect to the plane do not register under projection. Peaks in the joint likelihood image correspond to objects on the plane and are thresholded and segmented using a graph-cut algorithm. The main focus of these algorithms is on maintaining data association across views and robust tracking across occlusion. *All these algorithms treat inputs from different views identically, and the experimentation has been over views that are similar.* The models do not explicitly consider the effect of highly asymmetrical camera placement on estimation accuracy.

In this chapter, we study how projective transforms affect distributions of random variables and suggest methods to incorporate such models for various applications in multi-view estimation. We show that even a Normal random variable transforms to a mixture density containing a Cauchy component. This results in distributions that do not have well defined moments. However, when the region of interest that is imaged is far away from the Line at infinity (of the transformation) the transformed random variable can be assumed to have well defined moments. This allows us to provide a theoretical basis for understanding the poor performance of ground plane tracking algorithms near the horizon line. We also analyze existing observation models in multi-view tracking literature, and propose suitable modifications that allow for efficient fusion of multi-view information.

3.2 Problem Statement

Figure 3.2 shows an example of three cameras A, B and C looking at a plane Π, with the image plane of B parallel to Π. In contrast, the image planes of A and C are perpendicular to Π. Also shown on the image planes of the cameras

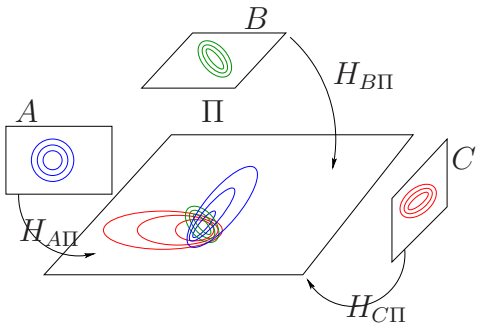


Figure 3.2: A schematic showing densities on the image planes of cameras and their transformations to the ground plane.

are iso-error contours representing the image plane distribution at each camera. The homographies between the cameras and the plane Π are $H_{A\Pi}$, $H_{B\Pi}$ and $H_{C\Pi}$ respectively. In this configuration, $H_{B\Pi}$ is not fully projective, defining only an affine transformation, as opposed to $H_{A\Pi}$ and $H_{C\Pi}$ which induce strong projective distortion. We would expect the density on B to retain its original form (similar error iso-contours) when projected on the plane.

We first consider a *static* estimation problem. Suppose that we have a setup with M cameras labeled $1, \ldots, M$. Let $\mathbf{u}_i, i = 1, \ldots, M$ be the location estimate on the image plane of each camera. By applying the corresponding homography between the i-th camera and the ground plane, we obtain $\mathbf{x}_i = H_{i\Pi}(\mathbf{u}_i)$, the estimate of location on the ground plane. Given $\mathbf{x}_i, i = 1, \ldots, M$ the estimates from M cameras simultaneously observing a target on the plane, we would like to fuse them. Figure 3.2 suggests that the $\mathbf{x}_i$'s are not identically distributed, *even if* $\mathbf{u}_i$ *are identically distributed*. Using the sample mean $\mathbf{x} = (1/M)\sum_{i=1}^{M} \mathbf{x}_i$ might not be optimal (say, in the minimum variance sense). A desirable estimator or fusion scheme would use the distribution of the $\mathbf{x}_i$'s appropriately. This requires a careful study of how random variables transform under projective transformations.

An identical problem arises when one uses dynamical systems to filter multi-view inputs. Consider a dynamical system with a state space defined on the location of a point on the ground plane Π. The image of the point on the various

image planes is governed by the basic imaging equation,

$$\begin{aligned} \mathbf{y}_i(t) &= H_{\Pi i}(\mathbf{x}(t)) + \mathbf{n}_i(t), i = 1, \ldots, M \\ &= \mathbf{u}_i(t) + \mathbf{n}_i(t), \text{ where } \mathbf{u}_i = H_{\Pi i}(\mathbf{x}) \end{aligned} \tag{3.1}$$

We further assume that the noise processes $\{\mathbf{n}_i(t), i = 1, \ldots, M\}$ are identically distributed as Normal random processes. This model is the standard imaging equation for a point object and has an intuitive appeal. However, the non-linearity of the equation makes it challenging for both analytical and computational inference.

Suppose we apply the transformation $H_{i\Pi}$ directly to (3.1), then we get the following expression.

$$\mathbf{z}_i(t) = H_{i\Pi}(\mathbf{y}_i(t)) = H_{i\Pi}\left(\mathbf{u}_i(t) + \mathbf{n}_i(t)\right) \tag{3.2}$$

Further insight into the expression for $\mathbf{z}_i(t)$ in (3.2) requires the understanding of how Normal random variables (such as $(\mathbf{u}_i(t) + \mathbf{n}_i(t))$) change under projective transformations.

Existing ground plane tracking algorithms [75] [47] circumvent this by assuming that $\mathbf{z}_i$ forms a complete observation of the state, i.e,

$$\mathbf{z}_i(t) = H_{i\Pi}(\mathbf{y}_i(t)) = \mathbf{x}(t) + \eta_i(t) \tag{3.3}$$

where $\eta_i(t)$ is a noise process.

Note that (from Figure 3.2) densities change their properties dramatically when projectively transformed. This is reflected in the statistical properties of the noise process η_i. However, analysis and computation of the parameters characterizing this is not straightforward. Further, it is imperative to study if (3.3) approximates (3.2) accurately, and the conditions when this is true.

3.3 Projective Transformation of Random Variables

Consider the projective transformation[1], $H : \mathscr{U} = \mathbb{P}^n \rightarrow \mathscr{X} = \mathbb{P}^n$, i.e, H is given is an $(n+1) \times (n+1)$ matrix defining the transformation $H : \mathbf{u} \mapsto \underline{\mathbf{x}} = H\underline{\mathbf{u}}$. We denote the i-th row of H as $\mathbf{h_i} \in \mathbb{R}^{n+1}$ ($H = [\mathbf{h_0}, \mathbf{h_1}, \ldots, \mathbf{h_n}]^T$).

At this point, it is useful to point out that H is a homogeneous matrix, in the sense that the transformation defined by it is unchanged if we scale it using a non-zero scale. It is often useful to resolve this ambiguity of scale explicitly. We assume that $\det(H) = 1$. This performs a scale normalization that allows us to define the transformation uniquely without having to consider scale ambiguities and simplifies the rest of discussion considerably. Finally, we assume that H is deterministic and is error-free.

The relation between $\mathbf{u}$ and $\mathbf{x}$ is linear in their homogeneous form. When $\mathbf{u}$ and $\mathbf{x}$ are finite points, this relationship can be recast as a non-linear equation

$$\mathbf{x} = \frac{1}{\mathbf{h_n^T}\underline{\mathbf{u}}} \left[\mathbf{h_0^T}\underline{\mathbf{u}}, \cdots, \mathbf{h_{n-1}^T}\underline{\mathbf{u}}\right]^T \tag{3.4}$$

Given a random variable $\mathbf{Z_U}$ whose probability density function (PDF) is known, we are interested in the random variable $\mathbf{Z_X}$ given by the relation,

$$\underline{\mathbf{Z_X}} \sim H\underline{\mathbf{Z_U}}, \mathbf{Z_X} = \begin{bmatrix} Z_x^0 \\ \vdots \\ Z_x^{n-1} \end{bmatrix} = \frac{1}{\mathbf{h_n^T}\underline{\mathbf{Z}}_\mathbf{U}} \begin{bmatrix} \mathbf{h_0^T}\underline{\mathbf{Z}}_\mathbf{U} \\ \vdots \\ \mathbf{h_{n-1}^T}\underline{\mathbf{Z}}_\mathbf{U} \end{bmatrix} \tag{3.5}$$

To proceed further, we need to know the distribution of $\mathbf{Z_U}$. The exact nature of the distribution depends heavily on the application and specifics of the origin of $\mathbf{Z_U}$. As an example, $\mathbf{Z_U}$ might represent the noisy location of a feature point on the image plane, and $\mathbf{Z_X}$, the projection of this point on to a world plane with H

[1] $\mathscr{U}$ and $\mathscr{X}$ are used to identify the domain and the co-domain/range respectively. For an intuitive understanding, it is useful to think of $\mathscr{U}$ as lines/planes on the image plane of a camera and $\mathscr{X}$ as those on real world (euclidean space).

representing the homography between the image-world planes. The distribution associated with $\mathbf{Z_U}$ could possibly be obtained from the underlying tracking algorithm, say a Normal fit if a Kalman filter is employed or a non-parametric density if a particle filter is used. However, analytic tractability is not guaranteed for arbitrary choices for the distribution. The transformation defined by (3.5) involves ratios of terms involving $\mathbf{Z_U}$. This non-linearity of the transformation makes analytic tractability and inference difficult for complicated image plane densities. For this reason, we assume that the distribution of $\mathbf{Z_U}$ is multi-variate Normal.

$$\mathbf{Z_U} \sim \mathrm{N}\left(\mathbf{m_u}, S_u\right), \mathbf{m_u} = \left[m_0, \cdots, m_{n-1}\right]^T \tag{3.6}$$

We also assume that the covariance matrix S_u is a constant, and does not change with the mean of the random variable $\mathbf{Z_U}$.

$\mathbf{Z_U}$ is assumed to have a Normal distribution, implying that $\mathbf{h_i}^T\underline{\mathbf{Z}}_\mathbf{U}, i = 0, \ldots, n$ are univariate Gaussian.

$$\begin{gathered}\mathbf{h_i}^T\underline{\mathbf{Z}}_\mathbf{U} \sim N(\mu_i, \sigma_i^2), i = 0, \ldots, n \\ \mu_i = \mathbf{h_i}^T\underline{\mathbf{m}}_\mathbf{u}, \sigma_i^2 = \mathbf{h_i}^T\hat{S}_u\mathbf{h_i}\end{gathered} \tag{3.7}$$

where $\hat{S}_u$ is the $n+1 \times n+1$ matrix obtained by adding a row and a column of zeros to S_u. Further, the correlation ρ_{ij} between $\mathbf{h_i}^T\underline{\mathbf{Z}}_\mathbf{U}$ and $\mathbf{h_j}^T\underline{\mathbf{Z}}_\mathbf{U}$ for $i \neq j$ is given as:

$$\rho_{ij} = \mathbf{h_i}^T\hat{S}_u\mathbf{h_j}/(\sigma_i\sigma_j) \tag{3.8}$$

From (3.5) and (3.7), the scalar random variables $Z_x^i, i = 0, \ldots, n-1$ arise from ratios of Normal densities. A detailed characterization of ratio of Normal densities can be found in [89,90]. A detailed derivation in the context of projective transformations can also be found in [115]. We summarize the basic results below.

The distribution of Ratios of Normals is a complicated expression [89,115], involving Cauchy components. The presence of the Cauchy component implies that

the mean, variance and in general, higher moments for Z_x^i are not defined [22]. Hence, theorems on weak/strong laws of large numbers are not applicable. Thus, estimates such as the sample mean will not converge asymptotically. Typically, inference in the presence of Cauchy distribution is done using the maximum likelihood estimate (MLE) [59], the median or mode. However, when the Cauchy component is weak, we can ignore the presence of the Cauchy distribution and approximate the density. However, this approximation makes sense when the denominator random variable $\mathbf{h}_n^T \underline{\mathbf{Z}}_\mathbf{U}$ has a mean that is far away from the origin. We first study the geometric setting when this happens.

We begin with expressing Z_x^i as the ratio:

$$\begin{aligned} Z_x^i = \frac{\mathbf{h}_i^T \underline{\mathbf{Z}}_\mathbf{U}}{\mathbf{h}_n^T \underline{\mathbf{Z}}_\mathbf{U}} &= \frac{\sigma_i Z_i + \mu_i}{\sigma_n Z_n + \mu_n}, \text{ where } Z_i \backsim N(0,1), Z_n \backsim N(0,1) \\ &= \frac{\sigma_n}{\sigma_i} \frac{Z_i + \mu_i/\sigma_i}{Z_n + \mu_n/\sigma_n} \end{aligned} \tag{3.9}$$

Prior work on ratio distributions tell us that the above density is well behaved (in terms of existence of moments) only when the denominator has negligible support around the origin. Mathematically, this happens when the mean of the denominator is far away from zero, or equivalently, $|\mu_n/\sigma_n| \gg 0$.

Geometrically, $|\mu_n|$ is directly proportional to the distance of the mean of the image plane distribution $\mathbf{Z}_U$ from the Line at Infinity of the plane on the image plane. We discuss this detail next.

3.3.1 Connections to the Line at Infinity

Any projective transformation $H = [h_0, h_1, \ldots, h_n]^T$ can be written as the product of an affine transformation H_A and a purely projective transformation H_P [60],

such that,

$$H = H_A H_P = \begin{bmatrix} & \mathbf{h}_0^T & \\ & \vdots & \\ & \mathbf{h}_{n-1}^T & \\ 0 & \cdots \quad 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ \vdots & \vdots & \cdots & \vdots \\ & \mathbf{h}_n^T & & \end{bmatrix} \tag{3.10}$$

This suggests the following properties for the matrix H: 1) All the projective distortions of the transformation are encoded in its last row $\mathbf{h}_n$, and 2) The set of all point which under transformation map to ideal points (or points at infinity) satisfy the relationship $\mathbf{h}_n^T \underline{\mathbf{u}} = 0$. This set of points forms a line (or an $n-1$ dimensional hyperplane, in general) and is referred to as the Line at Infinity of the ground plane. Hence $\mu_n = \mathbf{h}_n^T \underline{\mathbf{m}}_\mathbf{u}$ is the signed distance of the point $\mathbf{m}_\mathbf{u}$ from the Line at Infinity. *This implies that when the imaged point has a mean that is far away from the Line at Infinity, then the distribution of the transformed random variable is well-behaved, in the sense that its moments exist.*

More formally, the condition that

$$\frac{|\mu_n|}{\sigma_n} = \frac{|\mathbf{h}_n^T \underline{\mathbf{m}}_\mathbf{u}|}{\sigma_n} \gg 1 \tag{3.11}$$

ensures that the random variable Z_x^n is well-conditioned in the sense that its density can be closely approximated with a Gaussian density and its moments said to exist in a pseudo-sense. To show this, we need to show that denominator random variable is bounded away from zero, such that, that the probability of the random variable taking values near zero is negligibly small. This result comes directly from (3.11).

From Chebyshev's inequality,

$$Pr\left(\frac{|\mathbf{h}_n^T \underline{\mathbf{Z}}_U - \mu_n|}{\sigma_n} > c\right) \leq \frac{1}{c^2}, c > 0 \tag{3.12}$$

Assuming $\mu_n > 0$[2], and using the symmetry of the Gaussian density around its mean,

$$Pr(\mathbf{h}_n^T \underline{\mathbf{Z}}_U < -c\sigma_n + \mu_n) \leq \frac{1}{2c^2} \tag{3.13}$$

Letting $c = \mu_n/\sigma_n$, we get

$$Pr(\mathbf{h}_n^T \underline{\mathbf{Z}}_U < 0) \leq \frac{\sigma_n^2}{\mu_n^2} \tag{3.14}$$

Finally, the support of the denominator around the origin can be written as:

$$f_{\mathbf{h}_n^T \underline{\mathbf{Z}}_U}(0) \propto \exp(-\frac{\mu_n^2}{2\sigma^2}) \tag{3.15}$$

Hence, given a random variable whose mean $\mathbf{m_u}$ satisfies (3.11), the probability that the denominator random variable takes negative values (positive values) when $\mu_n > 0$ ($\mu_n < 0$) becomes negligible. Further, the probability that the denominator takes values close to zero becomes exponentially small with the distance of the mean from the Line at Infinity. Under such a setting, we can work under the condition that the denominator random variable is actually bounded away from zero. This allows for the existence of moments, and good approximations to the ratio density itself. In particular, it has been shown that a Normal distribution forms a good approximation to the ratio density [89,90] itself when the conditions of (3.11) is satisfied.

The fact that Normal densities can be approximated by Normal densities under the projective transformation (and assuming the requirements of (3.11) are satisfied) implies that the transformation of the random variable under a projective transformation can be modeled point-wise as an affine transformation, the parameters of which are dependent on the mean $\mathbf{m_u}$. The affinity does not extend to a transformation over a region because of the nature of the dependence of the

[2]The discussion is very similar for $\mu_n < 0$, the only difference being that we would then consider the probability that $\mathbf{h}_n^T \underline{\mathbf{Z}}_U$ takes positive values.

parameters on the mean $\mathbf{m_u}$. However, given the smoothness of such maps, in general a local affine approximation still remains valid over small neighborhoods, as long as the region that is imaged is sufficiently far away from the projection of the hyper-plane at Infinity. Such locally affine models have been used in existing literature [81], [120] for geometric grouping.

We now have a geometric setting in which we are guaranteed the existence of moments. These moments can be used for various minimum variance estimation techniques (such as the Kalman filter for dynamical systems). However, computation of these moments is not trivial given the non-linearity of the overall transformation. In section 3.4, we look at approximation techniques to compute these moments.

3.3.2 Role of theory in Observation Modeling

In section 3.2, we discussed two observations models and whether of not one forms a good approximation of the other. Using the theory described earlier, we prove that it is indeed the case.

For simplicity we use the scalar setting for this proof. The analysis extends easily for the multi-variate case. Consider the two observation models defined by the equations,

$$O1: \ y = H^{-1}(x) + n, n \sim N(0, \sigma^2) \tag{3.16}$$

$$O2: \ H(y) = x + \eta \tag{3.17}$$

where H is a projective transformation.

We are interested in listing the conditions when the observation model $O1$ reduces to $O2$. Starting from $O1$ we can apply the transformation H to get,

$$\begin{aligned} \tilde{O}1: \ H(y) &= H\left(H^{-1}(x) + n\right) \\ &= H(z), z \sim N(H^{-1}(x), \sigma^2) \end{aligned} \tag{3.18}$$

Given that H is a 1D projective transform and z a Normal distribution, the term $H(z)$ has a Ratio of Normals expression. However, as demonstrated earlier, if $H^{-1}(x)$, the mean of z satisfies the requirements of (3.11), then $H(z)$ *is well approximated by a Normal distribution*. This seems to indicate that the distribution of η in $O2$ should be Normal. However, to make that claim we need an additional result relating the mean of $H(z)$ to x. Note that we only know that $H(z)$ can be well approximated by a Normal distribution, say with mean μ and variance σ_x^2.

$$H(z) \approx \mu + \sigma_x z_1, z_1 \sim N(0,1) \tag{3.19}$$

where μ and σ_x depend heavily on x, σ^2 and the transformation H.

However, in general, μ is not equal to the mean projection of the image plane mean $H^{-1}(x)$ under the projection H. This is because of biases introduced by lack of symmetry of ratio distribution about its mode. However, when (3.11) is satisfied with an increasingly high value, then these asymmetries are negligible and, it can be shown easily μ is equal to the projection of the image plane mean $H^{-1}(x)$ through the projectivity H.

$$\mu = H(H^{-1}(x)) = x \tag{3.20}$$

Therefore, the observation model $O1$ reduces to $O2$ when the requirements of (3.11) are satisfied. We summarize this in the following lemma.

Lemma 3.3.1 *Consider the observation equation relating* $\mathbf{x} \in \mathscr{X} = \mathbb{P}^n$ *and* $\mathbf{y} \in \mathscr{Y} = \mathbb{P}^n$,

$$O1 : \mathbf{y} = H^{-1}(\mathbf{x}) + \mathbf{n}, \mathbf{n} \sim N(\mathbf{0}, \Sigma_0) \tag{3.21}$$

with H *being a projective transformation from* $\mathscr{Y}$ *to* $\mathscr{X}$. *Suppose the line at infinity of* $\mathscr{X}$ *under* H^{-1} *is imaged sufficiently far away from* $H^{-1}(\mathbf{x})$, *then the model* $O1$ *can be approximated with a high accuracy with a simpler equation given*

as

$$O2 : H(\mathbf{y}) = \mathbf{x} + \eta \tag{3.22}$$

where η is a multi-variate Normal random variable.

With this we have solved one of the two problems proposed in section 3.2. The only remaining issue is that of computing the covariance of the random variable η. As expected, the covariance matrix will depend not only on $\mathbf{x}$ but also on the transformation H itself. In the next section, we look at approximation techniques to compute these moments.

3.4 Approximation methods for Moment Computation

We look at approximations to the moment computation that can be made in the setting when the Cauchy component is of negligible strength. Our goal is to derive estimates for the mean and the covariance matrix of $\mathbf{Z_X}$. To begin with, we avoid the numerical techniques described in [89] due to their high computational costs. We look at first and second order approximation techniques to estimate these moments.

3.4.1 Affine Transformation

In the case of the affine transformation, $\mathbf{h_n} = (0, \cdots, 1)^T$, so that $\mathbf{h_n}^T \underline{\mathbf{Z}}_\mathbf{U}$ is no longer random, or alternatively $\mu_n = 1$ and $\sigma_n = 0$. This implies $b = \infty$, and the strength of the Cauchy component in the distributions of Z_x^i is zero. Further, $\mathbf{Z_x}$ is affine in $\mathbf{Z_u}$, making its components jointly Gaussian in distribution. Further, the distribution of $\mathbf{Z_X}$ is isotropic over $\mathscr{X}$ in the sense that its covariance is independent of its mean. One could approximate the projective transformation of the imaging with an affine model. However, such an approximation tends to be very coarse especially if the homography induces strong perspective projection. An affine camera maps points at infinity to points at infinity. In camera views,

where the line at infinity is projected close to the FOV, this would be a poor approximation.

3.4.2 Linear Approximation

We can approximate the transformation defined in (3.5) by linearizing it around the point $\mathbf{m_u} \in \mathscr{U}$ or equivalently, $\mathbf{m_x} \in \mathscr{X}$ (with $\mathbf{m_x}$ obtained by transforming $\mathbf{m_0}$ using the transformation H). This leads to a transformation of $\mathbf{Z_X}$ that is affine in $\mathbf{Z_U}$.

$$\mathbf{Z_X} \approx \mathbf{m_x} + J_H(\mathbf{m_u})(\mathbf{Z_U} - \mathbf{m_u}) \tag{3.23}$$

where

$$J_H(\mathbf{m_u}) = \left(\frac{1}{\mathbf{h_n}^T \underline{\mathbf{m}}_\mathbf{u}} \begin{bmatrix} \mathbf{h_0}^T \\ \vdots \\ \mathbf{h_{n-1}}^T \end{bmatrix} - \frac{\mathbf{m_x}\mathbf{h_n}^T}{\mathbf{h_n}^T \underline{\mathbf{m}}_\mathbf{u}} \right) \begin{bmatrix} \mathbb{I}_n \\ \mathbf{0}_n^T \end{bmatrix} \tag{3.24}$$

where $\mathbb{I}_n$ is rank n Identity matrix and $\mathbf{0}_n$ is a column of zeroes of length n.

We can now identify $\hat{\mu}_\mathbf{x} = \mathbf{m_x}$ to be an estimate of the mean of $\mathbf{Z_X}$ and $\hat{\Sigma}_X = J_H(\mathbf{m_u}) S_u J_H(\mathbf{m_u})^T$.

Corollary 3.4.1 *As the mean $\mathbf{m_u} \in \mathscr{U}$ approaches the inverse map of the hyperplane at infinity, the (approximated) variance of the transformed random variable $\mathbf{Z_X}$ goes to infinity.*

Proof: The term $\mathbf{h_n}^T \underline{\mathbf{m}}_\mathbf{u} \propto l_\infty^T \underline{\mathbf{m}}_\mathbf{u}$ appears in the denominator of the expression for $J_H(\mathbf{m_u})$. □

Linearization is a popular tool for covariance propagation given its analytic tractability. It has been used for matching, mosaicing [98], metrology [37] and tracking [47].

3.4.3 Unscented Transformation

An alternate and better approximation is to compute the means and covariances of $\mathbf{Z_X}$ using the unscented transformation [69]. The unscented transform is used to propagate the means and covariances (in general, moments) through a deterministic choice of points called sigma points and their associated weights.

The moments of the transformed random variable are computed as follows. First, we generate the sigma points $\nu_{\mathbf{i}}, i = 0, \ldots, 2n_u$ and the associated weights w_i, where n_u is the dimensionality of $\mathbf{Z_U}$, which in our setting is $n_u = n$. The sigma points are generated using the following selection scheme [131].

$$\begin{aligned} \nu_{\mathbf{0}} &= \mathbf{m_u} \\ \nu_{\mathbf{i}} &= \mathbf{m_u} + \left(\sqrt{(\kappa + n_u)S_u}\right)_i & i = 1, \ldots, n_u \\ \nu_{\mathbf{n_u+i}} &= \mathbf{m_u} - \left(\sqrt{(\kappa + n_u)S_0}\right)_i & i = 1, \ldots, n_u \end{aligned} \tag{3.25}$$

where κ is a constant scalar and $\left(\sqrt{(\kappa + n_u)S_0}\right)_i$ is the i-th row of the matrix square root of $(n_u + \kappa)S_0$. The weights take values as follows:

$$w_i = \begin{cases} \kappa/(\kappa + n_u) & i = 0 \\ 1/(2(\kappa + n_u)) & i \neq 0 \end{cases} \tag{3.26}$$

Each sigma point is propagated using the transformation in (3.4).

$$\chi_{\mathbf{i}} = \frac{1}{{\mathbf{h_n}}^T \underline{\nu}_{\mathbf{i}}} \begin{pmatrix} {\mathbf{h_0}}^T \underline{\nu}_{\mathbf{i}} \\ \vdots \\ {\mathbf{h_{n-1}}}^T \underline{\nu}_{\mathbf{i}} \end{pmatrix}, i = 0, \ldots, 2n_u \tag{3.27}$$

The estimates for the mean $\mu_{\mathbf{x}}$ and covariance matrix Σ_x of $\mathbf{Z_X}$ are given as,

$$\begin{aligned} \hat{\mu}_{\mathbf{x}} &= \textstyle\sum_{i=0}^{2n_u+1} w_i \chi_{\mathbf{i}} \\ \hat{\Sigma}_x &= \textstyle\sum_{i=0}^{2n_u+1} w_i (\chi_{\mathbf{i}} - \hat{\mu}_{\mathbf{0}})(\chi_{\mathbf{i}} - \hat{\mu}_{\mathbf{0}})^T \end{aligned} \tag{3.28}$$

The choice of the value of κ is important. A list of possible choices and the properties induced by them is given in [131]. The estimates of the mean and covariance matrix are accurate upto second order [131], more accurate than first order linearization.

We now have three ways to estimate the mean and the covariance matrix of $\mathbf{Z_X}$ as a function of the mean of the $\mathbf{Z_U}$, $\mathbf{m_u}$ and the transformation matrix H. Of the three, the affine approximation (3.4.1) is probably the least accurate except when the projection is truly affine. The unscented transformation (3.4.3) gives the most accurate estimate of the covariance matrix. The linearizing approach gives an analytical expression for the covariance matrix which is desirable for certain applications. *Finally, these moments exists only when the requirement of (3.11) is satisfied.* It is important to verify this before computing the approximations using the methods presented in this section.

In the next chapter, we discuss applications of the fusion theory developed here and discuss practical implications in the design of camera networks.

3.5 Extensions to Non-Gaussian Image plane distributions

In vision applications, especially tracking, it is common to encounter multi-modal distributions. In this sense the assumption of Gaussian distribution on $\mathscr{U}$ is limiting. There exist many representation schemes that capture multi-modal distributions. The popular ones include kernel based representation (such as Parzen windows) and particle sets. We start off with the simplest extension from a Gaussian, namely a mixture of Gaussian (MoG) assumptions for the image plane densities.

Assuming $\mathbf{Z_U}$ has a MoG distribution, it can be immediately seen that $\mathbf{h_i}^T\underline{\mathbf{Z_U}}$ has a univariate MoG distribution. So, the distributions of Z_x^i (as defined in (3.5)) is a *Ratio of Mixtures of Gaussian.*

3.5.1 Ratios of MoG = Mixture of Ratio of Gaussians

We show that *ratios of mixture of Gaussians* leads to a mixture comprising of ratio of Gaussians. Let $\mathbf{Z_U}$ be a bivariate random variable whose distribution is that of a mixture of Gaussians.

$$\mathbf{Z_U} \backsim \sum_{k=1}^{M} m_k \mathrm{N}(\mathbf{m_u}^k, S_u^k) \tag{3.29}$$

Alternatively, we can *generate* the distribution by a two-step process involving a multi-nominal distribution F and a set of Gaussian random variables.

$$\mathbf{Z_U} \backsim \sum_{k=1}^{M} I(F=k)\mathbf{X_k} \tag{3.30}$$

where $\mathbf{X_k} \backsim \mathrm{N}(\mathbf{m_u}^k, S_u^k)$ are Gaussian distributed, and F is multi-nominal taking values in the set $\{1, \ldots, M\}$, such that $\Pr(F = k) = m_k$. $I(\cdot)$ serves as an indicator function.

Now, consider the transformation to obtain Z_x^i.

$$Z_x^i = \frac{\mathbf{h_i}^T \underline{\mathbf{Z_U}}}{\mathbf{h_n}^T \underline{\mathbf{Z_U}}} = \frac{\sum_{k=1}^{M} I(F=k)\mathbf{h_i}^T \underline{\mathbf{X_k}}}{\sum_{k=1}^{M} I(F=k)\mathbf{h_n}^T \underline{\mathbf{X_k}}} \tag{3.31}$$

Using the total probability theorem,

$$f_{Z_x^i}(z) = \sum_{j=1}^{M} f_{Z_x^i}(z|F=j)\Pr(F=j) \tag{3.32}$$

$$Z_x^i|(F=j) = \frac{\mathbf{h_i}^T \underline{\mathbf{X_j}}}{\mathbf{h_n}^T \underline{\mathbf{X_j}}} \tag{3.33}$$

Hence, we can now write the expression for Z_x^i as

$$Z_x^i = \sum_{j=1}^{M} \frac{\mathbf{h_i}^T \underline{\mathbf{X_j}}}{\mathbf{h_n}^T \underline{\mathbf{X_j}}} I(F = k) \tag{3.34}$$

Noting that the ratio $\mathbf{h_i}^T \underline{\mathbf{X_j}} / \mathbf{h_n}^T \underline{\mathbf{X_j}}$ has a *Ratio of Gaussian* distribution, the distribution of Z_x^i is a mixture of such ratios of Gaussians. *The number of mixture components do not change, and neither does the mixture weights.* However, each mixture component has a much more complicated distribution.

We can now extend the theory developed earlier to this case.

Lemma 3.5.1 *Let* $\mathbf{Z_U}$ *be distributed as a mixture of multi-variate Gaussian as defined in (3.29), then* $\mathbf{Z_X}$ *has a Mixture of Ratio of Gaussians distribution. Further, as before, the moments of* Z_x^i *are well-defined when all the mixture means* $\mathbf{m_u^i}, i = 1, \ldots, M$ *are sufficiently far away from the inverse projection of the hyperplane at Infinity of* $\mathscr{X}$*.*

Each component can now be approximated with a Gaussian provide its parameters satisfy the requirements of (3.11), implying that the overall density is well approximated by a mixture of Gaussian assumption. Hence, to this effect under the assumptions, *a mixture of Gaussians gets projected to a mixture of Gaussians, retaining the same number of mixture components and identical mixture weights.* However, though the form of the density remains the same, each mixture component on $\mathscr{U}$ gets transformed by a different affine transformation to form the mixture component for the density on $\mathscr{X}$. Hence, the overall transformation of the random variable is not affine as in the earlier case.

Chapter 4

Fusion under the Homography Constraint

4.1 Multi-View Tracking

The most common example of an invertible projective transformation occurs when the scene is planar. In this scenario, the mapping between the image plane and the world plane (assuming a local coordinate reference on each plane) is defined by a two-dimensional projective transformation (also called a homography). In this section, we use the theory developed in Section 3.3 for location estimation and tracking using both single and multiple cameras.

Consider a single camera, whose image plane to ground plane is defined by a 3×3 matrix H. Given a Gaussian (or mixture of Gaussian) distribution on the image plane, we can use the approximations listed in 3.4.3 to get accurate moments of the transformed random variable on the plane, *provided the mean on the image plane is far away from the horizon line of the ground plane*. Further, it is expected that the variance on the ground plane changes as the mean on the image plane changes, specifically, it increases as the mean approaches the line at infinity.

As an extension, when multiple cameras observe a planar scene the individual image plane to world plane transformations are different. Now depending on the camera locations, the same object on the world plane can possibly be imaged at varying resolution on the individual image planes.

As an example, consider a point object being imaged onto multiple views. Due to imaging and modeling/estimation inaccuracies, the locations of points on the individual image planes are not accurately known. Even under the assumption

that the individual image plane error variances are identical, the variance on the ground plane from the individual views can be very different. And finally, such variations depend on the actual location of the imaged point. Figure 3.2 explains this concept graphically. Suppose, we are interested in an algorithm to fuse the individual estimates. It is immediately clear that estimates such as a sample mean will not be *efficient*, as the individual estimates are not identically distributed. Further, it is possible that one or more of the estimate might lie near the Line at infinity and might have ill-defined moments (either in terms of high variance or in the existence of the moments itself). For these reasons, the model is useful for fusing multi-view estimates.

Finally, we can use Lemma 3.3.1 to transform noisy image plane observations to the ground plane, and use the projected observations as complete observers of the state (location) of the target. Further, when multiple cameras are observing the same target, each camera provides an observation whose noise properties are different. The moments of the corrupting noise for each view can be obtained by applying the unscented transformation.

4.1.1 Multiple cameras

Given M cameras, and the homography matrices $H_i, i = 1, \ldots, M$ between the camera views and the ground plane, we provide an algorithm for fusing location estimates. Let $\mathbf{Z}_U^i$ be the random variable modeling the target location on the image plane of the i-th camera. We first assume that the random variables $\{\mathbf{Z}_U^i\}_{i=1}^M$ are statistically independent. This assumption is justified for imaging (sensor) noise. However, there are instances such as occlusion, parallax when the noise is due to an error in the modeling. In such cases, the noises/errors are correlated across cameras. However, estimation of this correlation is complicated.

The distribution of $\mathbf{Z}_U^i$ comes from the output of a tracking algorithm or a detection algorithm. However, we are only interested in the mean $\mathbf{m}_u^i$ and the covariances S_u^i of the distribution. If the underlying tracker is indeed a Kalman or

a particle filter, then we can readily obtain estimates of mean and the covariances as an output. In cases, when this is not possible (such as the KLT), we assume the mean to be the observation tracker output itself and assume suitable values for the covariance matrices.

We use the unscented transformation described in section 3.4.3, to get estimates of mean $\hat{\mu}_x^i$ and covariance matrix $\hat{\Sigma}_x^i$ of $\mathbf{Z}_X^i$ the random variable modeling location on the plane as estimated from the i-th camera. A minimum variance estimator for the location on the ground plane can now be formulated. Further, Lemme 3.3.1 ensures that the estimates $\hat{\mu}_x^i$ are unbiased (or $\mathrm{E}(\hat{\mu}_{\mathbf{x}}^i) = \mu_{\mathbf{x}}$ the true target location). The minimum variance estimate $\hat{\mu}_{\mathbf{x}} = (\hat{\mu}_x, \hat{\mu}_y)^T$ is computed as [127] [126],

$$\hat{\mu}_{\mathbf{x}} = \sum_{i=1}^{M} (\hat{\Sigma}_x^i)^{-1} \Sigma_{mv} \hat{\mu}_x^i, \Sigma_{mv} = \left(\sum_{j=1}^{M} (\hat{\Sigma}_x^j)^{-1} \right)^{-1} \tag{4.1}$$

It can be shown that among the class of linear estimators, the one defined in (4.1) is *optimal* in the sense of minimum variance. Finally, the covariance matrix of the minimum variance estimate $\hat{\mu}_{\mathbf{x}}$ can be computed from (4.1).

$$\mathrm{covar}(\hat{\mu}_{\mathbf{x}}) = \Sigma_{mv} = \left(\sum_{j=1}^{M} \left(\hat{\Sigma}_x^j \right)^{-1} \right)^{-1} \tag{4.2}$$

4.1.2 Dynamical system for tracking

In most cases, we are interested in tracking the location with time (and not just a static estimation). We formulate a discrete time dynamical system for location tracking on the plane. The state space comprises of the location and velocity on the ground plane. Let $\mathbf{x}_t$ be the state space at time t, $\mathbf{x}_t = [x_t, y_t, \dot{x}_t, \dot{y}_t]^T \in \mathbb{R}^4$.

The state evolution equations are defined using a constant velocity model.

$$\mathbf{x}_t = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{t-1} + \omega_t \tag{4.3}$$

where ω_t is a noise process. The observation model uses the mean and covariance models derived in section 3.3. The observation vector $\mathbf{y}_t \in \mathbb{R}^{2M}$ is just the stack of location means estimated from each camera using the unscented transformation. The observation model is given as,

$$\mathbf{y}_t = \begin{bmatrix} \hat{\mu}_{\mathbf{1}} \\ \vdots \\ \hat{\mu}_{\mathbf{M}} \end{bmatrix}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & & & \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x}_t + \Lambda(\mathbf{x_t})\Omega_t \tag{4.4}$$

where Ω_t is a zero mean noise process with an identity covariance matrix. $\Lambda(\mathbf{x_t})$ sets the covariance matrix of the overall noise, and is defined as,

$$\Lambda(\mathbf{x_t}) = \begin{bmatrix} \Sigma_x^1(\mathbf{x_t}) & \cdots & 0_{2\text{X}2} \\ \vdots & \ddots & \vdots \\ 0_{2\text{X}2} & \cdots & \hat{\Sigma}_x^M(\mathbf{x_t}) \end{bmatrix}^{\frac{1}{2}} \tag{4.5}$$

where $0_{2\text{X}2}$ is a 2x2 matrix with zero for all entries, and $\Sigma_x^i(\mathbf{x_t})$ is the covariance matrix of $\mathbf{Z}_X^i$ when the true location of the target on the ground plane is $\mathbf{x_t}$.

The observation model in (4.4) is a multi-view extension of the complete observer model proposed in section 3.2 and validated in 3.3. There are two important things that this model captures.

- The noise properties of the observations from the different view are different,

and the covariances depend not only on the view, but also on the true location of the target $\mathbf{x}_t$. This dependence is encoded in Λ.

- The MLE of $\mathbf{x}_t$ (i.e the value of $\mathbf{x}_t$ that maximizes the probability $p(\mathbf{y}_t|\mathbf{x}_t)$) is the minimum variance estimator described in the previous section.

Tracking of target(s) can now be performed using a Kalman or particle filter.

4.1.3 Results: Variance Maps for Static Estimation

Using homography matrices computed for each view in a camera network, we can compute the covariance matrix over regions of interest on the plane, using the unscented transformation at each camera. Figure 4.1 shows the results over a four camera network. The variances are plotted for the corresponding point in the top-down image representing the true target location. It is seen that the variances increase as one approaches the line at infinity for the corresponding view. The variance of the minimum variance estimator (4.2) can also be computed as a function of the true location of the target on the plane. Such plots are potentially of great use in camera placement problems, where given a set of cameras and a region of interest on a plane, we are interested in the placement of additional cameras that improve the performance of tracking of targets.

Figure 4.2 shows similar results for a three camera setup with a highly asymmetrical camera setup. The camera corresponding to the right-most column was used ONLY to provide ground truth for tracking experiments of section 4.1.4. Two of the cameras are placed such that they can estimate the location of the target at high accuracy in only one direction. By simultaneously combining the estimates from both cameras and using the covariance maps it is possible to get estimates that have low variances in both directions (last row of Figure 4.2.) This extreme case of camera deployment is used next to show the need for the explicit modeling of camera and plane geometry.

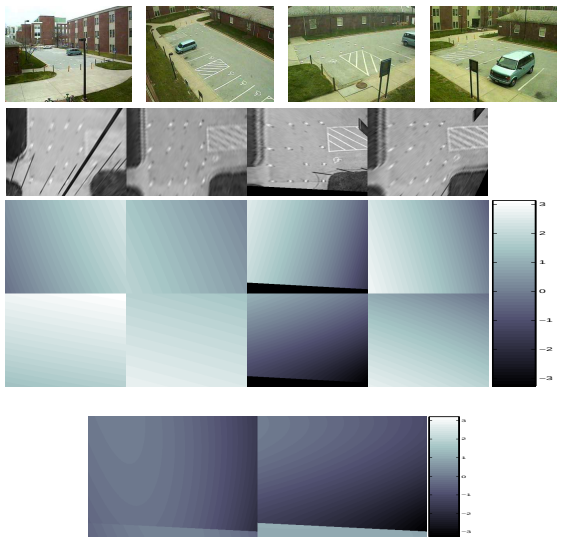


Figure 4.1: Variance estimates of the camera setup shown in the top row. (top row) Camera views (second row) Top view of the plane generated from the corresponding view from the first row (third row) Variance estimate of Z_x and Z_y for each camera in log-scale over the ground plane. (last row) Variance (in log scale) of the minimum variance location estimator along the two axes.

4.1.4 Results: Multi-camera Tracking

Three Camera Setup We now present an experiment for illustrating the need of such models in multi-camera tracking, especially when the camera views are highly asymmetrical. (see Figure 4.2). The region of interest is the chessboard and a laser pointer is used to create a target. A color based segmentation is used to detect the target created by the laser pointer. *The camera corresponding to the right column is used just for providing the ground truth.*

We compare the performance of two tracking systems: one whose observation model incorporates view-dependent error characterization (see section 4.1.2) and another which treats all cameras identically and uses an *isotropic* model across cameras. *A Kalman filter is used to track both systems.* The parameters characterizing the state transition model were learnt independently using data from a different camera. The tracking results are shown in Figure 4.3. For quantitative

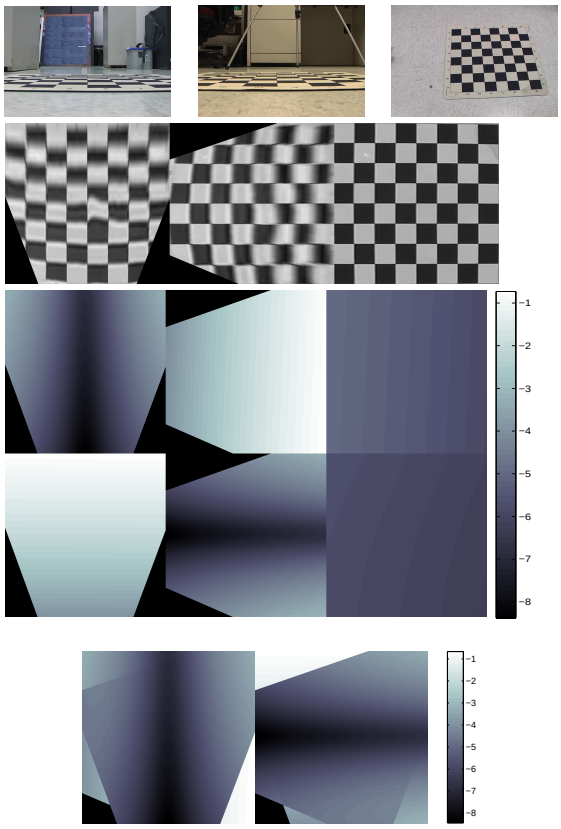


Figure 4.2: Three cameras imaging a chessboard, with two cameras placed very close to the ground, along the two orthogonal axes of the chessboard. (top row) Views from each camera. (second row) top views constructed from the homography estimated at each camera. (third-fourth rows) Variance estimate of Z_x and Z_y respectively in log-scale over the ground plane. Note that each camera can estimate location in only one direction accurately. (last row) Variance of the minimum variance estimator (in log scale) using *only the cameras corresponding to the left and middle columns.*

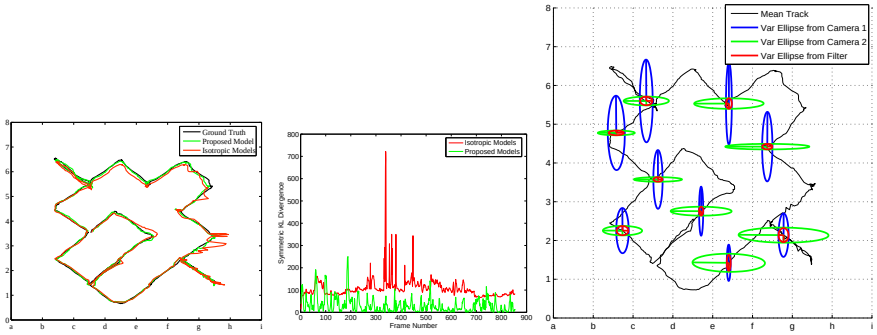


Figure 4.3: Comparison of tracking with two systems, one that uses the models proposed and the other that treats the two cameras identically. (left) plots of the filter outputs (center) Comparison of the outputs with the ground truth using symmetric KL-divergence. (right) Variance ellipses for location estimates from the two cameras and the filter using the proposed model.

evaluation, we compute the symmetric KL-divergence between the output of the two systems and the ground truth. Visually, the tracking results obtained by using the proposed models are extremely close to the ground truth and very smooth. The variance ellipses at various locations on the track demonstrate the efficacy of the modeling. While each of the cameras has a large uncertainty along one axis, the final estimate has low uncertainty along both axes.

Four Camera Setup: A multi-target tracking system was developed to test the efficacy of the proposed models over the realistic camera placement of Figure 4.1. The bottom-most point from each background subtracted blob at each camera is extracted and projected onto the world plane. Association of this data to trackers is performed using the classical Joint Probability Data Association Filter (JPDAF) [5] with data from each camera associated separately. Ground truth was obtained using markers. As before, two observation models are compared: one employing the proposed approach and the other that assumes isotropic modeling across views. Finally, a particle filter is used to track, the choice motivated given missing data points due to occlusion. Testing was performed over a video of 8000 frames with three targets introduced sequentially at frames 1000, 4300 and

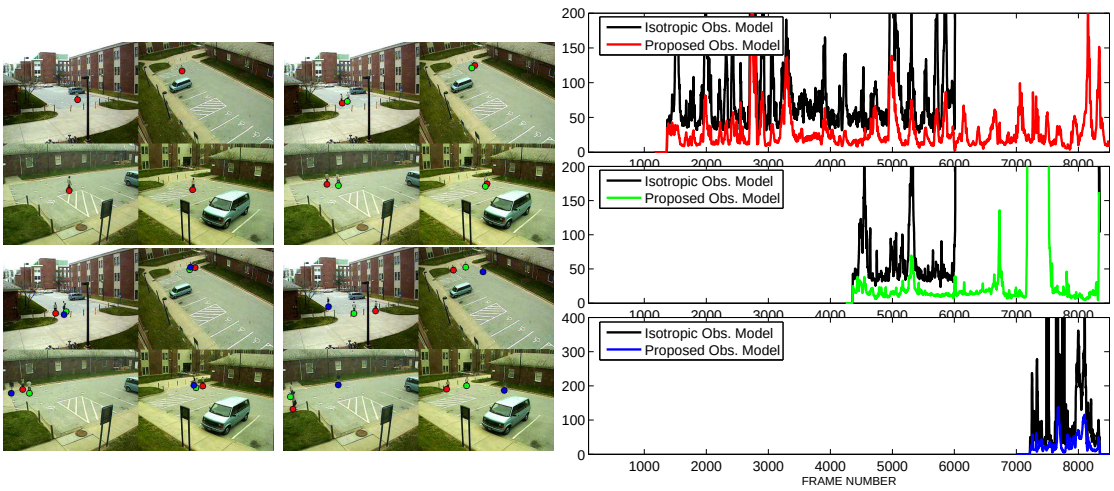


Figure 4.4: Tracking results for three targets over the 4 camera dataset. (best viewed in color/at high Zoom) (left) Snapshots of tracking output at various timestamps. (right) Evaluation of tracking using Symmetric KL divergence from ground truth. Two systems are compared: one using the proposed observation model and the other using isotropic models across cameras. Each plot corresponds to a different target. The trackers using isotropic models swap identities around frame 6000. The corresponding KL-divergence values go off scale.

7200. Figure 4.4 shows tracking results for this experiment. The proposed model consistently results in lower KL divergence to the ground truth.

Six Camera Setup In a similar experiment, we tested the algorithm on a 6 camera network. Nine targets were allowed to freely move around the region of interest. Figure 4.5 summarizes the result from this experiment.

In [117], we demonstrate a system for visualization of multi-camera data using the tracking algorithm developed here. As the front-end of the overall system, the multi-camera tracker detects and estimates trajectories of moving humans. Sequences of silhouettes extracted from each human are matched against models of known activities. Information of the estimated trajectories and the recognized activities at each time instant are then presented to a rendering engine that animates a set of virtual actors synthesizing the events of the scene. In this way, the visualization system allows for seamless integration of all the information inferred from the sensed data. Such an approach places the end-user in the scene, providing tools to observe the scene in an intuitive way, capturing geometric as well as spatio-temporal contextual information. Such ideas can potentially be applied to modeling and analysis of activities involving multiple humans exhibiting coordinated group behaviors such as in football games and training drills for security enforcement. Figure 4.6 shows snapshots of the visualization of the results in Figure 4.5.

4.2 Metrology

The most common way to measure lengths involve the use of cross-ratios and the vanishing lines of a plane along with the vertical vanishing point [37]. A graphical illustration of this is shown in Figure 4.7. Using the invariance of the cross-ratio between the points V_p, O_b, O_t and R'_t.

$$\frac{h_o}{h_r} = \frac{|O_t O_b|}{|V_p O_t|} \frac{|V_p R'_t|}{|R'_t O_b|} \tag{4.6}$$

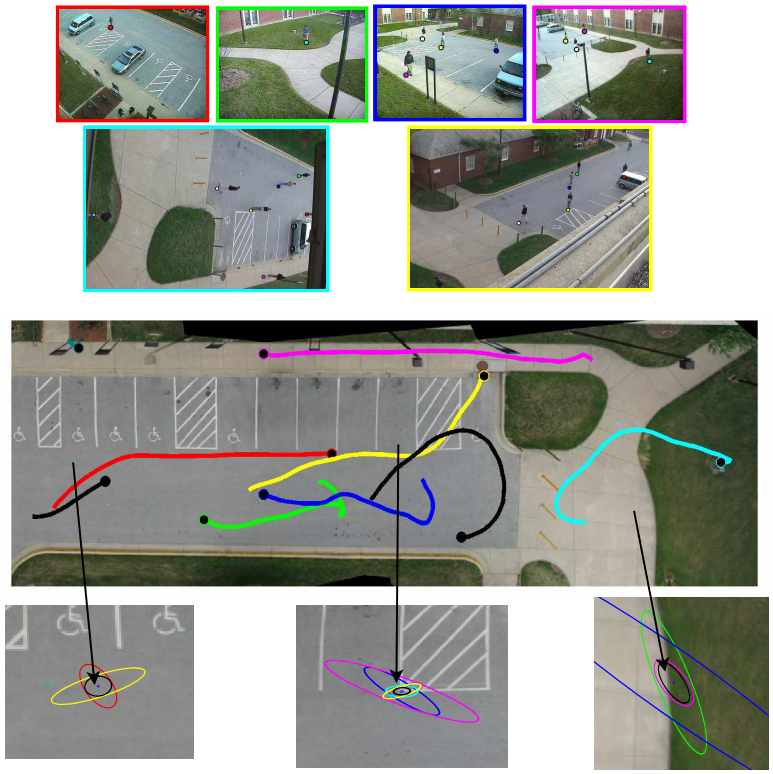
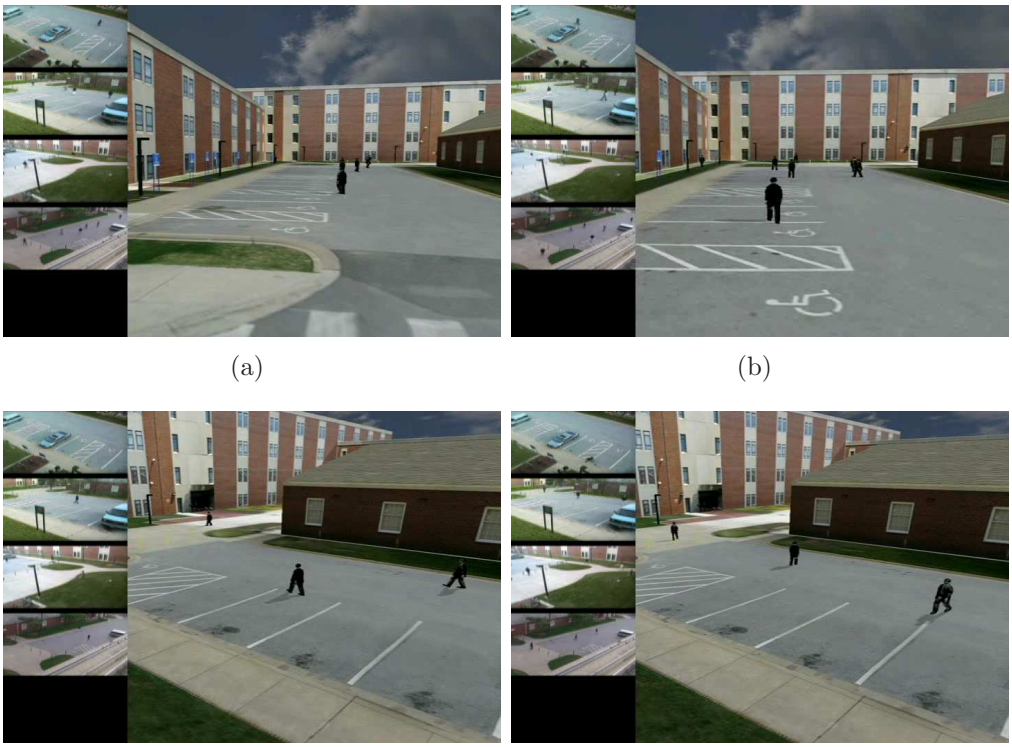


Figure 4.5: Tracking results on a six camera setup. (Top two rows) Snapshots from the camera views. Each view is color coded (boundary), used in describing the variance ellipses in the last row. (Third row) Top view of the sensed area with trajectories of targets tracked. (Last row) Variance ellipses at different locations on the plane. The color code associates the ellipse with the camera view from which the uncertainty is generated. The black ellipse corresponds to the minimum variance that can be obtained at the specific location.



(a) (b)

(c) (d)

Figure 4.6: Visualization of the multi-target tracking results of Figure 4.5. The left panel with each image shows snapshots from four of the six cameras. The proposed system allows for arbitrary placement of the virtual camera.

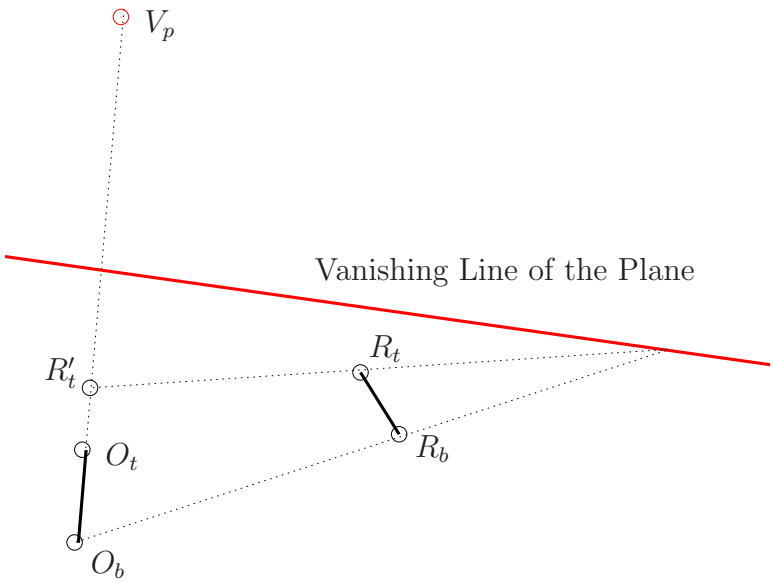


Figure 4.7: Using a reference object R_bR_t with known length h_r to measure the length of O_bO_t. Both objects are assumed to be oriented vertical to the plane. Knowledge of the vanishing line and the vertical vanishing point V_p is required.

where h_r is the true height of the reference object. The invariance of the cross-ratio ties in neatly with a fundamental property of 1D projective transformations or 1D homographies.

In the case of metrology, the length of the reference object and the vanishing point allow for the definition of the Euclidean reference. Identifying the line observed on the image plane as $\mathscr{U} = \mathbb{P}^1$ and choosing O_b as the origin, the points R'_t and V_p take the following values.

$$\underline{O_b} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \underline{R'}_t = \begin{pmatrix} |O_bR'_t| \\ 1 \end{pmatrix}, \underline{V_p} = \begin{pmatrix} s|O_bV_p| \\ 1 \end{pmatrix}, s = \pm 1 \qquad (4.7)$$

The value of s in the definition of V_p in (4.7) is chosen as $+1$ if V_p lies on the same side of O_b as O_t, and -1 otherwise.

In the Euclidean reference $\mathscr{X} = \mathbb{P}^1$ these points take the values (in homoge-

neous coordinates),

$$\underline{O}_b^x = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \underline{R'}_t^x = \begin{pmatrix} h_r \\ 1 \end{pmatrix}, \underline{V}_p^x = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{4.8}$$

Using this correspondence information, we can compute the 1D homography H_M between $\mathscr{U}$ and $\mathscr{X}$.

$$H_M = \begin{bmatrix} h_r(|O_bR'_t| - s|O_bV_p|) & 0 \\ |O_bR'_t| & -s|O_bR'_t||O_bV_p| \end{bmatrix} \tag{4.9}$$

Now, we can transform O_t using the homography H_M to obtain its length.

$$\underline{O}_t^x = \begin{pmatrix} h_o \\ 1 \end{pmatrix} \sim H \begin{pmatrix} |O_bO_t| \\ 1 \end{pmatrix} \tag{4.10}$$
$$h_o = \frac{h_r(|O_bR'_t| - s|O_bV_p|)|O_bO_t|}{|O_bR'_t|(|O_bO_t| - s|O_bV_p|)}$$

It can be easily verified that the expression for h_o in (4.10) is identical to that in (4.6).

Finally, it is noted that the homography defined in (4.9) varies as the point O_b corresponding to the point on the plane is changed. Given a static camera, the vanishing points are fixed. The reference object also forms a part of the calibration information. Hence, given a location on the image plane $\mathbf{U}$ and the necessary calibration information we can define the 1D homography $H_M(\mathbf{U})$ that defines the projective transformation mapping points on the line $\mathbf{U}$ and V_p to points on an Euclidean line where $\mathbf{U}$ maps to the origin, V_p to the point at infinity. The proximity of the vertical vanishing point to the point $\mathbf{U}$ (distance measured on the image plane) plays a pivotal role in accurate estimation of object heights.

4.2.1 Metrology under noisy observations

We now link the theory developed in Section 3.3 for estimation of lengths using cross ratios under noisy observations. We assume that all calibration information, such as the vanishing line, the vertical vanishing point and the reference object are available noise free.

Given a background subtracted blob of a human, we compute a line passing through the vertical vanishing point that best fits the blob (in the sense of minimum error). Using this line we locate O_b and O_t as the top and bottom points of the blob (the notion of *top* versus *bottom* is assumed to be a part of calibration, and is encoded in the sign of s in (4.7)). The noisy estimation of the background subtracted image and the deviations of an actual human silhouette from a *stick model* leads to errors in the location of O_b and O_t. Estimating the height of the person from the noisy data is of interest.

Equation (4.10) links this estimation problem with that of location estimation along a line under a 1D projective transformation. This ties the metrology problem with the estimation methods described in earlier sections. Specifically, the location of the vertical vanishing point (which is the inverse map of the point at infinity on $\mathscr{X}$) becomes of immense importance for accurate estimation. It is expected that views in which the vanishing point is close to O_t and O_b, such as a top-down view of the plan, would lead to estimates of inferior accuracy as compared to those in which the vanishing point is imaged at infinity, which happens if the principal ray of the camera is perpendicular to the plane normal.

However, we also note that when O_b, the bottom point is corrupted by noise it becomes necessary for estimating the correct location of the point along with the height of the person. This corresponds to a coupling of location and height estimation. In this context, it is preferable to simultaneously estimate the true location of the object along with the height of the person.

As in the location estimation on the plane, the accuracy of the metrology also

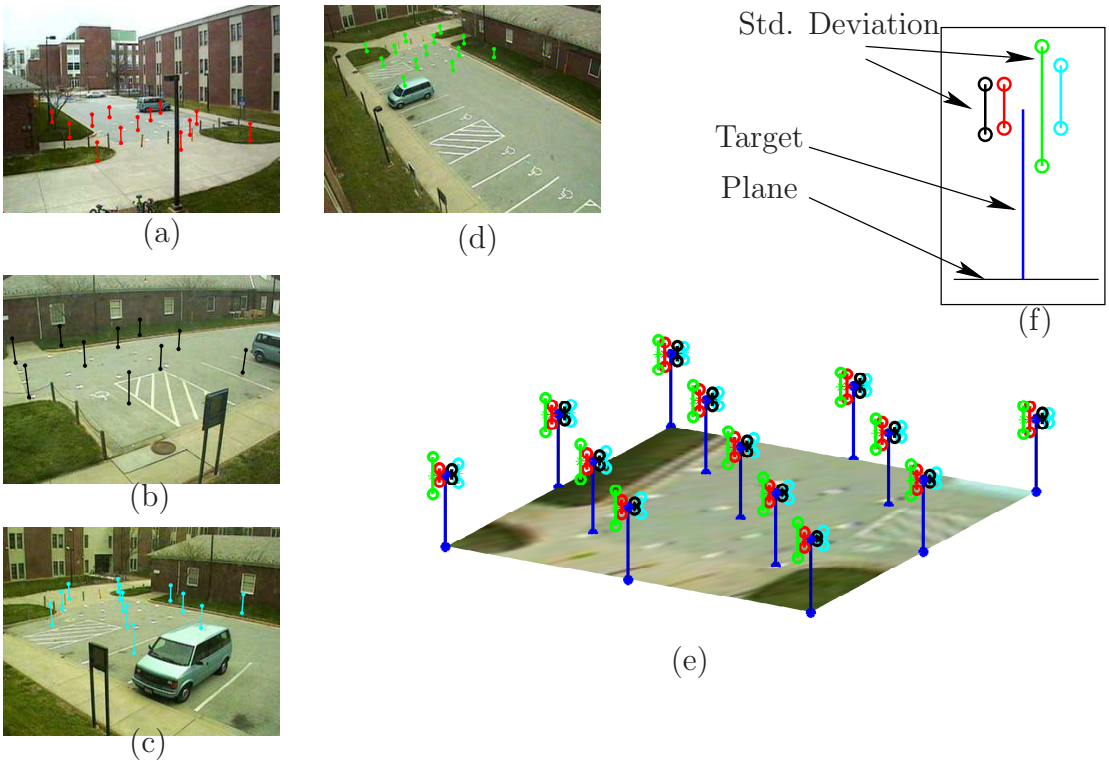


Figure 4.8: Multi-camera Metrology. (a)-(d) Color coded camera views showing possible location of a target (e) Standard deviation of height estimate from each camera view assuming a 2-pixel standard deviation error on the image plane. The color of the standard deviation bars represent the views in (a)-(d) (f) Legend. The point on the plane is assumed to be error free and an error standard deviation of $\sqrt{5}$ pixels for the top point.

depends on the location of the camera with respect to the object being imaged. When we have multiple estimates arising from different views, the estimates are bound to be of different distributions. As before the fusion scheme should account for the differences and utilize them appropriately. Figure 4.8 illustrates that variances of height estimates coming from different views can be very different. As before, such modeling is useful in designing efficient estimators. Note that these estimates are made under no noise assumption on the bottom point O_b and a standard deviation of $\sqrt{5}$ pixels on the top point O_t (on the image plane).

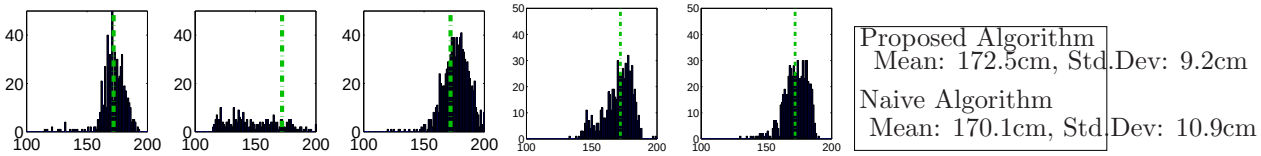


Figure 4.9: Histograms of height estimates from a three camera setup over a minute long video. Each frame of the synchronized video was used to obtain height estimates. (Cols 1 - 3) estimates from the individual cameras, (col 4) estimates after a naive fusion algorithm, (col 5) estimates from the proposed fusion method and (col 7) numerical summary of results. The height of the person under surveillance was 172.5 cm (to the nearest inch).

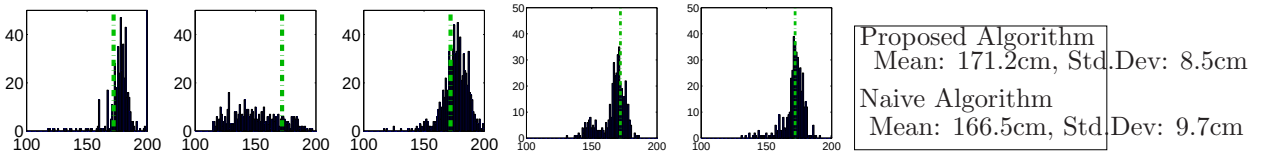


Figure 4.10: Histograms of height estimates from a three camera setup over a minute long video. Each frame of the synchronized video was used to obtain height estimates. (Cols 1 - 3) estimates from the individual cameras, (col 4) estimates after a naive fusion algorithm, (col 5) estimates from the proposed fusion method and (col 7) numerical summary of results. The height of the person under surveillance was 170 cm (to the nearest inch).

4.2.2 Metrology: Results

Figures 4.9 and 4.10 show results from a three camera sequence of about a minute long. We are able to obtain the height estimate for each frame at each camera using (4.6). Further, the estimates for each frame were fused using the proposed algorithm. Histograms of such estimates are displayed in Figures 4.9 and 4.10. The fusion algorithm shows improvement over both the individual camera estimates and a naive fusion strategy that does not explicitly account for the projective transformations. However, this improvement is only marginal, possibly due to the correlated nature of noise across frames in a video.

Chapter 5

Joint Acoustic Video Tracking

The epipolar constraint is the most general constraint governing imaging with two projective cameras. As mentioned in Chapter 2, tracking with epipolar constraint can be formalulated with state spaces involving common overlapping states. In this chapter, we explore the use of such state spaces for tracking.

Recently, hybrid nodes that contain an acoustic array collocated with a camera were proposed for vehicle tracking problems [31]. To intelligently fuse information coming from both modalities, novel strategies for detection and data association have to be developed to exploit the multi modal information. Moreover, the fused tracking system should be able to sequentially update the joint state vector that consists of multiple target motion parameters and relevant features (e.g., shape, color and so on), which is usually only partially observable by each modality.

It is well known that acoustic and video measurements are complementary modalities for object tracking. Individually, the acoustic sensors can detect targets [68, 134, 70], regardless of the bearing with low power consumption, and the video sensors can provide reliable high-resolution localization estimates [60], regardless of the target range, with high power consumption. Hence, by fusing the acoustic and video modalities, we (i) achieve tracking robustness at low acoustic signal-to-noise ratios (SNR) or during video occlusion, (ii) improve target counting/confirmation, and (iii) design algorithms that permit a power vs. performance trade-off for hybrid node management.

In the literature [132,30,51], one finds that fusion of acoustic and video modalities has been applied to problems such as tracking of humans under surveillance and smart videoconferencing. Typically, the sensors are a video camera and an

acoustic array (not necessarily collocated). In [132], the acoustic time delay-of-arrivals (TDOA's), derived from the peaks of the generalized cross-correlation function, are used along with active contours to achieve robust speaker tracking with fast lock recovery. In [30], jump Markov models are used for tracking humans using audio-visual cues, based on foreground detection, image-differencing, spatiospectral covariance matrices, and training data. The work by Gatica-Perez *et al.* [51] demonstrates that particle filters, whose proposal function uses audio cues, have better speaker tracking performance under visual occlusions.

Much of the work in acoustic video fusion for videoconferencing do not extend to the outdoor vehicle tracking problem. In particular, the issue of audio-video synchronization must be modeled to account for acoustic propagation delays. In vehicle tracking problems, the average target ranges of 100-600m result in acoustic propagation delays in the range of 0.3-2s. Acoustics and video asynchronization causes biased localization estimates that can lead to filter divergence. This is because the bias in the fused cost function increases the video's susceptibility to drift in the background. In addition, motion models should adaptively account for any rapid target motion. Moreover, the visual appearance models should be calculated online as opposed to using trained models for tracking. Although fixed image templates (e.g., wire-frames in [65,132,51]) are very useful for face tracking, they are not effective for tracking vehicles in outdoor environments. Adaptive appearance models are necessary for achieving robustness [82, 141, 66].

To track vehicles using acoustic and video measurements, we propose a particle filtering solution that can handle multiple sensor modalities. We use a fully joint tracker, which combines the video particle filter tracker [141] and a modified implementation of the acoustic particle filter tracker [25] at the state space level. We emphasize that combining the output of two particle filters is different from formulating one fully joint filter [80] or one interacting filter [31] (e.g., one modality driving the other). The generic proposal strategy described in [27] is used to carefully combine the optimal proposal strategies for the individual acoustic

and video state spaces such that the random support of the particle filter is concentrated where the final posterior of the joint state space lies. The resulting posterior has a lower Kullback-Leibler distance to the true target posterior than any output combination of the individual filters.

Both the visual and acoustic modalities are examples of projective devices with the planar acoustic microphone array being a 1D sensor. Under this setting, the direction of arrival of the acoustic array is the epipolar line associated with the multi-view constraint. The joint filter state vector includes the target heading direction $\phi_k(t)$, the logarithm of velocity over range $Q_k(t) = \log(v_k/r_k(t))$, observable only by the acoustics; target shape deformation parameters $\{a_1, a_2, a_3, a_4\}_k$, the vertical 2D image plane translation parameter $\eta_k(t)$, observable only by the video; and the target DOA $\theta_k(t)$, observable by both modalities (being the equivalent of the epipolar line). The subscript k refers to the k^{th} target. We also incorporate a time delay variable $\tau_k(t)$ into the filter state vector to account for acoustic propagation delays needed to synchronize the acoustic and video measurements. This variable is necessary to robustly combine the high resolution video modality with the lower resolution acoustic modality and to prevent biases in the state vector estimates.

The filter is initialized using a matching pursuit strategy to generate the particle distribution for each new target, one at a time [25,87]. A partitioning approach is used to create the multiple target state vector, where each partition is assumed to be independent. Moreover, the particle filter importance function independently proposes particles for each target partition to increase the efficiency of the algorithm at moderate increase in computational complexity.

5.1 Acoustic State Space

The acoustic state space, presented in this section, is a modified form of the one used in [26]. we choose this particular acoustic state space because of its flexible

observation model that can handle (i) multiple target harmonics, (ii) acoustic propagation losses, and (iii) time-varying frequency characteristics of the observed target acoustic signals, without changing the filter equations. Figure 5.1 shows the behavior of the acoustic state variables for a two-target example using simulated data.

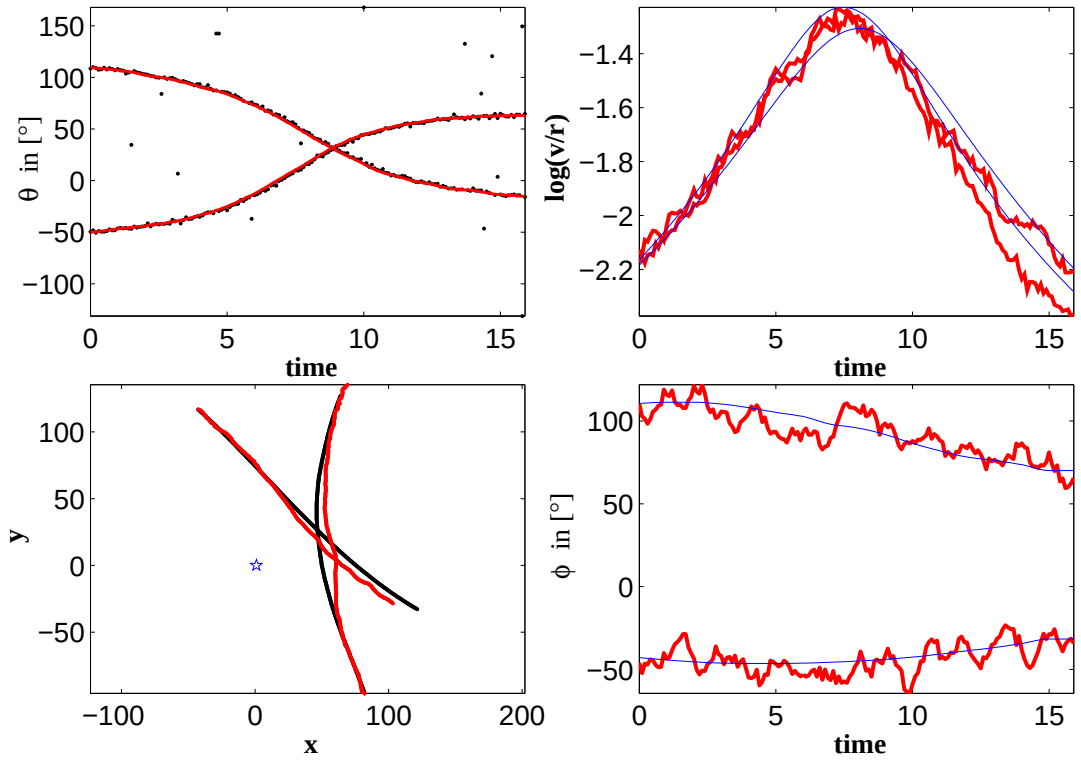


Figure 5.1: (*Top Left*) Particle filter DOA tracking example with two targets. (*Bottom Left*) True track vs. calculated track. Note that the particle filter track is estimated using the filter outputs and the correct initial position. The particle filter jointly estimates the target heading (*Bottom Right*) and the target velocity over range ratio (*Top Right*), while estimating the target bearing. Note that the heading estimates typically tend to be much noisier than the DOA estimates.

5.1.1 State Equation

The acoustic state vector for target k has three elements $x_k(t) \triangleq [\ \theta_k(t)\ ,\ Q_k(t)\ ,\ \phi_k(t)\ ]^T$, where $\theta_k(t)$ is the k^{th} target DOA, $\phi_k(t)$ is its heading direction, and $Q_k(t)$ is its logarithm of the velocity-range ratio. The angular parameters $\theta_k(t)$ and $\phi_k(t)$ are measured counterclockwise with respect to the x-axis.

The state update equation is derived from the geometry imposed by the locally constant velocity model. The resulting state update equation is nonlinear [143,25]:

$$x_k(t+\tau) = h_\tau\left(x_k(t)\right) + u_k(t), \tag{5.1}$$

where $u_k(t) \sim \mathcal{N}\left(0, \Sigma_u\right)$ with $\Sigma_u = \text{diag}\{\sigma^2_{\theta,k}, \sigma^2_{Q,k}, \sigma^2_{\phi,k}\}$ and $h_\tau(x_k(t)) =$

$$\begin{bmatrix} \tan^{-1}\left\{\frac{\sin\theta_k(t)+\tau\exp Q_k(t)\sin\phi_k(t)}{\cos\theta_k(t)+\tau\exp Q_k(t)\cos\phi_k(t)}\right\} \\ Q_k(t) - \frac{1}{2}\log\left\{1 + 2\tau\exp Q_k(t)\cos(\theta_k(t)-\phi_k(t)) + \right. \\ \left. \tau^2\exp(2Q_k(t))\right\} \\ \phi_k(t) \end{bmatrix}. \tag{5.2}$$

Reference [25] also discusses state update equations based on a constant acceleration assumption.

5.1.2 Observation Equation

The observations $\mathbf{y}_{t,f} = \{y_{t-m\tau,f}(p)\}_{m=0}^{M-1}$ consist of a batch of DOA estimates from a beamformer, indexed by m. Hence, the acoustic data of window-length T is segmented into M segments of length τ, equal to a single video frame duration (typically $\tau = 1/30$s). The target motion should satisfy the constant velocity assumption during a window-length T. For ground targets, $T = 1$s is a reasonable choice. Each of these segments is processed by a beamformer, based on the temporal frequency structure of the observed target signals, to calculate possible DOA estimates. This procedure can be repeated F times for each narrow-band

frequency indexed by f (Fig. 5.2). Note that only the peak locations are kept in the beamformer power pattern. Moreover, the peak values, indexed by p, need not be ordered or associated with peaks from the previous time in the batch and the number of peaks retained can be time-dependent.

The sliding batch of DOA's, $\mathbf{y}_{t,f}$, is assumed to form a normally distributed cloud around the true target DOA tracks. In addition, only one DOA is present for each target at each frequency f or the target is missed: multiple DOA measurements imply the presence of clutter or other targets. We also assume that there is a constant detection probability for each target denoted by κ^f, which might depend on the particular frequency f. If the targets are also simultaneously identified, an additional partition dependency, i.e., κ^f_k, is added.

For a given target, if we assume that the data is only due to its partition and clutter (hence, the DOA data corresponding to other targets are treated as clutter), we can derive the observation likelihood for the state $\mathbf{x}_t = \left[\ x_1^T(t),\ x_2^T(t),\ \ldots\ , x_K^T(t)\ \right]^T$ [26] as:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{k=1}^{K} p(\mathbf{y}_t|x_k(t)) = \prod_{k=1}^{K}\prod_{f=1}^{F}\prod_{m=0}^{M-1} \left\{ \kappa^f_{0,1}\left(\frac{\gamma}{2\pi}\right)^{P_{m,f}} + \kappa^f_{1,1}\left(\frac{\gamma}{2\pi}\right)^{P_{m,f}-1} \sum_{p=1}^{P_{m,f}} \frac{\psi_{t,m,f}\left(p\middle|x_k\right)}{P_{m,f}} \right\}, \tag{5.3}$$

where the parameters $\kappa^f_{n,K}$ ($\sum_n \kappa^f_{n,K} = 1$) are the elements of a detection (or confusion) matrix, $p = 0, 1, \ldots, P_{m,f}$ for each f and m, and $\gamma \gg 1$ is a constant that depends on the maximum number of beamformer peaks P, the smoothness of the beamformer's steered response, and the number of targets K. The function ψ in (5.3) is derived from the assumption that the associated target DOA's form a Gaussian distribution around the true target DOA tracks:

$$\psi_{t,m,f}\left(p_i\middle|x_i\right) = \frac{1}{\sqrt{2\pi}\sigma^2_\theta(m,f)} \exp\left\{ -\frac{\left(h^\theta_{m\tau}(x_i(t)) - y_{t+m\tau,f}(p_i)\right)^2}{2\sigma^2_\theta(m,f)} \right\}, \tag{5.4}$$

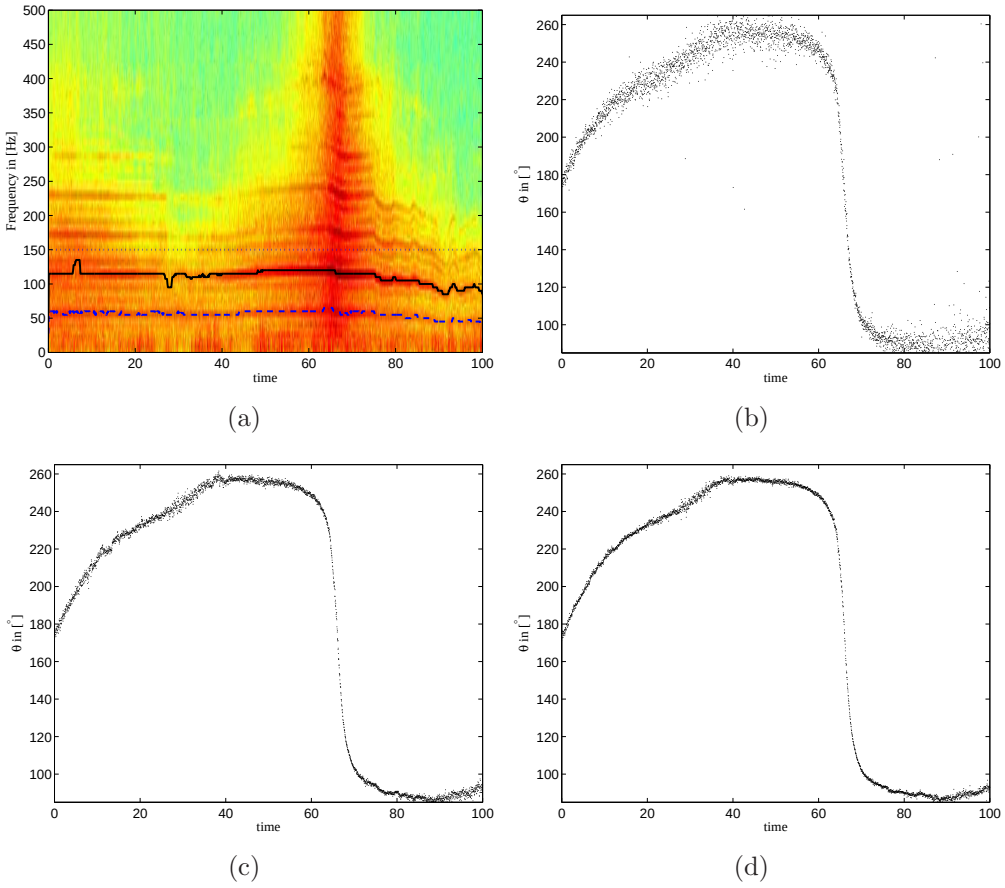


Figure 5.2: A 10-element uniform circular microphone array is used to record a target's acoustic signal, while it is moving on an oval track (refer to Fig. 5.14). The acoustic array's inter-microphone distance is 1.1m. Hence, the maximum beamforming frequency without aliasing is approximately 150Hz. The acoustic sampling frequency is 44100Hz. (a) The time-frequency plot of the received signal. We estimated the bearing track of the vehicle using the MVDR beamformer [68], where the beamforming frequencies are chosen to be the dashed line for (b), the solid line for (c), and the dotted line for (d). For each acoustic bearing estimate, 1470 acoustic data samples are used, corresponding to 30 bearing estimates per second. The bearing tracks in (b-d) are indexed by $f = 1, 2, 3$ in the acoustic state space derivation and $F = 3$.

where the superscript θ on the state update function h refers only to the DOA component of the state update and $\sigma_\theta^2(m, f)$ is supplied by the beamformer, using the curvature of the DOA power pattern at the peak location.

5.2 Video State Space

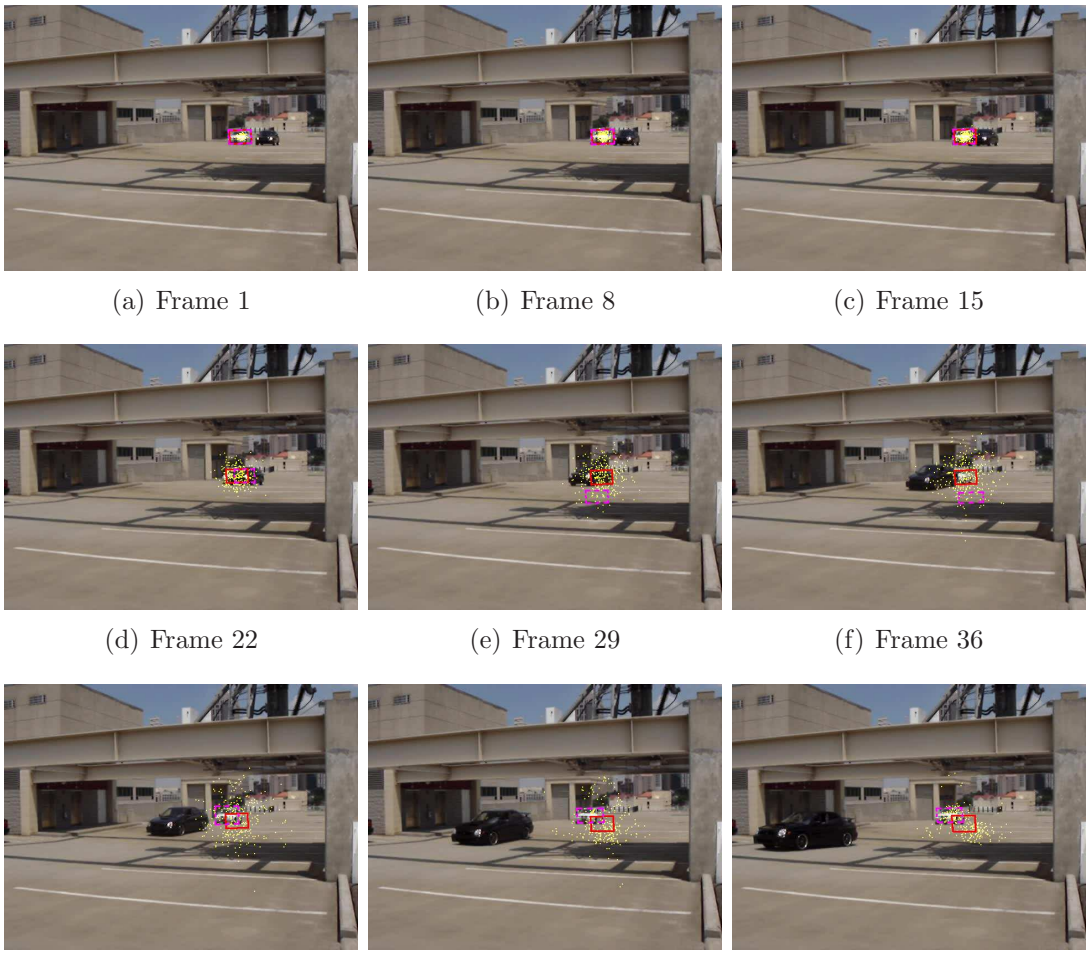
In this section, we give the details of the video state space. This video state space is also described in greater detail in [141]. We assume that the camera is stationary and is mounted at the center of the acoustic microphone array, at a known height above the ground. We also assume that the camera calibration parameters are known, which allows us to convert a location on the image plane to a DOA estimate while having the same reference axis as the acoustic state space. Figure 5.3 demonstrates a video tracker based on state space described in this section.

5.2.1 State Equation

The video state vector for target k has six elements: four affine deformation parameters $\mathbf{a}_k(t) = \left[\ a_{k,1,t}\ ,\ \ldots\ ,\ a_{k,4,t}\ \right]^T$, a vertical 2-D translation parameter $\eta_k(t)$, and the target DOA $\theta_k(t)$: $x_k(t) \triangleq \left[\ \mathbf{a}_k^T(t)\ ,\ \eta_k(t)\ ,\ \theta_k(t)\ \right]^T$. The affine deformation parameters linearly model the object rotation, shear and scaling (affine minus translation), whereas the translation parameter and the DOA account for the object translation, all on the image plane. The state update equation consists of a predictive shift and a diffusion component:

$$x_k(t) = h_\tau\left(x_k(t-\tau)\right) + u_k(t) = \hat{x}_k(t-\tau) + \nu_k(t) + u_k(t), \tag{5.5}$$

where $\nu_k(t)$ is an adaptive velocity component, affecting only $\eta_k(t)$ and $\theta_k(t)$ in the state vector. It is calculated using a first-order linear prediction method on two successive frames; $\hat{x}_k(t-\tau)$ is the maximum *a posteriori* estimate of the state



(a) Frame 1 (b) Frame 8 (c) Frame 15

(d) Frame 22 (e) Frame 29 (f) Frame 36

(g) Frame 43 (h) Frame 50 (i) Frame 57

Figure 5.3: Intensity-based visual tracking of the white car using a particle filter. The solid box shows the mean of the posterior, whereas the dashed box shows the location of the mode of the posterior. The dot cloud depicts spatial particle distribution. In this scenario, the white car is occluded for 1 second corresponding to 30 video frames. The particle spread during occlusion increases because the robust statistics measure [141] renders the likelihood function non-informative. The filter quickly locks back to the target after occlusion.

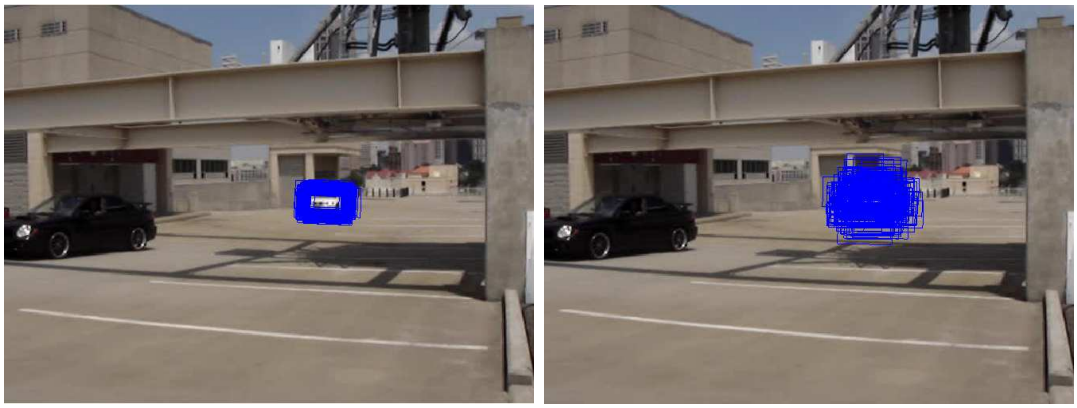
at time $t-\tau$; and $u_k(t)$ is an adaptive noise component, calculated by measuring the difference between the updated appearance and the calculated appearance at time t, as described in [141]. Note that the video state mode estimates $\hat{x}_k(t-\tau)$ are stored in the memory, because they are later used for adaptively determining a time delay variable for acoustic-video synchronization.

The state equation is constructed so that it can effectively capture rapid target motions. The adaptive velocity component accounts for the object's shift within the image frame, whereas the adaptive noise term captures its drift around its motion. Hence, the adaptive velocity model simply encodes the object's inertia into the tracker and generates particles that are tightly centered around the object of interest for improved efficiency (Fig. 5.4). If we do not account for the object's shift using the adaptive noise component, we need to increase the variance of the drift component to capture the actual movement of the object. Hence, we may start to lose our focus on the target as shown in Fig. 5.4(b) without the adaptive velocity component. In this case, if the background is somewhat similar to the target, it is automatically injected into the appearance models through the EM algorithm. Hence, the background also becomes part of the tracked object, thereby creating local minima to confuse the tracker in its later iterations.

The adaptive noise variance is based on residual motion errors generated by the adaptive velocity component. It decreases when the quality of the prediction from the adaptive velocity component is high, and increases when the prediction is poor. Finally, when the tracker is visually occluded (occlusion is defined in the next subsection), the target motion is characterized using a Brownian motion and $\nu_k(t) = 0$ is enforced. Hence, during an occlusion, the state dynamics changes to the following form:

$$x_k(t) = x_k(t-\tau) + u_k(t). \tag{5.6}$$

We avoid the use of the adaptive velocity model during occlusion because the object motion may change significantly during an occlusion.



(a) with the adaptive velocity model (b) without the adaptive velocity model

Figure 5.4: Comparison of the proposed particles when the adaptive velocity model is used. Note that the particles are tightly clustered around the target when we use the adaptive velocity model. In contrast, without velocity prediction, we need to use more particles to represent the same posterior, because most particles have very low weights.

5.2.2 Observation Equation

The observation model is a mixture of following adaptive appearance models: a wandering $\mathcal{W}_t$, a stable $\mathcal{S}_t$, and an optional fixed template model $\mathcal{F}_t$. The wandering model $\mathcal{W}_t$ captures transient appearance changes based on two successive frames, whereas the stable model $\mathcal{S}_t$ encodes appearance properties that remain relatively constant over a large number of frames (Fig. 5.5). The fixed template $\mathcal{F}_t$ is useful for tracking recognized targets, however it is not considered any further for this work. The adaptive observation model uses the pixel intensity values for these appearance models for computational efficiency as suggested in [141]. Although the image intensity values are typically not robust to changes in illumination, the appearance model described here can adapt to changes in illumination. However, it is still possible to lose track if there are sudden changes in illumination. We use a very simple model to circumvent this problem. We normalize the mean and the variance of the appearance as seen by each particle. This makes our tracker immune to uniform scaling of the intensities. If we know that the illu-

mination changes are severe, we can adopt an alternative feature at the expense of computation without chancing our filter mechanics, such as the spatial phase data of the object [66] that is more robust to illumination changes.

The observation model is dynamically updated by an online expectation maximization (EM) algorithm that adaptively calculates the appearance parameters $\{\mu_{i,t}, \sigma_{i,t}^2\}$, $(i = w, s)$ of the appearance models $A_t = \{\mathcal{W}_t, \mathcal{S}_t\}$, and the model mixture probabilities $m_{i,t}$, $(i = w, s)$ for each pixel [38, 111]. The details of the EM algorithm for calculating the mixture probabilities and model parameters can be found in [141, 66]. Omitting the details of the derivations, the observation likelihood is given by the following expression:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{k=1}^{K} \prod_{j=1}^{d} \left\{ \sum_{i=w,s} m_{i,t} \mathrm{N}(\mathcal{T}_k(y_t(j)); \mu_{i,t}(j), \sigma_{i,t}^2(j)) \right\}, \tag{5.7}$$

where $\mathcal{T}_k$ is the affine transformation that extracts the image patch of interest by using the state vector $x_k(t)$; d is the number of pixels in the image patch; and $\mathrm{N}(x; \mu, \sigma^2)$ is the density

$$\mathrm{N}(u; \mu, \sigma^2) \propto \exp\left\{ -\rho\left(\frac{u - \mu}{\sigma} \right) \right\}, \tag{5.8}$$

where u is normalized to have unit variance, and

$$\rho(u) = \begin{cases} \frac{1}{2}u^2, & \text{if } |u| \leq c; \\ c\,|u| - \frac{1}{2}c^2, & \text{o/w.} \end{cases} \tag{5.9}$$

The function $\rho(\cdot)$ is Huber's criterion function, which is commonly used for outlier rejection [63]. It provides a compromise between mean estimators that are susceptible to outliers and median estimators that are usually robust to outliers. The constant c is used to determine the outlier pixels that cannot be explained by the underlying models. Furthermore, methods from robust statistics allow us to for-

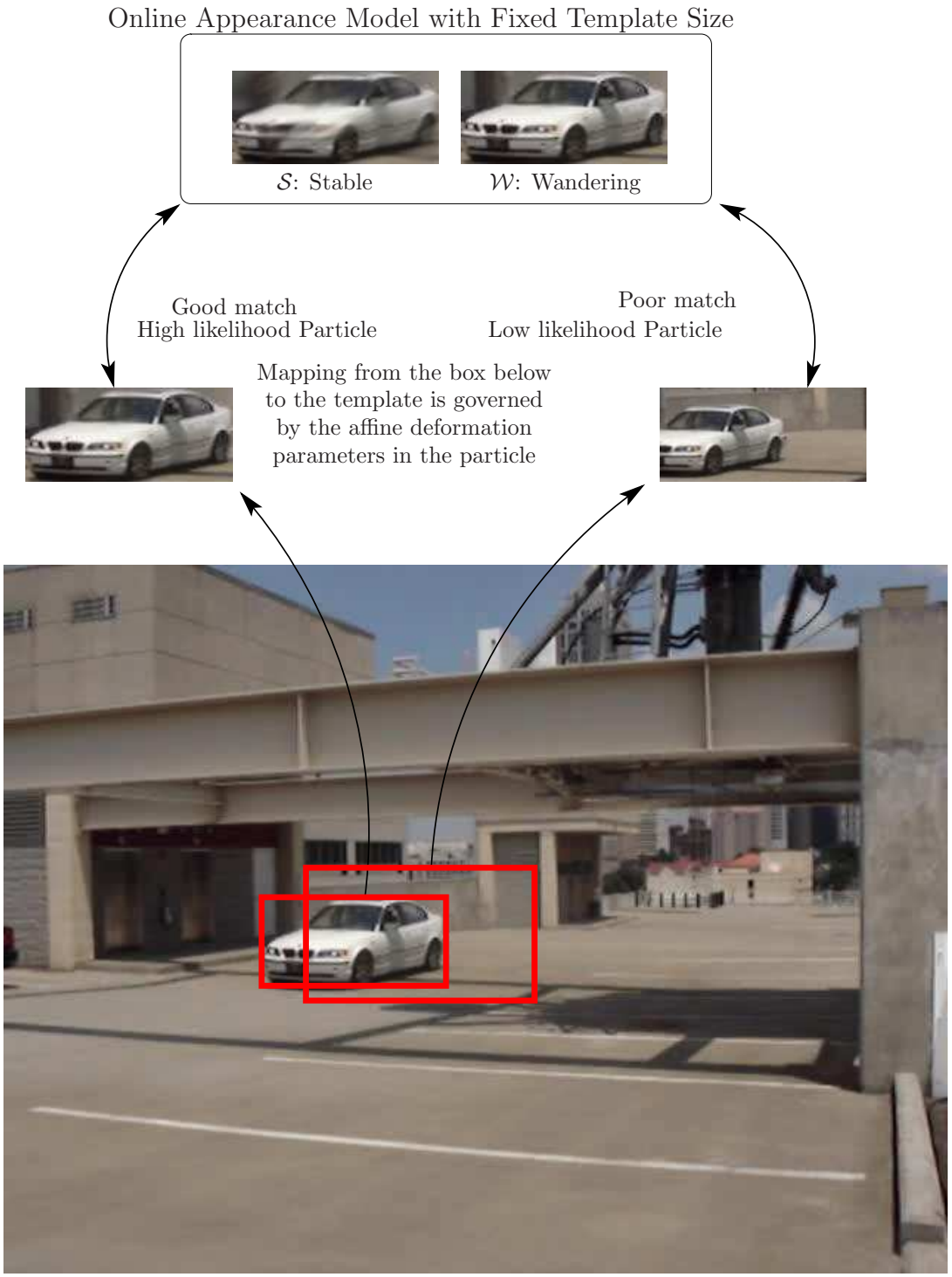


Figure 5.5: The online appearance model is illustrated. The model has two components: $\mathcal{S}$ (stable) and $\mathcal{W}$ (wandering). Note that each model uses a fixed size image template that is updated by an online EM algorithm [141]. To determine a particle's likelihood, an image patch is first determined using the particle elements. Then, the patch is mapped back to the template domain using the affine transformation parameters, where it is compared with the updated appearance model.

mally decide when the tracker is *visually occluded*, which implies that the particle with the highest likelihood has more than 50% of its pixels, which are classified as outliers by the appearance model. This criterion is discussed in greater detail in [141].

Deciding on whether or not an object is occluded is an arduous task. However, this task is alleviated when we also track the appearance. Our decision is based on the outlier statistics and is reliable. We provide a Monte Carlo run of the occlusion decision in the simulations section to show the reliability of our occlusion strategy. We show that the variability of the occlusion detection is rather small once a threshold is chosen. Further examples of this occlusion strategy can be found in [141]. The influence of an error on this decision is discussed in our observation model. If we are late in declaring an occlusion, the appearance of the occluding object injects itself into the target appearance, thereby causing local minima in the tracking algorithm. However, given the complexity of the problem, one should not expect superlative performance for all the possible cases.

Another issue in handling occlusion is the change in the appearance of the target during occlusion. This could happen due to changes in global illumination, changes in the pose of the target, or dramatic changes in the projected target size on the image plane. Recovery of visual tracking cannot be guaranteed, except when these changes are not severe. In cases, where the track is recovered, we update the appearance model using the appearance associated with the particle with maximum likelihood. We say that track has been regained after occlusion, when the tracker is not visually occluded (as defined before) for a fixed set of frames (ten frames for the experiments).

5.3 Bayesian Framework for Tracking the Joint state space

In this section, a Bayesian framework is described for combining the acoustic ($\mathcal{S}_1$) and video ($\mathcal{S}_2$) state spaces that share a common state parameter. The results

below can be generalized to time-varying systems including nuisance parameters. It is assumed that the state dimensions are constant even if the system is time-varying. Define

$$\begin{aligned} \mathcal{S}_{\mathrm{i}}: \qquad x_{i,t} &= \begin{bmatrix} \chi_t \\ \psi_{i,t} \end{bmatrix} \sim q_i(x_{i,t}|x_{i,t-1}) \\ y_{i,t} &\sim f_i(y_{i,t}|x_{i,t}), \end{aligned} \tag{5.10}$$

where the observed data in each space is represented by $\{y_{i,t}, i = 1, 2\}$, $\chi_t = \theta_t$ (overlapping state parameter), $\psi_{1,t} = [\ Q(t)\ ,\ \phi(t)\ ]^T$, and $\psi_{2,t} = \left[\ \mathbf{a}^T(t)\ ,\ \eta(t)\ \right]^T$. The state transition density functions $q_i(\cdot|-)$ are given by (5.1) and (5.5). The observations are explained through the density functions $f_i(\cdot|-)$, given by (5.3) and (5.7). The observation sets y_i are modeled as statistically independent given the state through conditionally independent observation densities. This assumption is justified in our problem: for example, a vehicle's time-frequency signature is independent of its colors or textures. In most cases, it may be necessary to verify this assumption mathematically for the problem at hand [80, 83] by using the specific observation models.

To track the joint state vector $x_t = [\chi_t, \psi_{1,t}, \psi_{2,t}]$ with a particle filter, the following target posterior should be determined:

$$\begin{aligned} p(x_t|x_{t-1}, y_{1,t}, y_{2,t}) &\propto p(y_{1,t}, y_{2,t}|x_t)p(x_t|x_{t-1}) \\ &= \pi_t(y_{1,t}, y_{2,t})\pi_{t-1}(x_t), \end{aligned} \tag{5.11}$$

where $\pi_s(\cdot) = p(\cdot|x_s)$. Note that the Markovian property is enforced in (5.11). That is, given the previous state and the current data observations, the current state distribution does not depend on the previous state track and the previous observations.

Equation (5.11) allows the target posterior to be calculated up to a proportionality constant, where the proportionality is independent of the current state

x_t. The first pdf on the right hand side of (5.11) is called the joint-data likelihood and can be simplified, using the conditional independence assumption on the observations:

$$\pi_t(y_{1,t}, y_{2,t}) = f_1(y_{1,t}|x_{1,t}) f_2(y_{2,t}|x_{2,t}). \tag{5.12}$$

The second pdf in (5.11), corresponding to a joint state update, requires more attention. State spaces $\mathcal{S}_1$ and $\mathcal{S}_2$ may have different updates for the common parameter set since they had different models.[1] This poses a challenge in terms of formulating the common state update for x_t. Instead of assuming a given analytical form for the joint state update as in [80], we combine the individual state update marginal pdfs for the common state parameter as follows:

$$\pi_{t-1}(\chi_t) = c p_1(\chi_t)^{o_1} p_2(\chi_t)^{o_2} r(\chi_t)^{o_3}, \tag{5.13}$$

where $c \geq 1$ is a constant, $p_i(\chi_t) \triangleq p(\chi_t|x_{i,t-1})$ is the marginal density, the probabilities o_i for $i = 1, 2$ ($\sum_i o_i = 1$) define an ownership of the underlying phenomenon by the state models, and $r(\chi_t)$ is a (uniform/reference) prior in the natural space of the parameter χ_t [13] to account for unexplained observations by the state models.

If we denote the Kullback-Leibler distance as D, then

$$D(\alpha(\chi_t)||\pi_{t-1}(\chi_t)) = -\log c + \sum_i o_i D(\alpha(\chi_t)||p_i(\chi_t)) \tag{5.14}$$

where α is the unknown true χ_t distribution. Hence, $D(\alpha||\pi_{t-1}) \leq \max_i\{D(\alpha||p_i)\}$. $\pi_{t-1}(\chi_t)$ always has a smaller KL distance to the true distribution than the maximum KL distance of $p_i(\chi_t)$. This implies that (5.13) alleviates the worst case divergence from the true distribution [2]. Hence, this proves that one of the track-

[1] There is no exact state update function for all targets. Individual state spaces may employ different functions for robustness, which is the case in our problem.

ers does assist the other in this framework.

The ownership probabilities, o_i, can be determined using an error criteria. For example, one way is to monitor how well each partition $x_{i,t}$ in x_t explains the information streams $y_{i,t}$ through their state-observation equation pair defined by $\mathcal{S}_i$, (5.10). Then, the respective likelihood functions can be aggregated with an exponential envelope to recursively solve for the o_i's (e.g., using an EM algorithm). In this case, the target posterior will be dynamically shifting towards the better self-consistent model while still taking into account the information coming from the other, possibly incomplete, model, which might be temporarily unable to explain the data stream.

If one believes that both models explain the underlying process equally well regardless of their self-consistency, one can set $o_1 = o_2 = 1/2$ to have the marginal distribution of χ_t resemble the product of the marginal distributions imposed by both state spaces. The proposal strategy in the next section is derived with this assumption on the ownership probabilities, because, interestingly, it is possible to show that assuming equal ownership probabilities along with (5.13) leads to the following conditional independence relation on the state spaces:

$$\pi_{t-1}(x_{1,t})\pi_{t-1}(x_{2,t}) = q_1(x_{1,t}|x_{1,t-1})q_2(x_{2,t}|x_{2,t-1}). \tag{5.15}$$

Equation (5.15) finally results in the following update equation:

$$\begin{aligned} \pi_{t-1}(x_t) &= \pi_{t-1}(\psi_{1,t}, \psi_{2,t}|\chi_t)\pi_{t-1}(\chi_t) \\ &= \pi_{t-1}(\psi_{1,t}|\chi_t)\pi_{t-1}(\psi_{2,t}|\chi_t)\pi_{t-1}(\chi_t) \\ &= \frac{\pi_{t-1}(x_{1,t})\pi_{t-1}(x_{2,t})}{\pi_{t-1}(\chi_t)} \\ \Rightarrow \pi_{t-1}(x_t) &= \frac{q_1(x_{1,t}|x_{1,t-1})q_2(x_{2,t}|x_{2,t})}{\pi_{t-1}(\chi_t)}, \end{aligned} \tag{5.16}$$

where

$$\pi_{t-1}(\chi_t) \propto \left[\iint q_1(x_{1,t}|x_{1,t-1}) d\psi_{1,t} q_2(x_{2,t}|x_{2,t}) d\psi_{2,t} \right]^{1/2}. \tag{5.17}$$

5.4 Proposal Strategy

A proposal function, denoted as $g(x_t|x_{t-1}, y_t)$, determines the random support for the particle candidates to be weighted by the particle filter. Two very popular choices are (i) the state update $g \propto q_i(x_t|x_{t-1})$ and (ii) the full posterior $g \propto f_i(y_t|x_t)q_i(x_t|x_{t-1})$. The first one is attractive because it is analytically tractable. The second one is better because it incorporates the latest data while proposing particles, and it results in less variance in the importance weights of the particle filter since, in effect, it directly samples the posterior [84, 42]. Moreover, it can be analytically approximated for faster particle generation by using local linearization techniques (see [42]), where the full posterior is approximated by a Gaussian. The analytical form of the proposal functions for acoustic and video state spaces, obtained by local linearization of the posterior, is given by

$$g(x_t|x_{t-1}, y_t) \sim \mathcal{N}(\mu_g, \Sigma_g), \tag{5.18}$$

where the Gaussian density parameters are

$$\begin{aligned} \Sigma_g &= \left(\Sigma_y^{-1} + \Sigma_u^{-1}\right)^{-1}, \\ \mu_g &= \Sigma_g \left(\Sigma_y^{-1} x_{\text{mode}} + \Sigma_u^{-1} h_\tau(x(t-\tau))\right), \end{aligned} \tag{5.19}$$

and where x_{mode} is the mode of the data likelihood, and $\Sigma_y^{-1}(k)$ is the Hessian of data likelihood at x_{mode}. The details of these proposal functions can be found in [25, 141]. Hence, in either way of proposing particles, one can assume that an analytical relation for g_i, defining the support of the actual posterior for each state space, can be obtained.

Figure 5.6 describes the proposal strategy used for the joint state space. Each

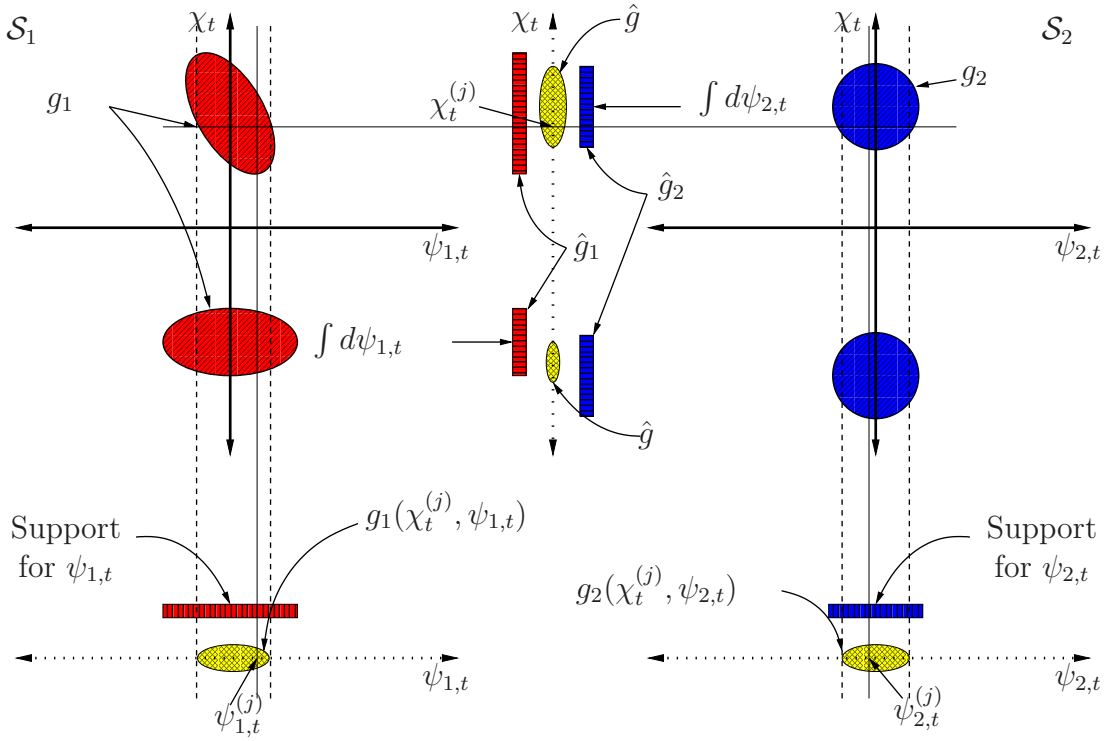


Figure 5.6: The supports, g_i's, for the posterior distribution in each state space, $\mathcal{S}_i$, are shown on the axes χ_t vs. $\psi_{i,t}$. Particles for the joint state are generated by first generating χ_t's from the combined supports of the marginal distributions of χ_t. Then, the $\psi_{i,t}$'s are sampled from the g_i's as constrained by the given χ_t realization.

state space has a proposal strategy described by the analytical functions $\{g_i, i = 1, 2\}$ defined over the whole state spaces. Then, the proposal functions of each state g_i are used to propose particles for the joint space by carefully combining the supports of the individual posteriors. First, marginalize out the parameters $\psi_{i,t}$:

$$\hat{g}_i(\chi_t|x_{i,t-1}, y_{i,t}) = \int g_i(x_{i,t}|x_{i,t-1}, y_{i,t})d\psi_{i,t}. \tag{5.20}$$

The functions, $\hat{g}_i$, describe the random support for the common state parameter χ_t and can be combined in the same way as the joint state update (5.13). Hence, the following function

$$\hat{g}(\chi_t|x_{t-1}, y_{1,t}, y_{2,t}) \propto [\hat{g}_1(\chi_t|x_{1,t-1}, y_{1,t})\hat{g}_2(\chi_t|x_{2,t-1}, y_{2,t})]^{1/2} \tag{5.21}$$

can be used to generate the candidates $\chi_t^{(j)}$ for the overlapping state parameters. Then using $\chi_t^{(j)}$, one can generate $\psi_{i,t}^{(j)}$ from $g_i(\chi_t^{(j)}, \psi_{i,t}|x_{i,t-1}, y_{i,t})$ and form $x_t^{(j)} = [\chi_t^{(j)}, \psi_t^{(j)}, \varphi_t^{(j)}]$.

In general, Monté-Carlo simulation methods can be used to simulate the marginal integrals in this section [112]. Here, we show how to calculate the marginal integrals of the state models. Simulation of the other integrals are quite similar. Given $\chi_t^{(j)}$, draw M samples using $\psi_{i,t}^{(m)} \sim g_i(\chi_t^{(j)}, \psi_{i,t}|x_{i,t-1}, y_{i,t})$.[2] Then,

$$\int q_1(\chi_t^{(j)}, \psi_{i,t}|x_{1,t-1})d\psi_{i,t} \approx \frac{1}{M}\sum_{m=1}^{M} \frac{q_1(\chi_t^{(j)}, \psi_{i,t}^{(m)}|x_{1,t-1})}{g_1(\chi_t^{(j)}, \psi_{i,t}^{(m)}|x_{1,t-1}, y_{1,t})}. \tag{5.22}$$

The pseudo-code for the joint strategy is given in Table 5.1. Finally, the importance weights for the particles generated by the joint strategy described in

[2]It is actually not necessary to draw the samples directly from $g_i(\chi_t^{(j)}, \psi_{i,t}|-)$. An easier distribution function approximating only q_i can be used for simulating the marginalization integral (5.22).

Table 5.1: Pseudo Code for Joint Proposal Strategy

i. Given the state update q_i and observation relations f_i for the individual state spaces $\{\mathcal{S}_i, i = 1, 2\}$, determine analytical relations for the proposal functions g_i's. For the individual proposal functions g_i, it is important to approximate the true posterior as close as possible because these approximations are used to define the random support for the final joint posterior. For this purpose, Gaussian approximation of the posterior (5.18) or linearization of the state equations can be used [42].

ii. Determine the support for the common state parameter χ_t using (5.21). The expression for $\hat{g}$ may have to be approximated or simulated to generate candidates $\chi_t^{(j)}$, $j = 1, 2, \ldots, N$ where N is the number of particles.

iii. Given $\chi_t^{(j)}$,

- calculate the marginal integrals by using (5.22) to determine g_i,
- generate $\psi_{i,t}^{(j)} \sim g_i(\chi_t^{(j)}, \psi_{i,t} | x_{i,t-1}, y_{i,t})$,
- form $x_t^{(j)} = [\chi_t^{(j)}, \psi_{1,t}^{(j)}, \psi_{2,t}^{(j)}]$, and
- calculate the importance weights, $w^{(j)}$'s, using (5.23).

this section can be calculated as follows:

$$w^{(j)} \propto \frac{p(x_t^{(j)} | x_{t-1}, y_{1,t}, y_{2,t}) \hat{g}(\chi_t^{(j)} | x_{t-1}, y_{1,t}, y_{2,t})}{g_1(\chi_t^{(j)}, \psi_{1,t}^{(j)} | x_{1,t-1}, y_{1,t}) g_2(\chi_t^{(j)}, \psi_{2,t}^{(j)} | x_{2,t-1}, y_{2,t})}. \tag{5.23}$$

5.5 Time Delay Parameter

The joint acoustic video particle filter sequentially estimates its state vector at video frame rate, as the acoustic data arrives. Hence, the joint filter state estimates are delayed with respect to the actual event that produces the state, because the acoustic information propagates much slower than the video informa-

tion. Although it is possible to formulate a filter so that estimates are computed as the video data arrives, the resulting filter cannot use the delayed acoustic data. Hence, it is not considered here. The adaptive time delay estimation also allows position tracking on the ground plane. However, small errors in the time delay estimates translate into rather large errors in target range estimates, resulting in large errors in target position estimates. Hence, the main reason for estimating time delay is to ensure the stability of the joint filter.

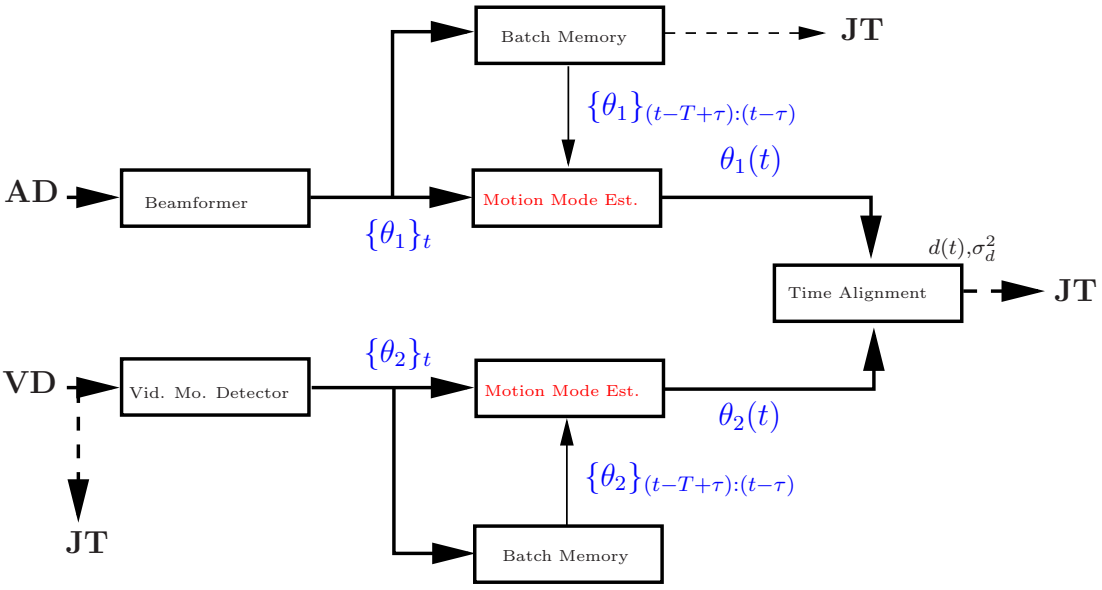


Figure 5.7: At time t, τ seconds of acoustic data (**AD**) and a frame of video data (**VD**) are processed to obtain possible target DOA's $\{\theta_i\}_t$. This preprocessing is done by a beamformer block and a video motion detector block, respectively. With the guidance of the joint tracker (**JT**), these DOA's are used to determine the DOA mode tracks, $\theta_i(t)$ (Fig. 5.8), to estimate the time delay $d(t)$. The estimated time delay parameters are then used in the proposal function of the joint tracker.

To synchronize the audio-video information, we add an additional time delay variable $d_k(t)$ for each target k to form an augmented joint filter state:

$$x_k(t) \triangleq \left[\ \mathbf{a}_k^T(t)\ ,\ \eta_k(t)\ ,\ \theta_k(t)\ ,\ Q_k(t)\ ,\ \phi_k(t)\ ,\ d_k(t)\ \right]^T. \tag{5.24}$$

The time delay $d_k(t)$ is defined geometrically as:

$$d_k(t) = ||\boldsymbol{\xi} - \boldsymbol{\chi}_k\left(t - d_k(t)\right)||/c, \tag{5.25}$$

where $\boldsymbol{\xi} = [s_x, s_y]^T$ is the hybrid node position in $2D$, $\boldsymbol{\chi}_t = [x_{k,target}(t), y_{k,target}(t)]^T$ is the k^{th} target position, and c is the speed of sound. Using the geometry of the problem, it is possible to derive an update equation for $d_k(t)$:

$$\begin{aligned} &d_k(t+\tau) = d_k(t)\exp\{u_{d,k}(t)\} \\ &\sqrt{1 + 2\tau\exp\{Q_k(t)\}\cos\left(\theta_k(t) - \phi_k(t)\right) + \tau^2\exp\{2Q_k(t)\}}, \end{aligned} \tag{5.26}$$

where the Gaussian state noise $u_{d,k}(t)$ is injected as multiplicative.

We suppress the partition dependence on the variables from now on for brevity. Figure 5.7 illustrates the mechanics of time delay estimation. To determine $d(t)$, we first determine the mode of the acoustic state vector within a batch period of T seconds. Given the calculated acoustic data mode, which is also used in the proposal stage of the particle filter, $x_{1,\text{mode}}(t)$, an analytical relation for acoustic DOA track $\theta_1(t)$ (Fig. 5.8) is determined, using the state update function (5.2). This functional estimate $\theta_1(t)$ of the acoustic DOA's and acoustic data is used to determine an average variance of the DOA's $\widetilde{\sigma}^2_{1,\theta}$ around the functional, between times t and $t - T$. Note that $\widetilde{\sigma}^2_{\theta}$ is estimated using the missing and spurious data assumptions similar to the ones presented in Sect. 5.1.

Next, we search the stored mode estimates of the video state, which is used in the video state update function (5.5), to determine $M = T/\tau$ (i.e., the number of video frames per second) closest video DOA estimates. These DOA's are used, along with the constant velocity motion assumption, to determine a functional estimate $\theta_2(t)$ of the DOA track and an average DOA variance $\widetilde{\sigma}^2_{2,\theta}$, based on the video observations, as shown in Fig. 5.8. The observation likelihood for the time

delay variable $d(t)$ is approximated by the following Gaussian:

$$p(d(t)|\mathbf{y}_{1,t}, \mathbf{y}_{2,t}) \approx \mathcal{N}\left(\mu_d \left(1 + Te^{Q_{\text{mode}}} \cos\left[(\theta_1(t-T) + \theta_1(t))/2 - \phi_{\text{mode}}\right] + T^2 e^{2Q_{\text{mode}}}/4\right)^{\frac{1}{2}}, \sigma_d^2\right), \tag{5.27}$$

where the mean is the average distance between the functional inverses of $\theta_1(t)$ and $\theta_2(t)$:

$$\mu_d = \left| \frac{\int_{\theta_1(t)}^{\theta_1(t-T)} \left[\theta_1^{-1}(\theta') - \theta_2^{-1}(\theta')\right] d\theta'}{\theta_1(t) - \theta_1(t-T)} \right|. \tag{5.28}$$

The variance σ_d^2 is determined by dividing the average DOA variances by the functional slope average:

$$\sigma_d^2 = \left| \frac{\theta_1(t) - \theta_1(t-T)}{\int_{t-T}^{t} \frac{\partial \theta_1(t')}{\partial t'} dt'} \right| \widetilde{\sigma}_{1,\theta}^2 + \left| \frac{\theta_1(t) - \theta_1(t-T)}{\int_{t-T}^{t} \frac{\partial \theta_2(t')}{\partial t'} dt'} \right| \widetilde{\sigma}_{2,\theta}^2. \tag{5.29}$$

In the joint filter, the particles for the time delay parameter are independently proposed with a Gaussian approximation to the full time delay posterior, using (5.26) and (5.27) [42].

5.6 Algorithm Details

The joint acoustic-video particle filter tracker code is given in Table 5.2. In the following subsections, we discuss other practical aspects of the filter.

5.6.1 Initialization

The initialization algorithms for the video and acoustic trackers are employed to initialize the joint filter. The joint filter initialization requires an interplay between the modalities, because the state vector is only partially observable by either modality. In most cases, the video initializer is cued by the acoustics, because the video modality consumes significantly more power. Below, we describe the

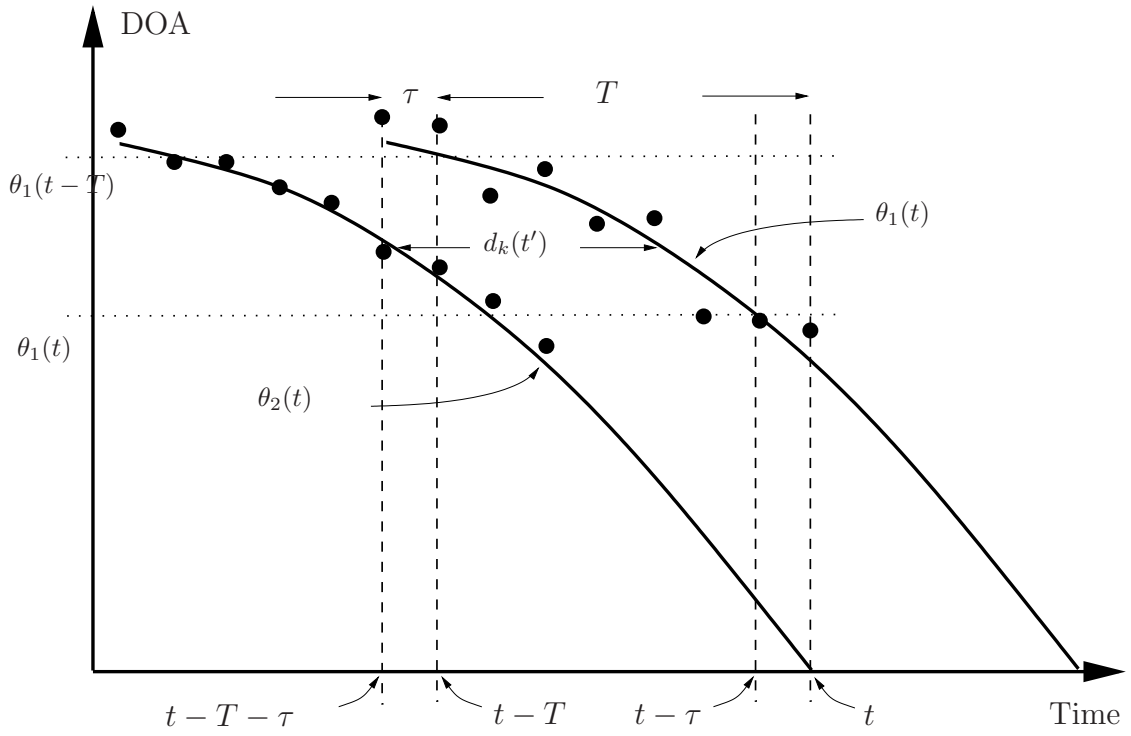


Figure 5.8: The time delay $d_k(t)$ between the acoustic and video DOA tracks, $\theta_1(t)$ and $\theta_2(t)$, respectively.

general case where each modality is turned on.

Briefly, the organic initialization algorithms work as follows. In video, motion cues and background modeling are used to initialize target appearance models, $\mathbf{a}_k^T(t)$, $\eta_k(t)$, and $\theta_k(t)$ by placing a bounding box on targets and by coherent temporal processing of the video frames [141]. In acoustics, the temporal consistency of the observed DOA's is used to initialize target partitions by using a modified Metropolis-Hastings algorithm [28, 25].

To initialize targets, a matching-pursuit idea is used [87, 25]. The most likely target is initialized first and then its corresponding data is gated out [5]. Note that the target motion parameters alleviate the data association issues between the video and acoustic sensors, because both modalities are collocated. Hence, the overlapping state parameter θ is used to fuse the video shape parameters and acoustic motion parameters.

When a target is detected by the organic initialization algorithms, the time delay variable is estimated using the scheme described in Sect. 5.5. The initialization scheme in [25] is used to determine the target motion parameters, where the video DOA mode estimates are used as an independent observation dimension to improve the accuracy. Finally, a target partition is deleted by the tracking algorithm at the proposal stage if both acoustic and video modalities do not see any data in the vicinity of the proposed target state.

5.6.2 Multi Target Posterior

The joint filter treats the multiple targets independently, using a partition approach. The proposal and particle weighting of each target partitions are independent. This allows a parallel implementation of the filter where a new single target tracking joint filter is employed for each new target. Hence, the complexity of the filter increases linearly with the number of targets. Note that for each target partition, it is crucial that data corresponding to the other target partitions are treated as clutter. This approach is different from the joint probability density

association (JPDA) approach that would be optimal for assigning probabilities to each partition by adding mixtures that consist of data permutations and partition combinations [5]. In JPDA, no data would be assigned to more than one target. However, in our approach, the same DOA might be assigned to multiple targets.

Notably, it is shown in [25] that the independence assumption for the joint state space is reasonable for the acoustic tracker. There is a slight performance degradation in bearing estimation, when the targets cross; however, it is not noticeable in most cases. Moreover, the JPDA approach is not required by the video tracker. When the targets cross, if the targets are not occluding each other as their DOA's cross, the vertical 2-D translation parameter $\eta_k(t)$ resolves the data association issue between the partitions. The motion parameters also resolve the data association, similar to the acoustic tracker, to alleviate the filter performance. If there is occlusion, it is handled separately using robust statistics as described below.

5.6.3 Handling Occlusion

In video, if the number of outlier pixels, defined in (5.9), is above some threshold, occlusion is declared. In that case, the updates on the appearance model and the adaptive velocity component in the state update (5.5) are stopped. The current appearance model is kept and the state is diffused with increasing diffusion variance. The data likelihood for the occluded target is set to 1 for an uninformative response under the influence of robust statistics. Similarly, the acoustic data likelihood is set to 1 when the number of DOA's within the batch gate of a partition is less than some threshold (e.g., M/2).

5.7 Simulations

Our objective with the simulations is to demonstrate the robustness and capabilities of the proposed tracker. We provide two examples. In the first example, a

Table 5.2: Joint Acoustic Video Particle Filter Tracker Pseudo-Code

1. For each particle i $(i = 1, 2, \ldots, N)$ and each partition k $(k = 1, 2, \ldots, K)$
   - Sample the time delay $d_k^{(i)}(t) \sim g_d(d_k(t)|y_{1,t}, y_{2,t}, x_k^{(i)}(t-T))$, where $g_d(\cdot)$ is the Gaussian approximation to (5.26) and (5.27).
   - Using the procedure illustrated in Table 5.1, sample $\chi_k^{(i)}(t)$, $\psi_k^{(i)}(t)$, and $\varphi_k^{(i)}(t)$ from $x_k^{(i)}(t-T)$ with the time synchronized acoustic and video data $y_{1,t}$ and $y_{2,t-d^{(i)}(t)}$.
2. Calculate the weights $w_t^{*(i)}$ using (5.23). Determine visual and acoustic occlusions by looking at the likelihood estimates of each particle: $p(y_{1,t}|\chi^{(i)}(t), \psi^{(i)}(t))$ (acoustics) and $p(y_{2,t}|\chi^{(i)}(t), \varphi^{(i)}(t))$ (video). separately as in [25].
3. Calculate the weights using (5.23) and normalize.
4. Perform the estimation [42]: $E\{f(\mathbf{x}_t)\} = \sum_{i=1}^{N} w_t^{(i)} f(\mathbf{x}_t^{(i)})$.
5. Resample the particles: Only states that are observable participate in resampling. For example, if the observations are visually occluded then the states $\varphi(t)$ are not resampled. Similarly, if the observations are acoustically occluded, then the states $\psi(t)$ are not resampled.
6. Update the appearance model with the appearance corresponding to the particle with maximum likelihood, if this likelihood value exceeds the threshold.

vehicle is visually occluded and the acoustic mode enables track recovery. In the second example, we provide joint tracking of two targets and provide time delay estimation results.

5.7.1 Tracking through Occlusion

Figure 5.9 shows the tracking results for a car that is occluded by a tree. The role of the DOA variable in the state space is crucial for this case. In the absence of information from any one of the modalities, the DOA still remains observable and is estimated from the modality that is not occluded. However, the rest of the states corresponding to the failed modality remains unobservable, and the variance of the particles along these dimensions continues to increase as the occlusion persists. Hence, it is therefore sometimes necessary to use an increasing number of particles to regain track until the failed modality is rectified.

The video modality regains the track immediately, as the target comes out of occlusion. The spread of particles (the dot cloud in Fig. 5.9) gives an idea of the observability of the vertical location parameter on the image plane. Further, the dramatic reduction in this spread as the target comes out of occlusion, demonstrates the previously unobservable visual components recovering the track. It is also interesting to compare the spread of particles in Fig. 5.9 with the pure visual tracking example in Fig. 5.3, where the spread of particle increases isotropically on the image plane, due to complete occlusion. Hence, the joint tracking reduces the uncertainty through the second modality. For this example, the simulation parameters are given in Table 5.3. The acoustic bearing data is generated by adding Gaussian noise to the bearing track that corresponds to the ground truth. The acoustic bearing variance is 4 degrees between $t = 1$s to $t = 5$s, when the vehicle engine is getting occluded by the tree. It is 2 degrees when the vehicle engine is not occluded.

Figure 5.10 shows the results of a Monte-Carlo run, where the filter is rerun with different acoustic noise realizations. The threshold for declaring an occlusion

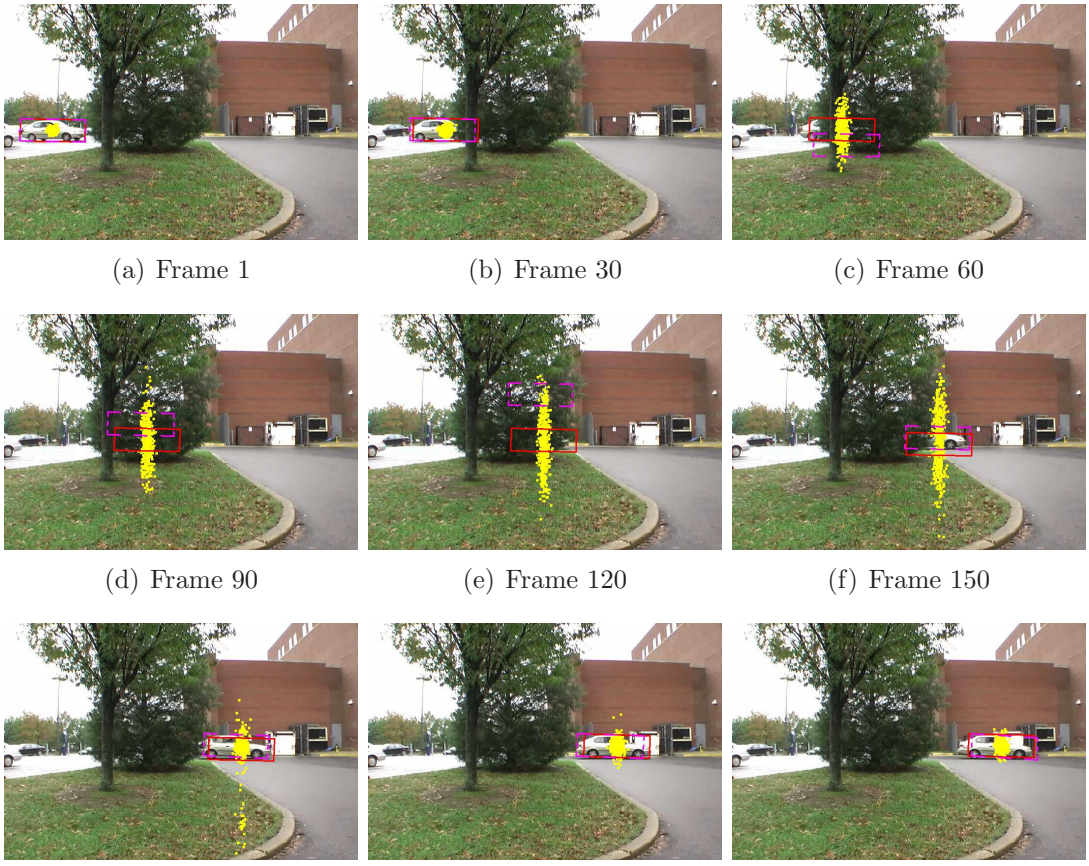
is set as 40%. Figure 5.10(a) shows the joint bearing estimate results whereas Fig. 5.10(b) and (c) show the acoustics-only and video-only tracking results, respectively. In Fig. 5.10(a), there is a small positive bias in the bearing estimates at the end due to the target's pose change. As can be seen in Fig. 5.9(h) and (i), the rear end of the vehicle is visible after the vehicle comes out of the occlusion. The online appearance model locks on the front of the vehicle, whose appearance was stored before the occlusion. Hence, the rear end of the vehicle is ignored, causing the bias. We see in Fig. 5.10(c) that the video-only tracker cannot handle this persistent occlusion without the help of the acoustics.

Note the time evolution of the estimate variances shown in Figs. 5.10(d) and (e) for the joint tracker and the acoustics-only tracker. When the video modality is unable to contribute, the variance of the estimate approaches acoustics-only results. When the video recovers, the estimate variance drops sharply. Figures 5.10(f) and (g) show the distribution of the vertical displacement parameter. When the occlusion is over at $t = 6$s, the video quickly resolves its ambiguity in the vertical displacement (Fig. 5.10(g)), whereas the variance of the vertical displacement in Fig. 5.10(f) increases linearly with time due to divergence. Figures 5.10(h) and (i) demonstrate the occlusion probability of the target.

Table 5.3: Simulation Parameters

Number of particles, N	1000
$\varphi(t)$ noise Σ_φ	diag $[0.02, 0.002, 0.002, 0.2, 2]$
θ noise $\sigma_{\theta,k}$	1°
Q noise $\sigma_{Q,k}$	0.05s^{-1}
ϕ noise $\sigma_{\phi,k}$	4°
Video Measurement noise σ_θ	$0.1,^\circ$
App. Model Template Size	15×15 (in pixels)
Beamformer batch period, τ	$\frac{1}{30}$s
Frame Size	720×480

Figure 5.11 illustrates the particle filter tracking results through dust. The video data, which was presented as a challenge for video tracking algorithms,



(a) Frame 1 (b) Frame 30 (c) Frame 60

(d) Frame 90 (e) Frame 120 (f) Frame 150

(g) Frame 180 (h) Frame 195 (i) Frame 210

Figure 5.9: Joint tracking of a vehicle that is occluded by a tree. The particle cloud at each frame represents the discrete support of the posterior distribution of the vehicle position in the image plane. Note that the particle spread during the occlusion increases along the vertical axis. This spread suddenly decreases, once occlusion is gone. The target is occluded in frames 40 to 180.

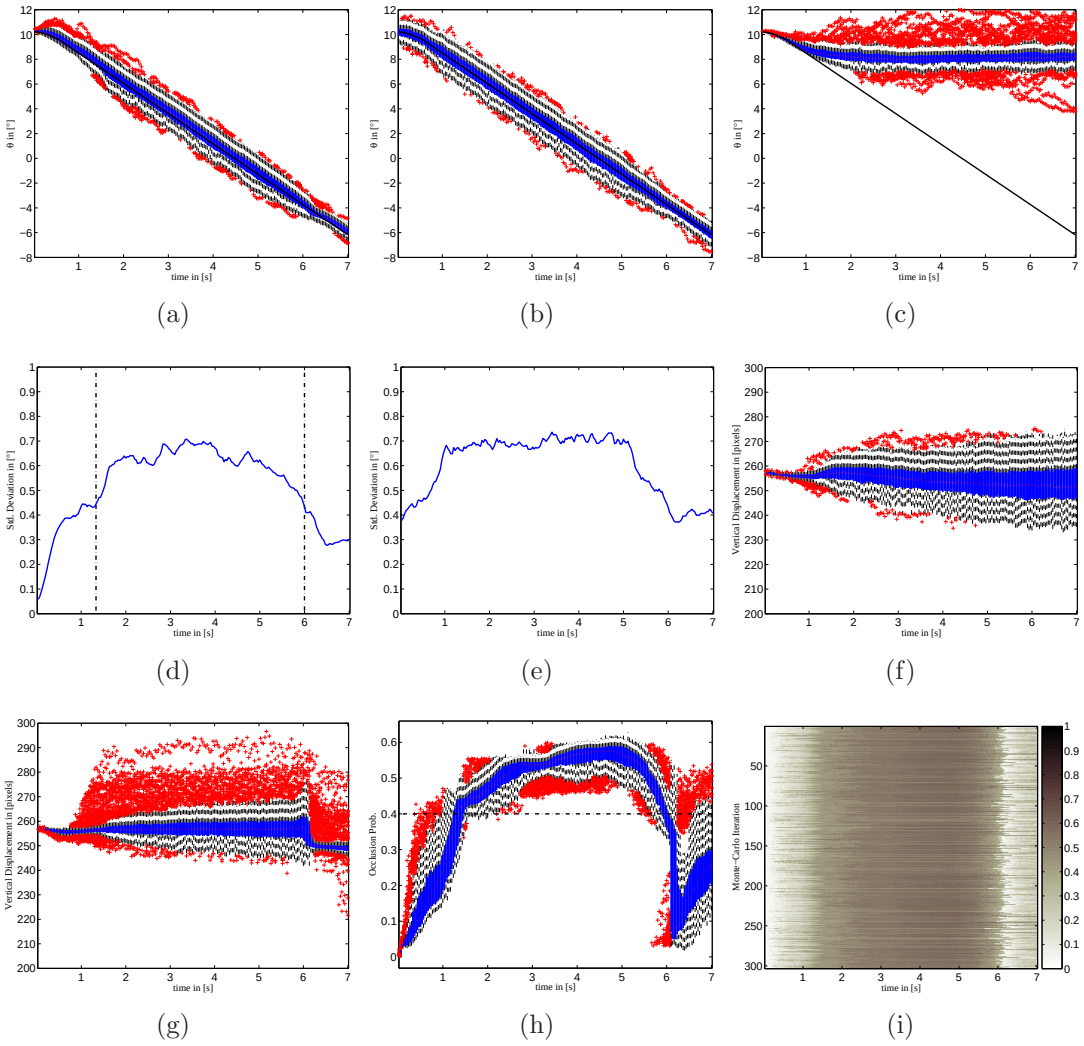


Figure 5.10: Results of 300 independent Monte-Carlo simulations of the experiment illustrated in Fig. 5.9. (a) MATLAB's boxplot of the estimated target DOA track with the joint tracker. The visual occlusion is between $t = 1$s and $t = 6$s. (b) The estimated DOA track using acoustics-only. (c) The estimated DOA track using video-only. (d-e) The time evolution of the estimate variances is shown for the joint filter and acoustics only, in their respective order. (f) Vertical displacement is unobservable during the visual occlusion. Hence, the video-only estimate variance increases linearly with time. (g) Note the variance of the estimates dramatically reduces once the target becomes unoccluded, demonstrating the recovery speed of the tracker. (h) The occluded percentage of pixels, corresponding to the MAP particle. (i) Probability of occlusion for the Monte-Carlo runs. The track recovery after occlusion is robust as illustrated by the Monte-Carlo runs.

was collected at the Aberdeen Proving Grounds in 2002. Due to the persistent occlusion, the video-only tracking algorithm cannot maintain target track after the gradual rise of the dust. Using the ground truth, we simulated acoustic data for the target motion. The joint acoustic video was employed to track the target. While the tracker fails to lock back on the target, due to considerable pose change, it is instructional to see that the joint tracking allows the horizontal displacement on the image plane to be tracked correctly.

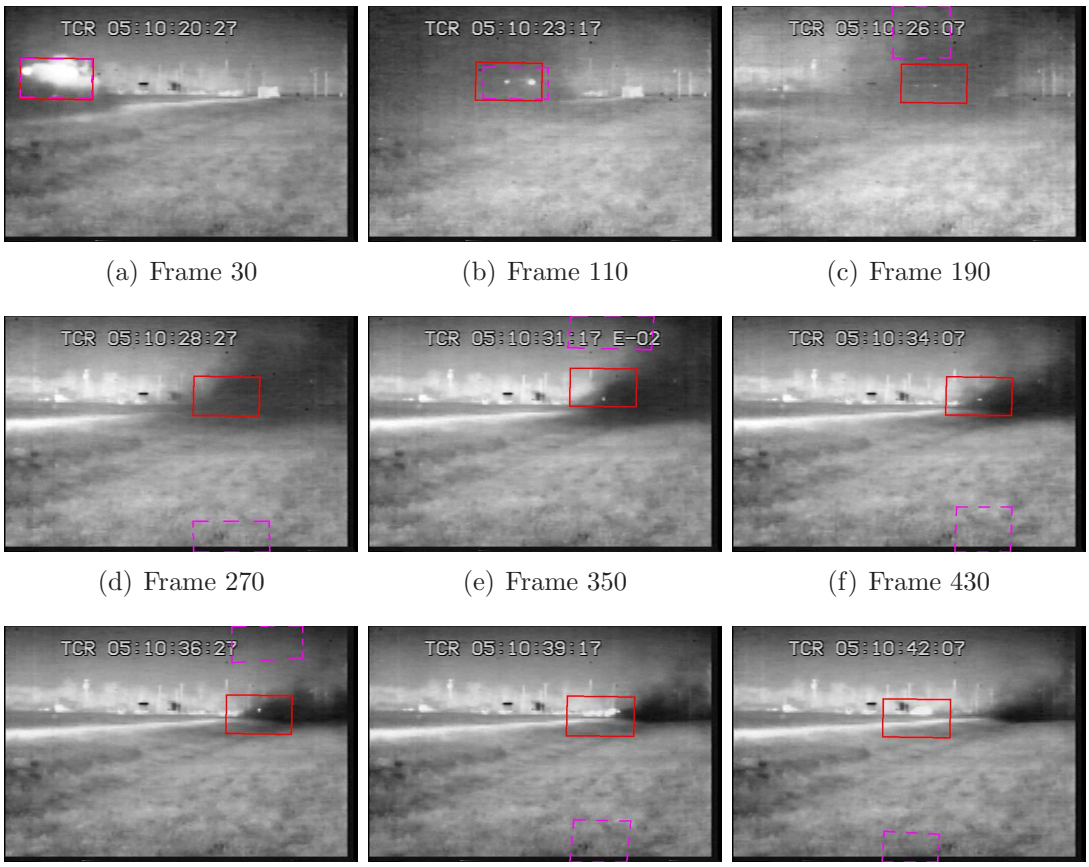
Finally, figure 5.13 shows additional results on a scenario where the target stops and reverses when it is occluded.

5.7.2 Time Delay Estimation

We performed a simulation with the time delay variable on a synthetically constructed multi-target data set. The simulation parameters are given in Table 5.4. The temporal tracks of two targets are shown in Fig. 5.14. The simulation parameters are given in Table 5.2. The results of the DOA and time delay estimation are shown in Fig. 5.15. The filter handles multiple targets independently by treating the data of the other target as clutter. Note the variance of the time delay estimates decreases as the targets get closer to the hybrid sensors. It is important to account for this time delay, because filter instability occurs due to the estimation biases when filtered with the unsynchronized data.

Table 5.4: Simulation Parameters

Number of particles, N	1000
θ noise $\sigma_{\theta,k}$	1°
Q noise $\sigma_{Q,k}$	0.05s^{-1}
ϕ noise $\sigma_{\phi,k}$	4°
Time delay d noise $\sigma_{d,k}$	0.2s
Acoustic Measurement noise σ_θ	1°
Video Measurement noise σ_θ	$0.1,^\circ$
Beamformer batch period, τ	$\frac{1}{30}$s

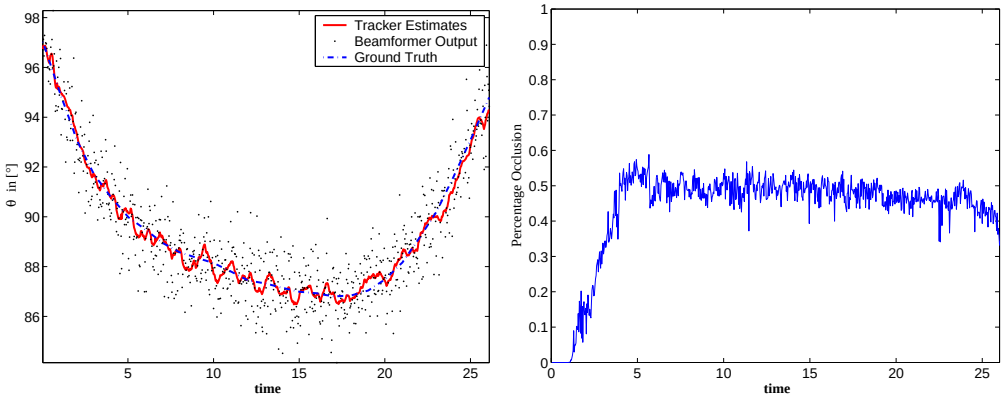


(a) Frame 30 (b) Frame 110 (c) Frame 190

(d) Frame 270 (e) Frame 350 (f) Frame 430

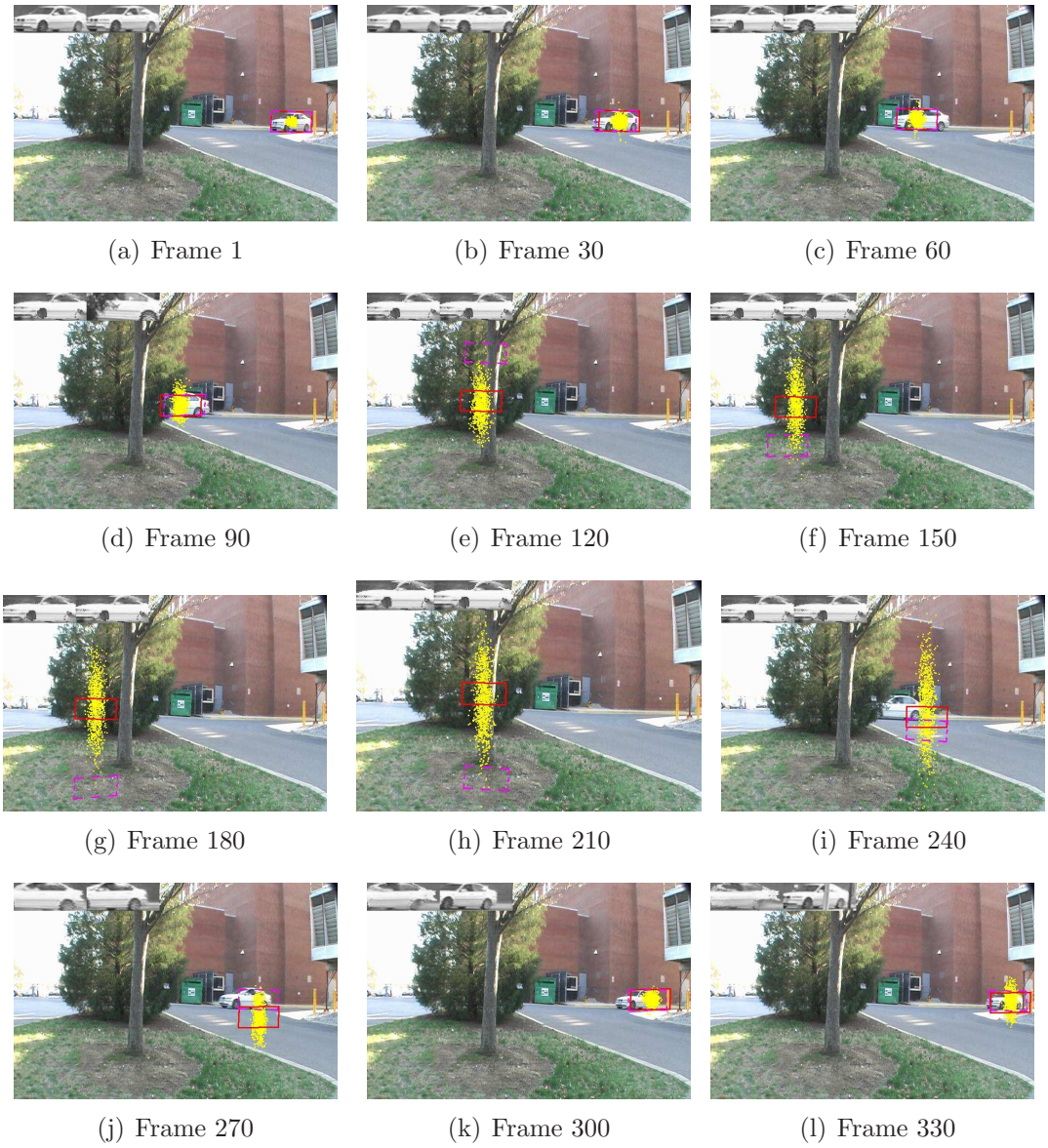
(g) Frame 510 (h) Frame 590 (i) Frame 670

Figure 5.11: Joint tracking results of a vehicle as it is occluded by its own dust. The solid box shows the posterior mean and the dashed box shows the posterior mode. The target is completely occluded for more than 20 seconds. Note that there is a dramatic change in target appearance due to the simultaneous change in target pose and scale. While the visual components of the state space do not immediately regain track after occlusion, the mean state estimates remain robust and on target, enabling the recovery and data association.



(a) (b)

Figure 5.12: (a) The video sensor is turned on at time $t = 72$s because the target is now in its field-of-view. The joint tracker estimates of the vehicle DOA track in Fig. 5.11 are compared with the ground truth. (b) Note the increase in the percentage of the occluded pixels. The gradual increase in the occlusion percentage after the first few seconds is attributed to the gradual rise of dust.



(a) Frame 1 (b) Frame 30 (c) Frame 60

(d) Frame 90 (e) Frame 120 (f) Frame 150

(g) Frame 180 (h) Frame 210 (i) Frame 240

(j) Frame 270 (k) Frame 300 (l) Frame 330

Figure 5.13: Tracking a target through occlusion, with the target reversing when it is occluded. The particle cloud is shown in yellow. The mean and map particle are in red and broken-magenta respectively.

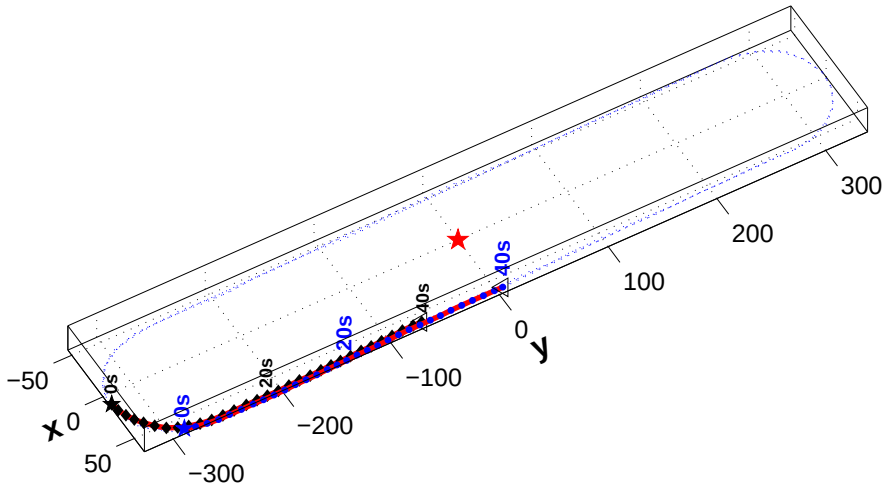
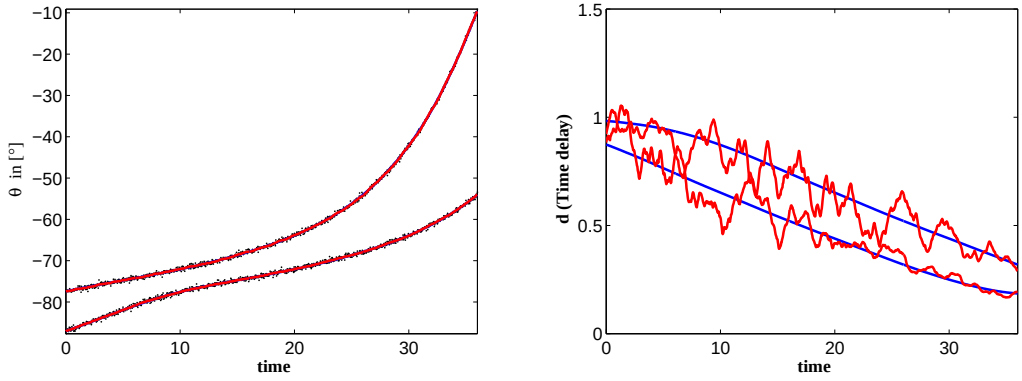


Figure 5.14: Two targets follow an oval track (dotted line). The hybrid node is situated at the origin.



(a) (b)

Figure 5.15: (a) Tracking of multiple targets with simulatenous estimation of time delay. (b) Estimated time delays. Note the reduction in the variance of the time delay estimates as the time delays get smaller.

Chapter 6

Computationally Efficient Particle Filtering

Particle filtering has been applied to a wide variety of problems such as tracking, navigation, detection [64, 107] and video based object recognition. This generality of particle filters comes from a sample (or particle) based approximation of the posterior density of the state vector. This allows the filter to handle both the non-linearity of the system as well as the non-Gaussian nature of noise processes. However, the resulting algorithm is computationally intensive and as a results, the need for efficient implementations of the algorithm, tuned specifically towards hardware or multi-processor based implementations.

Many methods for algorithmic and hardware implementations of particle filtering have been proposed in the literature. The authors of [16] identify resampling algorithms as the main computational step in the algorithm that is completely independent of the underlying application. They also propose new resampling algorithms that reduce the complexity of hardware implementations. Architectures for efficient distributed pipelined implementations using FPGAs have been proposed in [17]. A detailed analysis of the basic problem, addressing many hardware and software issues can also be found in [15, 4].

The resampling algorithms presented in the above references are modifications of the basic *systematic resampling* algorithm presented in [43], which by itself creates bottle-necks in a streamlined implementation. In [116], the authors propose a methodology to overcome this limitation by rederiving the basic theory, with an alternate resampling algorithm which is similar to the Monte Carlo Markov Chain (MCMC) Tracker for interacting targets in video presented in [74]. There have been a number of resampling schemes that have been proposed in the literature.

Liu and Chen [84] list and compare a number of such schemes. Of sufficient interest and relevance are the so called *Local Monte Carlo Methods* that are described in [84]. Specifically, this chapter analyzes the computational challenges in the implementations of particle filters, and *provides a general design methodology for particle filtering using pipelining and parallelization*; these are constructs that are commonly used in both hardware and multi-processor based systems.

Particle filtering involves three main modules: proposition, weight evaluation and resampling modules. Standard implementations of particle filtering typically use what is commonly known as *systematic resampling* (SR). Systematic resampling poses a significant challenge for pipelined implementations as it can only begin when all the weights are computed at the weight computation stage, and the cumulative sum of the weights is available. This means that any pipelined implementation would start the resampling only after all the weights are computed. This increases the latency of the whole implementations.

We present algorithmic and implementation schemes for particle filters for speeding up the basic computations, thereby making particle filtering-based solutions amenable to real time constraints. We demonstrate a computational methodology where the need for the knowledge of cumulative sum of weights is removed. This implies that, in contrast to traditional particle filtering implementation, the proposed approach does not suffer any bottlenecks in pipelining. Further, this allows us to speedup the filter and reduce its latency through pipelining and parallelization. We further demonstrate the performance of these implementations using a cluster of PCs. This allows us to achieve speedups that are linear in the number of cluster nodes.

The rest of the chapter is organized as follows. In section 6.1, we present the MCMC sampling theory and use it to propose a computational methodology in Section 6.2. Section 6.3 analyzes the implementations using the proposed methodology. Finally, in section 6.4, we demonstrate the performance of the proposed implementations for the problem of tracking in videos using a cluster of PCs.

6.1 Independent Metropolis Hastings Algorithm

In this section, we introduce Monte Carlo sampling techniques, discuss in detail the Metropolis Hastings Algorithms and its derivative, the Independent Metropolis Hastings Algorithms [113]. Further, we "redesign" the basic particle filtering algorithm using these techniques for sampling.

Particle filtering is a special case of more general MCMC based density sampling techniques, specifically suited for dynamical systems. The Metropolis Hastings Algorithm (MHA) [32, 62] is considered the most general MCMC based sampling. Popular samplers such as the Metropolis Sampler [95] or the Gibbs Sampler [53] are special cases of this algorithm.

The MHA and the particle filter both address the issue of generating samples from a distribution whose functional form is known (upto a normalizing factor) and is difficult to sample. In this section, we present a hybrid sampler that uses the sampling methodologies adopted in MCMC samplers (specifically, the MHA algorithm) for the problem of estimating posterior density functions. We later show that such a scheme is computationally more favorable than systematic resampling.

6.1.1 Metropolis Hastings Algorithm

We first present the general theory of MCMC sampling using the MHA algorithm and then state the conditions under which the general theory fits into the particle filtering algorithm presented before. The MHA generates samples from the desired density (say $p(x), x \in \mathcal{X}$) by generating samples from an easy to sample *proposal distribution*, say $q(x|y), x \in \mathcal{X}, y \in \mathcal{X}$. MHA produces a sequence of states $\{x^{(n)}, n \geq 0\}$, which by construction is Markovian in nature, through the following iterations.

---

1. Initialize the chain with an arbitrary value $x^{(0)} = x_0$. Here, x_0 could be user

specified.

2. Given $x^{(n)}, n \geq 0$, generate $\hat{x} \backsim g(\cdot|x^{(n)})$, where g is the sampling or proposal function.

3. Accept $\hat{x}$ with probability $\alpha(x^{(n)}, \hat{x})$ as defined below

$$\alpha(x^{(n)}, \hat{x}) = \min\left\{\frac{p(\hat{x})}{p(x^{(n)})}\frac{g(x^{(n)}|\hat{x})}{g(\hat{x}|x^{(n)})}, 1\right\} \tag{6.1}$$

That is, for a uniform random variable $u \backsim U[0,1]$

$$x^{(n+1)} = \begin{cases} \hat{x} & \text{if} \quad u \leq \alpha(x^{(n)}, \hat{x}) \\ x^{(n)} & \text{otherwise} \end{cases} \tag{6.2}$$

Under mild regularity conditions, it can be shown that the Markov Chain $\{x^{(n)}\}$ *as constructed by the MHA converges and has* $p(x)$ *as its invariant distribution, independent of the value* x_0 *chosen to initialize the chain [113].*

The MHA is used to generate a Monte Carlo Markov Chain whose invariant distribution is the distribution $p(x)$. However, there is an initial phase when the chain is said to be in a transient state, due to the effects of the initial value x_0 chosen. However, after sufficient samples, the effect of the starting value diminishes and can be ignored. The time during which the chain is in a transient state is referred to as *burn-in* period. This is usually dependent on both the desired function $p(x)$, the proposal function $q(x|y)$ and most importantly, on the initial state x_0. In most cases, an estimation of this burn-in period is very difficult. It is usually easier to make a conservative guess of what it could be. There are heuristics that estimate the number of burn-in samples (say N_b). Samples that are in the burn-in period are discarded.

6.1.2 Independent Metropolis Hastings Algorithm

The Independent Metropolis Hastings Algorithm (IMHA) is a special case of the general MHA where the proposal function $q(x|y)$ is set as $q(x)$. This makes the proposal function independent of the previously accepted sample in the chain. This would mean that the acceptance probability (6.1) $\alpha(x^{(n)}, \hat{x})$ of a proposal $\hat{x} \in \mathcal{X}$ with the chain at $x^{(n)} \in \mathcal{X}$,

$$\alpha(x^{(n)}, \hat{x}) = \min\left\{\frac{p(x^{(n)})}{g(x^{(n)})}\frac{g(\hat{x})}{p(\hat{x})}, 1\right\} \tag{6.3}$$

The IMH algorithm has strong convergence properties. Under mild regularity conditions, it has been shown to converge at a uniform rate independent of the value x_0 used to initialize the chain. A study of such convergence properties can be found in [94, 113].

Both IMHA and SISR are algorithms designed to generate samples according to a probability density function, with the SISR suited specifically to the sequential nature of dynamical systems. In this regard, the key difference between the IMHA and the SISR algorithm lies in the fact that the SISR algorithm requires the knowledge of cumulative sum of weights (the term $\sum_{j=1}^{N} \tilde{w}_t^{(j)}$ in (2.19)). This is important as the cumulative sum can only be computed when the weights corresponding to the whole particle set is known. Hence, SR can only begin after all particles are generated and their weights are computed. In contrast, the IMHA poses no such bottlenecks. In the next section, we exploit this property to design a filter that does not suffer from the bottle-necks introduced by SR.

6.2 Proposed Methodology

The bottlenecks introduced by the SR technique can be overcome by using IMHA for resampling. However, there are some basic issues that need to be resolved before we achieve this. To begin with, the generation of particles using importance

sampling works differently for the two algorithms. Particle filtering allows for the importance function to be defined locally for each particle. Mathematically, the i^{th} particle at time t is generated from an importance function, represented as $g(x_t|x_{t-1}^{(i)}y_t)$, parametrized by $x_{t-1}^{(i)}$. This poses a problem in the application of IMHA to estimate the posterior, because the concept of importance functions associated with each particle does not extend to IMHA. In contrast, the MHA algorithm requires the importance function to depend functionally only on the last accepted sample in the chain, and in the case of the IMHA, the importance function remains the same.

Given a set of unweighted samples $\{x_{t-1}^{(i)}, i = 1, \ldots\}$ sampled from the posterior density $p(x_{t-1}|y_{1:t-1})$ at time $t-1$, we can approximate the posterior by

$$p(x_{t-1}|y_{1:t-1}) \approx \frac{1}{N}\sum_{i=1}^{N} \delta_{x_{t-1}}(x_{t-1}^{(i)}) \tag{6.4}$$

where $\delta_{x_{t-1}}(\cdot)$ is the Dirac Delta function on x_{t-1}. Using (2.12) and (6.4), we can approximate the posterior at time t,

$$p(x_t|y_{1:t}) \approx \frac{p(y_t|x_t)}{p(y_t|y_{1:t-1})}\frac{1}{N}\sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)}) \tag{6.5}$$

Sampling from this density can be performed using MHA or IMHA. The issue of choice of importance function now arises. The importance function typically reflects and exploits the knowledge of application domain or could be a clever approximation to the posterior. For this reason, we would like to reuse the importance function corresponding to the underlying model.

Keeping this in mind, we propose a new importance function of the form,

$$g'(x_t|y_t) = \sum_{i=1}^{N} \frac{1}{N} g(x_t|x_{t-1}^{i}y_t) \tag{6.6}$$

Note that $g'(x_t|y_{1:t})$ qualifies to be an importance function for use in IMHA, given

its dependence on only one state variable. To sample from $g'(x_t|y_t)$, we need to first sample $I \backsim U[1, 2, \ldots, N]$, and then sample from $g(\cdot|x_{t-1}^I y_t)$. The sampling of I can be done deterministically given the ease of sampling from uniform densities over finite discrete spaces. *Finally, although the new importance function is functionally different from the one used in the SISR algorithm, the generated particles will be identical.*

The overall algorithm proceeds similar to IMHA. We first propose particles using the new importance function $g'(x_t|y_{1:t})$. The acceptance probability now takes the form

$$\alpha(x_t, \hat{x}) = \min\left\{\frac{w'(\hat{x})}{w'(x_t)}, 1\right\} \tag{6.7}$$

$$w'(x_t) = p(y_t|x_t)\frac{\sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)})}{\sum_{i=1}^{N} g(x_t|x_{t-1}^{(i)} y_t)} \tag{6.8}$$

Further, if the choice of the importance function were the same as the state transition model, i.e, $g(x_t|x_{t-1}y_t) = p(x_t|x_{t-1})$, then the acceptance probability becomes a ratio of likelihoods,

$$\alpha(x_t^{(n)}, \hat{x}) = \min\left\{\frac{p(y_t|\hat{x})}{p(y_t|x_t^{(n)})}, 1\right\} \tag{6.9}$$

We can now avoid the systematic resampling of traditional particle filtering algorithms. The intuition is that we will use IMHA to generate unweighted particle set/stream from the desired posterior.

As before, we have an unweighted particle set S_{t-1}, that contains particles approximating the posterior at time $t-1$, $\pi_{t-1}(x_{t-1})$. We aim to estimate an approximation to the posterior at time t. As before, the algorithm is initialized with S_0 containing samples from the prior $p(x_0)$. The main steps are stated below:

- **Importance Sampling (step 1):** Generate $N+N_b$ indices $J(i), i = 1, \ldots, N+N_b$ uniformly from the set $\{1, 2, 3, \ldots, N\}$, where N_b is an estimate of the *burn*

in period and N is the number of particles required. between $1..N$ with uniform density.

- **Importance Sampling (step 2):** From the particle set $S_{t-1} = \{x_{t-1}^{(i)}, i = 1, \ldots, N\}$ at time $t-1$, propose $N + N_b$ particles to form the set $\hat{S}_t = \{\hat{x}_t^{(i)}, i = 1, \ldots, N + N_b\}$ using the rule:

$$\hat{x}_t^{(i)} \backsim g(\cdot | x_{t-1}^{J(i)} y_t) \tag{6.10}$$

- **Compute Importance Weights:** For each particle in $\hat{S}_t$, evaluate the importance weights $w'^{(i)}_t$, for each i using (6.8).

- **Inference:** Estimate the expected value of functions of interest. Compute

$$\hat{I}_t(f_t) = \frac{\sum_{i=1}^{N+N_b} f(x_t^{(i)}) w'^{(i)}_t}{\sum_{i=1}^{N+N_b} w'^{(i)}_t} \tag{6.11}$$

Note that samples discarded during burn-in can still be used in the computation of (6.11) as the unnormalized particle set $\{x_t^{(i)}, w'^{(i)}_t, i = 1, \ldots, N + N_b\}$ *is still properly weighted (when normalized) [130].*

- **MCMC Sampler:** Use the IMH sampler to parse through the set $\hat{S}_t$, to generate a new unweighted set of particles using the following steps.

  1. Initialize the chain with $x_t^{(1)} = \hat{x}_t^{(1)}$ the first particle proposed.
  2. for $i = 2, \ldots, N + N_b$,

$$x_t^{(i)} = \begin{cases} \hat{x}_t^{(i)}, & \text{with prob.} \quad \alpha(x_t^{(i-1)}, \hat{x}_t^{(i)}) \\ x_t^{(i-1)}, & \text{with prob.} \quad 1 - \alpha(x_t^{(i-1)}, \hat{x}_t^{(i)}) \end{cases} \tag{6.12}$$

  where $\alpha(\cdot, \cdot)$ is the acceptance probability as defined in (6.1).

Discarding the first N_b samples for burn in, the remaining N samples form $S_t = \{x_t^{(i)}, i = N_b + 1, \ldots, N\}$, the approximation of $p(x_t|y_{1:t})$.

We can now compare the algorithm given above with the classical SISR discussed in Section 2.2.2. Note that the SISR algorithm involves a weight normalization step (equation (2.19)). However, the proposed algorithm works with ratios of unnormalized weights and requires no such normalization. This allows for the following advantages in the proposed methodology:

- *The IMH sampler works with ratios of importance weights. This obviates the need for knowledge of normalized importance weights, as we can work with unnormalized weights. This allows the IMH sampler to start parsing through the particles as they are generated, and not wait for the entire particle set to be generated and the importance weights computed.*

- *In contrast, in SISR, the resampling can begin only when all particles are generated and the cumulative sum or normalized weights are known.*

The ability to resample particles as they are generated allows for faster implementations. This is analyzed further in section 6.3.

6.2.1 Drawbacks of the proposed Framework

The proposed framework overcomes the drawbacks of the SISR algorithm by adopting an MCMC sampling strategy as opposed to the traditional SR technique. However, the new framework does introduce extra computations that add to increased overall complexity. We discuss these drawbacks, and an alternate formulation that can circumvent this issue.

Consider the expression for weight computation, given in (6.8). The expression involves computing the summations $\sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)})$ and $\sum_{i=1}^{N} g(x_t|x_{t-1}^{(i)}y_t)$, which require additional computation time. The computation of both terms does not

present a severe bottleneck, as it can be easily pipelined. Further, when the proposal density matches the state transition model, the terms cancel each other out.

Nonetheless, it is possible to circumvent this problem using the auxiliary particle filtering paradigm [42, 105].

6.2.2 Auxiliary Particle Filters

Auxiliary particle filtering refers to techniques that extend the state space of the problem to include a particle index. Consider the new state space $\{x_t, k\}$, where $k \in [1, \ldots, N]$ denotes the particle index. The posterior $p(x_t k|y_{1:t})$ is defined as

$$p(x_t k|y_{1:t}) \propto p(y_t|x_t)p(x_t|x_{t-1}^k) \tag{6.13}$$

Marginalizing (6.13) over the state k gives the expression in (6.5) for $p(x_t|y_{1:t})$.

Let us further assume that we sample the joint space using a proposal $g(x_t k|x_{t-1}y_t)$, i.e, $(x_t^{(i)}, k^{(i)}) \sim g(\cdot|x_{t-1}y_t)$. The unnormalized weights can be constructed as

$$w_t^{(i)} = \frac{p(y_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{k^{(i)}})}{g(x_t^{(i)}, k^{(i)})|x_{t-1}^{(i)}y_t)} \tag{6.14}$$

As before, we can resample using an MCMC chain, and the expression for acceptance probability remains the ratio of unnormalized weights as given in (6.1). At the inference step, we first marginalize across the particle index state k. However, it is easy to see that the marginalization is identical to discarding the particle index information at each particle, given the nature of the particle-based representation of the underlying density. In a nutshell, the use of auxiliary variable allows us to completely avoid the summation of (6.8) and the associated computational cost.

Finally, there exist many choices for the proposal density in the extended state space. A discussion on this can be found in [105].

6.3 Implementation based on Proposed Methodology

In this section, we present approaches for implementing the theory presented in section 6.1. We assume that the basic computational blocks for importance sampling, computation of importance weight and parsing of particles as per the IMH algorithm are available. We use these blocks to propose three implementations: a sequential implementation and two parallel implementations.

6.3.1 Sequential Implementation

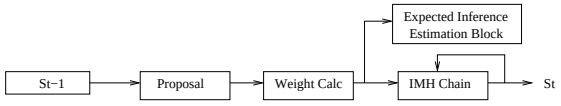


Figure 6.1: Sequential Implementation

Figure 6.1 illustrates a straight-forward implementation of the proposed algorithm. It consists of the following blocks.

Proposal Block: The proposal block takes S_{t-1}, the particles from the previous time step and proposes new particles $x_t^{(i)}$ (one particle at a time) by sampling the proposal function. For the IMHA-based algorithm, this amounts to generating a uniform number $J(i) \sim U[1, 2, \ldots, N]$ to randomly pick one particle from S_{t-1}, say $\{x_{t-1}^{J(i)}\}$. The particle $x_t^{(i)}$ is obtained from sampling $g(x_t|x_{t-1}^{J(i)}y_t)$. We assume that this blocks proposes particle one at a time. When we use the auxiliary variable framework, this involves sampling both the state $x_t^{(i)}$ and the associated particle index state $k^{(i)}$ from a proposal function $g(x_t k|x_{t-1}y_t)$.

Weight Calculator: This block is an implementation of (6.8) (or (6.14) when we use auxiliary variables).

IMH Chain: This block is an implementation of (6.7) in which the acceptance probability α is calculated for the new particle and the previously accepted particle. Further, an uniform random-number $u \sim U[0, 1]$ is generated and if it

is smaller than α then the new particle is retained in S_t, else the last accepted particle in the chain is replicated once more.

Inference Estimation Block: This block estimates the inference function (equation 2.9). The computation can be performed in parallel with the IMH chain, and has no effect on the overall computation.

The characteristics of this basic implementation are as follows.

- **Sequential Processing of Particles:** Each block in the implementation processes one particle at a time. So, to process Q particles each block needs to run Q times. Note that, if we need to generate N particles to represent the posterior density, then we will have to iterate $N + N_b$ times where N_b is the burn-in period. The last N particles in the IMH chain is the sample set S_t.

- **Pipelining:** By pipelining the blocks, processing in each block can be made to overlap in time, leading to an overall increase in the throughput of the system.

- **Computation Time:** We now estimate the time required to process $Q = N + N_b$ particles under this implementation. Let us suppose that the target application is such that the proposal block can generate one particle every T_p time units. The weight computation block generates the weight of a particle in T_w time units, and the IMH chain process particles once in every T_d time units. Further, we assume that the overall time required to process is not constrained by the inference block (and therefore ignored in this analysis). Under this setting, we can compute the total time required to process Q particles.

The implementation in Figure 6.1 will take $T_p + T_w + T_d$ time units to produce the first particle $x_t^{(1)}$). Thereafter, it will be able to produce one particle every $\max(T_d, T_p, T_w)$ time units. The total latency for generating $N_b + N$ particles would be $(N_b + N - 1)\max(T_d, T_p, T_w) + T_p + T_w + T_d$ time units. This basic sequential implementation can be made faster by replicating the proposal, weight

computation and the IMH chain blocks. In order to exploit the parallelism in processing of particles, we present a refinement of the sequential implementation.

6.3.2 Parallel Implementation: Single Chain

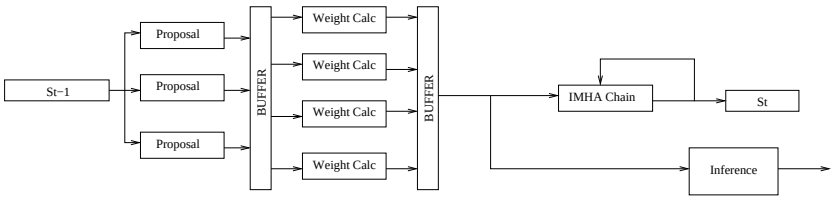


Figure 6.2: Parallel Implementation with a single IMH Chain.

Figure 6.2 illustrates the parallel implementation of the proposed algorithm. We still retain a single IMH Chain, though the proposal and the weight computation blocks are replicated. Having multiple IMH chains introduces additional issues involving burn-in in each chain. For this reason, we first restrict ourselves to single chain implementations. We relax this restriction later in Section 6.3.3. Let the number of proposal blocks be R_p and the number of weight computation blocks be R_w. We would like to compute the total time required to process Q particles as a function of R_p and R_w (and the latency of the blocks T_p, T_w and T_d). Further, we would like to choose specific values of R_p and R_w to achieve the smallest total processing time.

The total computational time is determined by bottlenecks in processing created due to differing rates of processing of particles at each stage. The rate at which the proposal blocks process particles is R_p/T_p, the weight computation blocks at R_w/T_w and the IMHA blocks at $1/T_d$. The total computational time is predominantly dependent on which of the three rates is the smallest.

6.3.2.1 Case A: $R_p/T_p \leq R_w/T_w \leq 1/T_d$.

In this scenario, the proposal blocks have the smallest rate of processing, followed by the weight computation blocks. Suppose we need to process Q particles,

then the proposal blocks by themselves will need $(Q/R_p)T_p$ time units to process all particles. The weight computation and IMHA processing happen in parallel. Given the quicker processing rate at both weight computation and IMHA, by the time the last set of R_p particles is processed at the proposal blocks, all earlier particles have already been processed through the weight computation blocks. The amount of time required to process the last set of R_p particles at the weight computation blocks and the IMHA block is $R_pT_w/R_w + R_wT_d$. Allowing R_p and R_w to take values over the real line (and not just positive integers) the total time for processing τ_A is,

$$\tau_A(R_p, R_w) = \frac{Q}{R_p}T_p + \frac{R_p}{R_w}T_w + R_wT_d \tag{6.15}$$

We are now interested in computing the values of R_p and R_w that minimize τ_A, keeping in mind that such solutions must satisfy the assumptions of Case A. To begin with, we note that both R_p and R_w take positive values. This allows a natural change of coordinate frames of the form,

$$\begin{aligned} \tilde{R}_p &= \log(R_p) \\ \tilde{R}_w &= \log(R_w) \end{aligned} \tag{6.16}$$

In terms of $\tilde{R}_p$ and $\tilde{R}_w$, the expression for τ_A can be written as,

$$\tau_A(\tilde{R}_p, \tilde{R}_w) = QT_P^{-\tilde{R}_p} + T_we^{\tilde{R}_p - \tilde{R_w}} + T_de^{\tilde{R}_w} \tag{6.17}$$

The constraints for the minimization come from the assumptions made on the ordering of the rates in Case A.

$$\begin{aligned} \tilde{R}_p - \tilde{R}_w - \log\left(\tfrac{T_p}{T_w}\right) &\leq 0 \\ \tilde{R}_w - \log\left(\tfrac{T_w}{T_d}\right) &\leq 0 \end{aligned} \tag{6.18}$$

Finally, R_p and R_w are naturally bounded by the value of Q. This leads a convex

optimization problem with inequality constraints stated as,

$$\min_{\tilde{R}_p,\tilde{R}_w} \tau_A(\tilde{R}_p,\tilde{R}_w) = QT_p e^{-\tilde{R}_p} + T_w e^{\tilde{R}_p-\tilde{R_w}} + T_d e^{\tilde{R}_w} \tag{6.19}$$

$$\begin{array}{cc} \tilde{R}_p - \tilde{R}_w - \log\left(\frac{T_p}{T_w}\right) \leq 0 & \tilde{R}_p - \log Q \leq 0 \\ \tilde{R}_w - \log\left(\frac{T_w}{T_d}\right) \leq 0 & \tilde{R}_w - \log Q \leq 0 \end{array} \tag{6.20}$$

We now note that the expression for τ_A is convex in both $\tilde{R}_p$ and $\tilde{R}_w$. Further, the inequality constraint is also convex in $\tilde{R}_p$ and $\tilde{R}_w$. One can use a host of techniques [18] designed specifically for convex optimization.

6.3.2.2 Case B: $R_w/T_w \leq R_p/T_p, R_w/T_w \leq 1/T_d$.

Using a line of reasoning identical to Case A, we can derive an expression for the amount of time τ_B needed to process Q particles, as a function of R_p and R_w.

$$\tau_B(R_p,R_w) = \frac{R_w}{R_p}T_p + \frac{Q}{R_w}T_w + R_w T_d \tag{6.21}$$

We note that a value of R_p greater than R_w is impractical leading to a constraint on R_p of the form $R_p \leq R_w$. As before, we can recast the set of equations in terms of $\tilde{R}_p$ and $\tilde{R}_w$ (as defined in (6.16)) to get the cost and constraint equations.

$$\min_{\tilde{R}_p,\tilde{R}_w} \tau_B(\tilde{R}_p,\tilde{R}_w) = T_p e^{\tilde{R}_w-\tilde{R}_p} + QT_w e^{-\tilde{R}_w} + T_d e^{\tilde{R}_w} \tag{6.22}$$

$$\begin{array}{cc} \tilde{R}_w - \tilde{R}_p - \log\left(\frac{T_w}{T_p}\right) \leq 0 & \tilde{R}_w - \log Q \leq 0 \\ \tilde{R}_w - \log\left(\frac{T_w}{T_d}\right) \leq 0 & \tilde{R}_p - \tilde{R}_w \leq 0 \end{array} \tag{6.23}$$

Both the cost function and the inequality constraints are convex in $\tilde{R}_p$ and $\tilde{R}_w$.

6.3.2.3 Case C: $R_p/T_p \leq 1/T_d \leq R_w/T_w$.

In Case C, the main bottleneck is in the proposal block, followed by the IMH chain. Accordingly, the total time τ_C for processing of Q particles is

$$\tau_C(R_p, R_w) = \frac{Q}{R_p} T_p + T_w + R_p T_d \tag{6.24}$$

Using the transformation of variables in (6.16), we can write down expressions for both the cost τ_C and the constraints.

$$\min_{\tilde{R}_p, \tilde{R}_w} \tau_C(\tilde{R}_p, \tilde{R}_w) = QT_p e^{-\tilde{R}_p} + T_w e^{\tilde{R}_p - \tilde{R}_w} + T_d e^{\tilde{R}_w} \tag{6.25}$$

$$\begin{aligned} \tilde{R}_p - \tilde{R}_w - \log\left(\tfrac{T_p}{T_w}\right) \leq 0 \quad & \tilde{R}_p - \log Q \leq 0 \\ \tilde{R}_w - \log\left(\tfrac{T_w}{T_d}\right) \leq 0 \quad & \tilde{R}_w - \log Q \leq 0 \end{aligned} \tag{6.26}$$

As before, both the cost and the inequality constraints are convex over $\tilde{R}_p$ and $\tilde{R}_w$.

6.3.2.4 Case D: $1/T_d = \min(R_p/T_p, R_w/T_w, 1/T_d)$.

The final scenario is when the main bottleneck is at the IMH chain. The expression for total time τ_D is given as,

$$\tau_D(R_p, R_w) = T_p + T_w + QT_d \tag{6.27}$$

τ_D is not dependent on the choice of R_p and R_w. So the whole feasibility set forms the solution set when we optimize for minimum processing time. For completeness, we again formulate it as a convex program with the following cost and constraints.

$$\min_{\tilde{R}_p, \tilde{R}_w} \tau_D(\tilde{R}_p, \tilde{R}_w) = T_p + T_w + QT_d \tag{6.28}$$

Case	Rate Ordering	Cost	Constraint
A	$R_p/T_p \leq R_w/T_w \leq 1/T_d$	$QT_p e^{-\tilde{R}_p} + T_w e^{\tilde{R}_p - \tilde{R_w}} + T_d e^{\tilde{R}_w}$	$\tilde{R}_p - \tilde{R}_w - \log\left(\frac{T_p}{T_w}\right) \leq 0 \quad \tilde{R}_p - \log Q \leq 0$ $\tilde{R}_w - \log\left(\frac{T_w}{T_d}\right) \leq 0 \quad \tilde{R}_w - \log Q \leq 0$
B	$R_w/T_w = \min(R_p/T_p, R_w/T_w, 1/T_d)$	$T_p e^{\tilde{R}_w - \tilde{R}_p} + QT_w e^{-\tilde{R}_w} + T_d e^{\tilde{R}_w}$	$\tilde{R}_w - \tilde{R}_p - \log\left(\frac{T_w}{T_p}\right) \leq 0 \quad \tilde{R}_p - \log Q \leq 0$ $\tilde{R}_w - \log\left(\frac{T_w}{T_d}\right) \leq 0 \quad \tilde{R}_w - \tilde{R}_p \leq 0$
C	$R_p/T_p \leq 1/T_d \leq R_w/T_w$	$QT_p e^{-\tilde{R}_p} + T_w + T_d e^{\tilde{R}_p}$	$\tilde{R}_p - \log\left(\frac{T_p}{T_d}\right) \leq 0 \quad \tilde{R}_p - \log Q \leq 0$ $-\tilde{R}_w + \log\left(\frac{T_w}{T_d}\right) \leq 0 \quad \tilde{R}_w - \log Q \leq 0$
D	$1/T_d = \min(R_p/T_p, R_w/T_w, 1/T_d)$	$T_p + T_w + QT_d$	$-\tilde{R}_p + \log\left(\frac{T_p}{T_d}\right) \leq 0 \quad \tilde{R}_p - \log Q \leq 0$ $-\tilde{R}_w + \log\left(\frac{T_w}{T_d}\right) \leq 0 \quad \tilde{R}_w - \log Q \leq 0$

Table 6.1: Expressions for total time taken to process Q particles for bottlenecks at various stages in the pipeline.

$$\begin{aligned} -\tilde{R}_p + \log\left(\frac{T_p}{T_d}\right) \leq 0 \quad \tilde{R}_p - \log Q \leq 0 \\ -\tilde{R}_w + \log\left(\frac{T_w}{T_d}\right) \leq 0 \quad \tilde{R}_w - \log Q \leq 0 \end{aligned} \tag{6.29}$$

As stated above, in Case D all points in the feasible set form the solution set.

Depending on the exact location of the bottle neck, it is possible to have upto 6 different scenarios. However, some of these scenarios collapse to identical expressions for the total cost leading to the four cases A through D discussed above. The expressions for the cost and the associated constraints are summarized in Table 6.1. We note that each case results in a convex cost function and convex inequality constraints. This allows us to design an algorithm for determining the global minima for total computation time for processing Q particles given values of T_p, T_w and T_d.

---

1. Given values of T_d and T_w, formulate FOUR convex programs associated with the four cases illustrated in Table 6.1.

2. Solve each convex program to obtain minimum times $\tau_{i,\min}, i \in \{A, B, C, D\}$ and associated values of $\tilde{R}_p$ and $\tilde{R}_w$.

3. Choose the configuration that gives the least total processing time.

The above algorithm allows us to obtain design specifications with minimum processing time given values of T_p, T_w, T_d and Q. Note that the basic computation tools used are optimization techniques for convex programs. Convex optimization is a well studied problem, and there are techniques that solve convex programs very efficiently and reliably [18]. Further, convex programs have very desirable properties with respect to local minima. All local minima are also global minima, and further the set of all local (global) minima form a convex set themselves. Finally, we note that analytic solutions to the convex program are highly dependent on the individual values of Q, T_p , T_w and T_d.

It is possible that the four convex program may not have unique solutions. Ambiguity in choice of R_p and R_w over the solution set can be resolved, if we have additional considerations such as resource or energy constraints. *It is noted that the set of all solutions to a convex program is also convex [18]. This property could be effectively used to design alternate cost functions to resolve the ambiguity in the choice of R_p and R_w.*

6.3.3 Parallel Implementation: Multiple Chains

Figure 6.3 shows a parallel implementation of the proposed algorithm with multiple IMH chains. The implementation basically replicates the structure proposed in Figure 6.2 multiple times. This implementation gives speedup proportional to the number of IMH Chains.

Let P be the number of IMH chains. Under this implementation, to generate a set S_t with N particles, each chain would need to generate only N/P particles, excluding that required for burn in, leading to a total of $N_b + N/P$ particles at each IMH chain. Hence, the time required for obtaining an N-particle set is equal to the time required to process $N_b + N/P$ particles in the implementation as per Figure 6.2. With this, we can easily compute the total time required to generate S_t for different scenarios using the same analysis as before, and restricting the total number of particles per IMH chain to $N_b + N/P$.

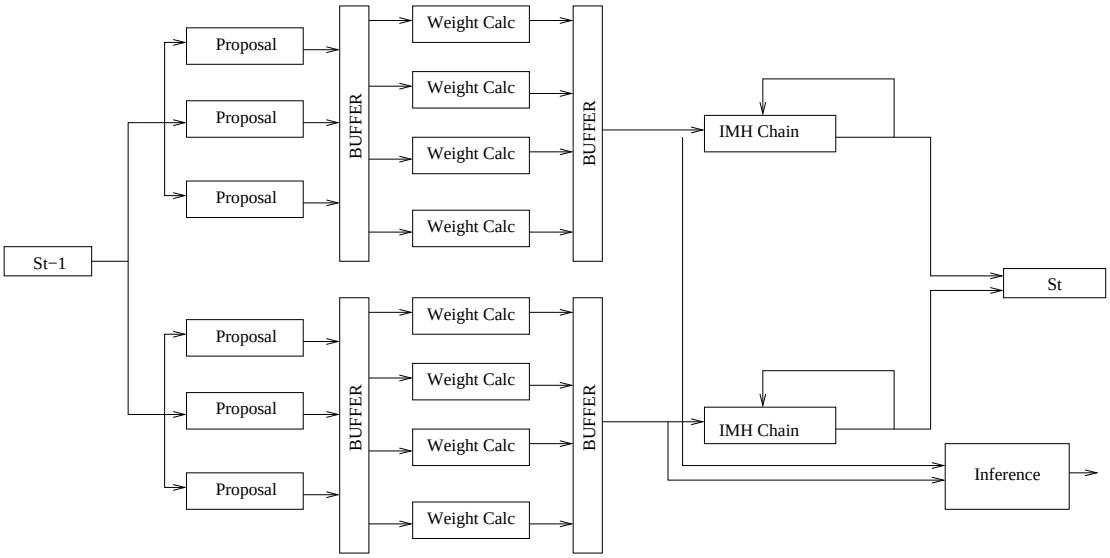


Figure 6.3: A Parallel Implementation with Multiple IMH Chains.

6.4 Experimental Verification

The design methodologies proposed were verified for two applications: a synthetic example originally discussed in [56] and for the problem of visual tracking . The testbed was the UMIACS Red/Blue cluster. The Red cluster consists of 16 PII (400 MHz) PCs running Redhat 7.3, with each PC having a RAM of 1GB. The Blue cluster consists of 12 PIII (550Mhz). We used MPICH [57] [58], an implementation of the Message Passing Interface (MPI) for communication between threads.

We chose to implement over a multi-processor cluster framework as the underlying theory applies both to hardware based design as well as to clusters. In general, MPI has large overheads; however, such overheads are common and identical to both SISR as well as the MCMC based schemes. The conclusions from our experimental observations still remain the same.

Further, as mentioned earlier, computation of the burn-in period is a hard problem by itself. However, in sequential estimation, the proposal density is in

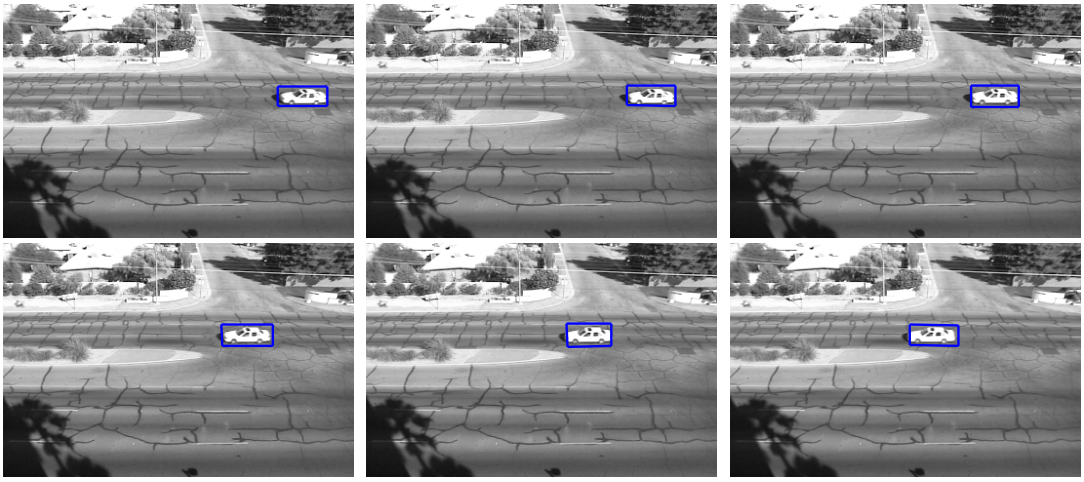


Figure 6.4: Frames 1,4,8,12,16,20 of the tested set. The output of the tracker is inlaid on top

general a good guess of the posterior. In such cases, the adverse effects of burn-in period are reduced. For the experiments below, we set $N_b = 0$.

6.4.1 Visual Tracking

We implemented the particle filter based online tracking algorithm presented in [141] using the Red Cluster. We discuss the finer details of the filter and its implementation below.

6.4.1.1 Model Details

We first summarize the tracking algorithm, detailing its computational aspects. A typical tracking example in shown in Figure 6.4. The models defining the dynamical system is described below.

- **State Space:** The state x_t is a 6-dimensional vector ($\mathcal{X} = \mathbb{R}^6$) defining affine deformations of a rectangular template.

- **State Transition Model:** A simple random walk model with Gaussian noise is used to model the state transition.

$$x_t = x_{t-1} + n_t,\ n_t \sim N(0, \Sigma_n) \tag{6.30}$$

- **Observation Model:** The frame of the video at time t forms the observation. The likelihood model $p(y_t|x_t)$ involves comparing the appearance model A_t suitably deformed by the state x_t with the observation y_t. The appearance model employs a mixtures of Gaussians with three mixtures to model the appearance. Each value of x_t defines a patch (parallelogram shaped) on the image y_t. Let $z_t = T(y_t; x_t)$ be the patch defined by x_t over y_t. Then,

$$p(y_t|x_t) = p(y_t|x_t A_t) = p(z_t|A_t) \tag{6.31}$$

- **Proposal Density:** The algorithm uses the proposal density to be the same as the state transition model.

6.4.1.2 Implementation Details

An estimate of T_p, T_w and T_d was first obtained by running each block over a single PC many times over and averaging the individual runs.

- $T_p = 8\mu s, T_w = 1.1ms, T_d = 1\mu s$

With this, we can see that the main computational bottleneck for this particular application is the evaluation of weights. The weight computation has an unusually large latency, primarily because it involves retrieval from the memory. Given a particle, to compute the weight we need to first obtain the template $z_t^{(i)} = T(y_t; x_t^{(i)})$. This involves retrieval of elements from the memory (containing the current frame). Further, the evaluation $p(z_t^{(i)}|A_t)$ involves evaluation of the mixture of Gaussian, which is far more complicated than the simple proposal and

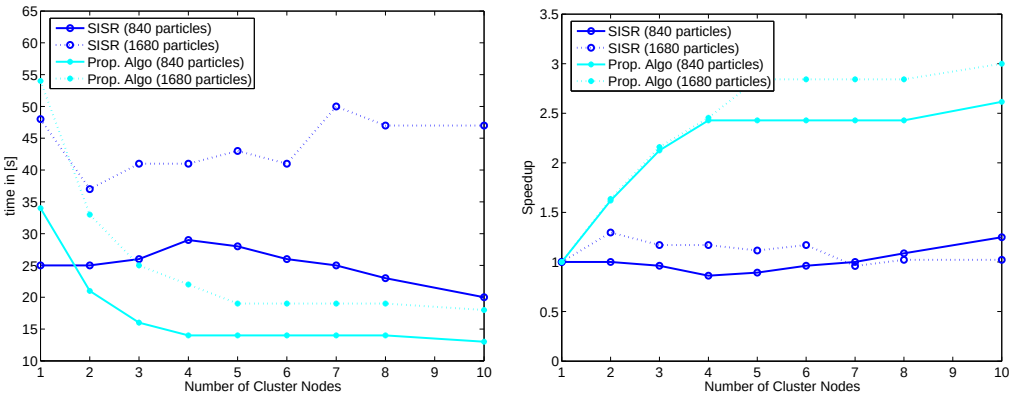
the IMHA blocks. For this reason, we fixed $R_p = R_d = 1$ and analyzed the performance of the architecture for various values of R_w. The implementation of the proposed methodology over the cluster was as follows. Each block (as shown in figure 6.3) was assigned a cluster node for itself, i.e, a total of $R_p + R_w + R_d$ cluster nodes were employed, with R_p of them performing particle proposal, R_w of them computing weights and R_d, the IMH chains. Communication between these PCs was performed using MPICH libraries. Holding the values of R_p and R_w at unity ($R_p = 1 = R_d$), the tracker was tested for varying number of weight computation blocks.

For comparison purposes, we also implemented traditional SISR with the same specifications as the proposed methodology. The main difference was that the node that performed resampling would now wait till all particles are delivered from the weight computation blocks before starting the SR algorithm.

6.4.1.3 Results

The algorithm was used to process 20 frames of a video sequence, tracking a car. Figure 6.4 shows typical tracking results. The filter was run with 840 and 1680 particles, with the number of cluster nodes for weight computation R_w varying from 1 to 6. $R_w = 1$ corresponds to the sequential implementation, and $R_w > 1$ corresponds to the parallel implementation with a single chain. Under the same setup, we tried an implementation of SISR, replacing the IMH Chain with a systematic resampler. The main difference between the algorithms is that the systematic resampler could begin only when all particles were processed and the normalized weights are known.

Figure 6.5(a) shows the actual time taken (in seconds) to process 20 frames of video, with 840/1680 particles for the proposed algorithm and SISR. Note the $1/x$-like decay exhibited by the time taken by the proposed algorithm. Figure 6.5(b) shows the speedup of each algorithm when we add more and more computing nodes. The $1/x$-like behavior now translates to a linear increase in speedup



(a) Timing (b) Speedup

Figure 6.5: (Left) Actual time (in seconds) taken to process 20 frames, with a filter of 840 particles, with varying number of R_w. (Right) Speedup obtained by replication of the weight computation node. Note the linear speedup obtained with the proposed algorithm.

with the number of processing nodes. The two plots demonstrate the pipelinability of the proposed algorithm. It can be seen that the speedup tapers-off as number of cluster nodes increases. This is attributed to increasing communication delays between the nodes. There are no standard models for communication delays when using MPICH. As we use more and more processors, inter-processor communication becomes the dominant source of delay, and further parallelization does not help.

6.4.2 Synthetic Example

We applied the design methodology and implementation strategies proposed here for a synthetic example. The problem specifications were first introduced in [56]. The system has a scalar state space, i.e, $\mathcal{X} = \mathbb{R}$. The state transition model is defined by

$$x_t = x_{t-1} + \frac{25x_{t-1}}{1 + x_{t-1}^2} + 8\cos\left(2(t-1)\right) + w_t, w_t \sim N(0, 10) \tag{6.32}$$

The observation model is given by the equation

$$y_t = \frac{1}{20}x_t^2 + v_t,\ v_t \sim N(0,1) \tag{6.33}$$

We could then estimate the times $T_p : T_w : T_d$ to be in the proportion 19.2 : 1 : 7. For filtering with $Q = 840$ particles, we can now formulate and solve the four convex programs. The convex programs were solved using the Matlab's optimization toolbox. The constraints that are active (the constraints that are satisfied with equality at a feasible point are called *active*) at the minima were noted to give a qualitative interpretation to the result.

Case A: Minimum is achieved with the following two active constraints.

$$\frac{R_p}{T_p} = \frac{R_w}{T_w} = \frac{1}{T_d} \tag{6.34}$$

Note that this is the rate balancing condition. The corresponding minimum time is

$$\tau_{A,\min} = QT_d + T_p + T_w \tag{6.35}$$

Case B: Again the minimum is achieved at the boundary with the same active constraints.

$$\begin{aligned} R_w &= R_p \\ \tfrac{R_w}{T_w} &= \tfrac{1}{T_d} \end{aligned} \tag{6.36}$$

This gives us a minimum time of

$$\tau_{B,\min} = T_p + QT_d + T_w \tag{6.37}$$

Case C: Note that the cost function is independent of R_w. The minimum is achieved at the following active constraint.

$$\frac{R_p}{T_p} = \frac{1}{T_d} \tag{6.38}$$

giving a minimum time

$$\tau_{C,\min} = QT_d + T_w + T_p \tag{6.39}$$

Case D: The cost function is constant over the feasible set. Hence, the minimum time is

$$\tau_{D,\min} = T_p + T_w + QT_d \tag{6.40}$$

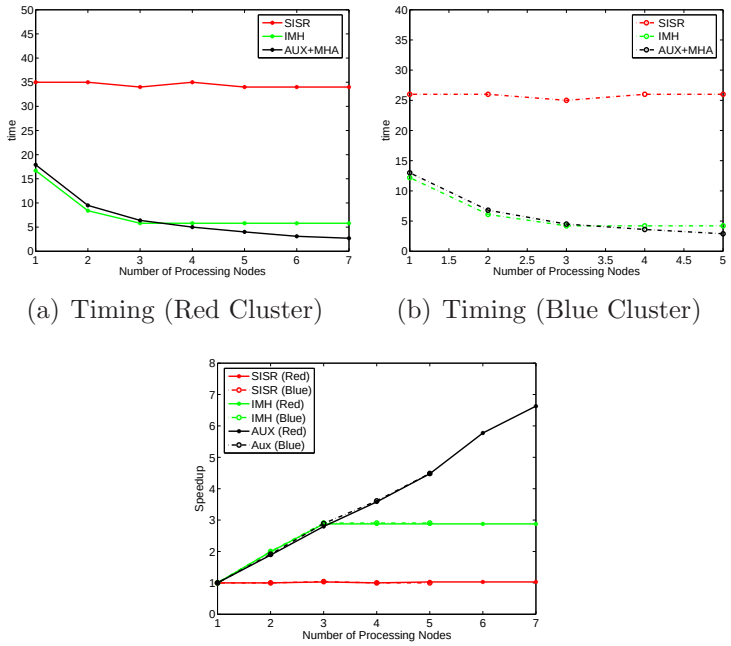
It turns out that all four convex program give the same minimum time, and this also corresponds to the solution given when the rates are balanced as in (6.34). This is interesting as balanced rates have an intuitive appeal.

We implemented three filters and tested them on the Red and Blue clusters. The first two filters were those using SISR and IMH for resampling, with the proposal density, being same as the state transition model. The third filter was based on auxiliary particles, with a complicated proposal density defined as follows:

$$\begin{gathered} g(x_t, k|x_{t-1}y_t) \propto g(x_t|x_{t-1}^{(k)}y_t)g(k) \\ g(k) = c, \\ g(\cdot|x_{t-1}^k y_t) \sim N(\pm\sqrt{20|y_t|} + \hat{x}_{t|t-1}^{(k)}, 1), \\ \text{where } \hat{x}_{t|t-1}^{(k)}, = x_{t-1}^{(k)} + \frac{25x_{t-1}^{(k)}}{1+(x_{t-1}^{k})^2} + 8\cos(2(t-1)) \end{gathered} \tag{6.41}$$

This particular proposal density samples the auxiliary state randomly, and *mixes* the observation with the predicted state to concentrate more particles near the posterior modes.

Figure 6.6 shows the actual time for computation and the achieved speedup with parallelization for the three filters, tested on both clusters. We tested the algorithm for varying R_p as the bottle-neck is initially in the proposal stage. However for $R_p > T_p/T_d \approx 3$ the bottleneck shifts to the IMH sampler and further increase in the value of R_p does not produce any significant gains in the overall processing time. This is reflected in the saturation of the plots associated with the proposed algorithm (IMH) in Figure 6.6. In contrast, in SISR the resampling begins only when all the particles are generated. The overall time for processing



(a) Timing (Red Cluster) (b) Timing (Blue Cluster)

(c) Speedup

Figure 6.6: Timing and speedup over the synthetic example.Saturation occurs after $R_p = 3$ because of the shift of the bottle neck from the Proposal block to the IMH sampler.

does not scale as well. Finally, auxiliary particle filtering scales linearly with the number of processing nodes, and offers the best speedup.

The proposed resampling method and the associated implementation schemes , allows for a pipeline that is free of bottle-necks. Further, implementations using the proposed methodologies show a speedups that increases linearly with the number of processing nodes utilized. This allows for us to parallelize the algorithm to achieve the desired runtime rate. In contrast, implementations based on SISR do not scale that easily with the number of the processing nodes used. systematic resampling step.

Chapter 7

Compressive Acquisition of Visual Signals

Sampling is the process of converting a continuous domain signal into a set of discrete samples in such a manner that it allows the approximate or exact reconstruction of the continuous domain signal from the discrete samples. The Shannon-Nyquist sampling theorem states that if a signal is band-limited, then it lies in a subspace given by discrete set of Fourier basis functions. If a signal lies in a subspace then the signal can be accurately reconstructed by measuring the projections of the signal onto the subspace. The subspace can either be data-independent like Fourier transform, Wavelet transform etc., or data dependent like the principal component analysis (PCA).

More recently, sparse representations and estimation using sparse approximations have gained popularity motivated by recent results from Compressive Sensing (CS). The basic idea is that if a signal is sparse in some basis, then it may be efficiently recovered and reconstructed using few linear measurements of the signal. Compressive sensing has found applications in much the same domains as the subspace sampling. Visual data such as images, videos, light-fields and reflectance fields which were all earlier measured and compressed using subspace sampling can potentially be measured using compressive sensing. We know that in principle if a signal has no further redundancy than sparsity, then it cannot be subspace limited. Similarly, if a signal has no further redundancy than being subspace limited, then sparsity provides no additional information. This leads us to believe that real visual signals are neither truly sparse, nor just subspace compressible. In fact, we argue that these signals are hybrid subspace-sparse signals.

Consider the three images and their wavelet decomposition shown in Figure

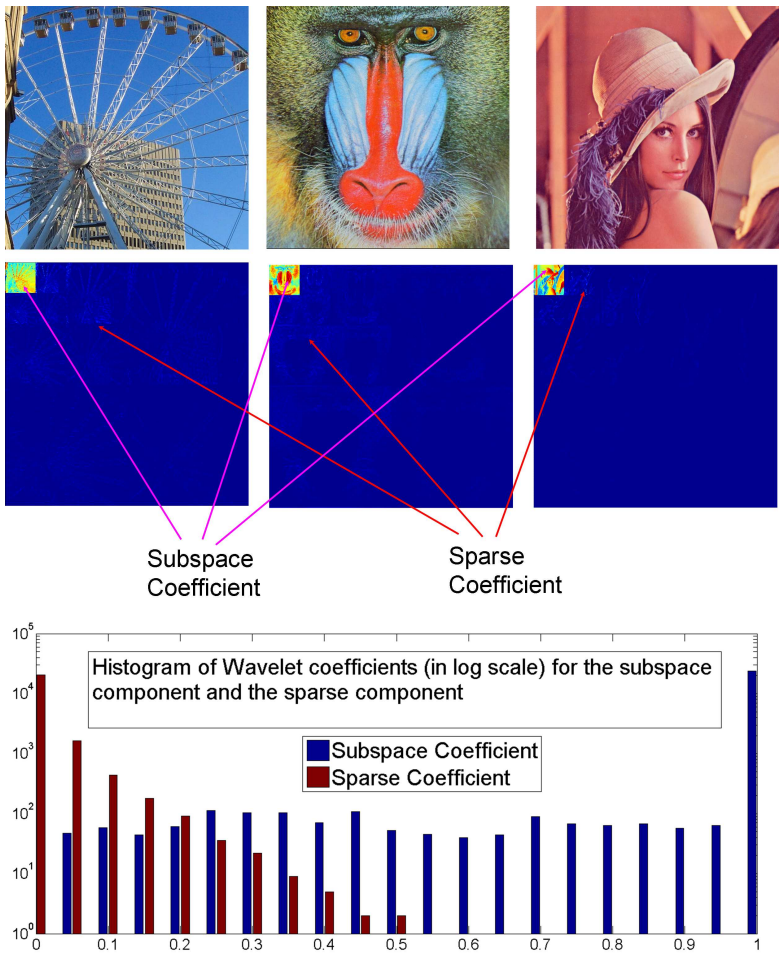


Figure 7.1: Three images and their wavelet transforms. The approximation coefficients of the wavelet decomposition are much more likely to be non-zero than the detail coefficients which are sparse. (Bottom) The histogram of the wavelet coefficients (in log scale) for the approximation (subspace) and the detail (sparse) coefficients. Notice that while the detail coefficients have a very high likelihood of being zero and are therefore sparse, the approximation coefficients are almost always non-zero. This indicates that images are hybrid subspace sparse signals.

7.1. As can be seen the wavelet domain coefficients of images have the following property (a) The lower frequency approximation coefficients of the wavelet decomposition are almost always non-zero (b) The detail coefficients of the wavelet transform are sparse. This indicates that to optimally encode, compress or measure such signals one needs to take into account a signal model that accounts

for both these properties simultaneously. Similar properties are true for several classes of visual signals including reflectance fields (Refer Figure 7.2). This suggests that a wide variety of visual signals including images, reflectance fields all have the property that they contain significant energy in a fixed subspace and the components orthogonal to that subspace are sparse. Here, we develop a Hybrid Subspace Sparse signal model which models signals as being a sum of two terms one which lives in a fixed low dimensional subspace and another which is sparse in the null-space. We derive optimal projections for measuring such signals and propose a reconstruction algorithm that is based on solving a convex optimization problem.

Compressive sensing revolves around the notion of sparse representations of signals. Compressive sensing [7, 19, 39] has been successfully applied to various acquisition problems in vision and graphics. Noteworthy among these are the single pixel camera (SPC) [44] for image acquisition, the compressive acquisition of reflectance fields [102, 122], and the L_1 regularized occlusion insensitive face recognition system [136].

Our goal is on exploiting structure inherent to a wide range of images/imaging data. Baraniuk et al. [8] propose the paradigm of model-based compressive sensing where prior structure of sparsity if exploited to get better signal recovery properties. In a sense, the hybrid subspace signal model fits under the paradigm of model-based compressive sensing. Eldar and Mishali [45] propose a sparse mixture of subspace representation for signals.

We present Reflectance Field (RF) acquisition as an empirical case-study for our signal model. Imaging objects and scenes under varying illumination conditions is an important problem in several imaging, computer vision and graphics tasks. Firstly, obtaining the images of an object with varying illumination allows for the study of reflectance properties of materials which is an important ingredient in solving problems such as illumination invariant recognition [9]. Empirical studies relating to the reflectance field of face images have led to significant ad-

vances in illumination-invariant face recognition [79] [11] [54] and a similar understanding of the reflectance properties of everyday objects might lead to extending such results for arbitrary objects. Acquiring these images are the basis of several relighting algorithms that are based on image-based rendering. Several recent algorithms for fast photo-realistic relighting and rendering of dynamic scenes have been based on a model of BRDF or the reflectance fields [12] [78] of these scenes and such methods are gaining in popularity. One of the primary disadvantage with current methods for acquiring reflectance fields is the fact that the number of images required for reconstruction increases quadratically with the sampling on the angular dimension (number of images required is equal to the number of individual lighting directions which increases quadratically with the required angular sampling). Here, we show that modeling reflectance fields as Hybrid Subspace Sparse signals results in a technique for reflectance field acquisition, where the number of required images increases linearly with the sparsity of the reflectance field which is far smaller than the number of lighting directions.

7.1 Sensing Signals: Subspace and Sparse

In this section, we describe two important signal sensing strategies: subspace sampling and compressive sensing.

7.1.1 Subspace Sampling

Subspace sampling is an efficient acquisition method for sensing signals that are well represented by a linear basis. Consider a signal $\mathbf{x} \in \mathbb{R}^N$ and a linear basis $\mathbf{B} = [\mathbf{B}_p \ \mathbf{B}_p^{\perp}]$. Further, the subspace defined by the span of $\mathbf{B}_p \in \mathbb{R}^{N \times M_1}$ approximates the signal to a high degree, in the sense, that $\|B_p^T \mathbf{x}\|/\|\mathbf{x}\| = 1 - \delta$, where δ is small. Under such a scenario, just sensing the projections of the signal $\mathbf{x}$ over the subspace defined by $\mathbf{B}_p$ is sufficient to obtain a high approximation of the signal. That is, we can sense $\mathbf{s}_p \in \mathbb{R}^{M_1}$, such that $\mathbf{s}_p = \mathbf{B}_p \mathbf{x} + \text{noise}$. The measured

coefficients can be used to reconstruct $\hat{\mathbf{x}}_1 = B_p \mathbf{x}_p$.

7.1.2 Compresive Sampling

Consider a signal $\mathbf{x} \in \mathbb{R}^N$, which is sparse in a basis $\mathbf{B}$, that is, $\mathbf{s} \in \mathbb{R}^N$ defined as $\mathbf{x} = \mathbf{B}\mathbf{s}$ is sparse. We call a vector K-sparse if it has utmost K non-zero components, or equivalently, if $\|\mathbf{s}\|_0 \leq K$, where $\|\cdot\|_0$ is the L_0 norm or the number of non-zero components.

We are interested in the problem of sensing the signal $\mathbf{x}$ from linear measurements. Ideally, with no additional knowledge about $\mathbf{x}$, we would require N linear measurements of $\mathbf{x}$, which would then form an invertible linear system. The theory of compressed sensing shows that it is possible to reconstruct $\mathbf{x}$ from M measurements even when $M << N$ by exploiting the sparsity of $\mathbf{s} = \mathbf{B}^T\mathbf{x}$. Consider a measurement vector $\mathbf{y} \in \mathbb{R}^M$ obtained using a $M \times N$ measurement matrix $\mathbf{\Phi}$, such that

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + e = \mathbf{\Phi}\mathbf{B}\mathbf{s} + e \tag{7.1}$$

where e is the measurement noise. For $M < N$, estimating $\mathbf{x}$ from the linear measurements is an ill-conditioned problem. However, when there exists a basis $\mathbf{B}$ such that $\mathbf{s}$ as defined above is K sparse, then compressive sensing allows for recovery of $\mathbf{s}$ (or alternatively, $\mathbf{x}$) from $M = O(K \log(N/K))$ measurements. In particular, when $\mathbf{B}$ is a fixed basis, it can be shown that using a randomly generated measurement matrix Φ allows for the recovery of $\mathbf{x}$ with a high probability. Typical choices for such measurement matrix are the Bernoulli matrices and the Gaussian matrix [19]. Such randomly generated matrices satisfy the *Restricted Isometry Property* [6, 20], which ensures that all sub-matrices formed by choosing (a certain number of) columns of $\mathbf{\Phi}$ are orthogonal (and hence, invertible) with a high probability.

7.1.3 Signal Recovery

Estimating K sparse vectors that satisfy the measurement equation of (7.1) can be formulated as the following L_0 optimization problem:

$$(P0): \quad \min \|\mathbf{s}\|_0 \text{ s.t. } \|\mathbf{y} - \mathbf{\Phi B x}\|_2 \leq \epsilon \tag{7.2}$$

where the L_0 norm, $\|\cdot\|_0$ counts the number of non-zero elements. This is typically a NP-hard problem. However, the equivalence between L_0 and L_1 norm for under-determined linear systems [40] allows us to reformulate the problem as one of L_1 norm minimization.

$$(P1): \quad \min \|\mathbf{s}\|_1 \text{ s.t. } \|\mathbf{y} - \mathbf{\Phi B s}\| \leq \epsilon \tag{7.3}$$

with ϵ being a bound for the measurement noise e in (7.1). It can be shown that the solution to the (P1) is with a high probability the K sparse solution that we seek. There exist a wide range of algorithms that solve $P1$ to various approximations or reformulations [21] [67] [39]. Most of them note that the problem $(P1)$ is a convex problem, and in particular, can be recast as a Second Order Cone Program (SOCP) for which there exist efficient numerical techniques. There also exist greedy techniques for solving the same problem such as matching pursuit [88], orthogonal matching pursuit [100] and CoSaMP [96].

7.2 Hybrid Subspace Compressive Sensing

Invariably, real world signals are neither completely subspace compressible nor are they truly sparse. Towards this end, we propose a hybrid signal model, that encompasses both Subspace sampling as well as compressive sampling.

We model the signal $\mathbf{x} = \mathbf{B}_p\mathbf{s}_p + \mathbf{B}_c\mathbf{s}_c$, where $\mathbf{s}_p \in \mathbb{R}^{N_1}$, $\mathbf{s}_c \in \mathbb{R}^{N_2}$, such that $N_1+N_2 = N$. Further, $\mathbf{s}_c$ is assumed to be sparse. We are now interested in solving

the inverse problem of find $\mathbf{x}$ from under-determined set of linear measurements.

$$\mathbf{y} = A\mathbf{x} = A\mathbf{B}_p\mathbf{s}_p + A\mathbf{B}_c\mathbf{s}_c = A_p\mathbf{s}_p + A_c\mathbf{s}_c + \mathbf{n} \tag{7.4}$$

where $\mathbf{n}$ is additive measurement noise. Under the constraint that $\mathbf{s}_p$ is sparse, we now pose the problems of the (optimal) choice of measurement matrix A and the inversion algorithm to recover $\mathbf{x}$.

7.2.1 A Two-Step Sensing Algorithm

Before we describe the algorithm for recovering $\mathbf{x}$ from measurements $\mathbf{y}$, it is interesting to consider a two step sensing and recovery process. The first step consists of measuring the projection of the signal over the subspace $\mathbf{B}_p$, and recovering the part of the signal $\mathbf{x}$ that is spanned by the subspace.

$$\mathbf{y}_p = \mathbf{B}_p^T\mathbf{x} + \mathbf{n}_p \tag{7.5}$$

We can now recover the part of signal $\mathbf{x}$ spanned by the subspace $\mathbf{B}_p$,

$$\hat{\mathbf{x}}_p = \mathbf{B}_p\mathbf{y}_p \tag{7.6}$$

The second step works under the premise that $\mathbf{x}_c = \mathbf{x} - \mathbf{x}_p$ is sparse, and hence poses the recovery of $\mathbf{x}_c$ under the traditional compressive sensing framework. Let $\mathbf{y}_c = \mathbf{\Phi}\mathbf{x} + \mathbf{n}_c$, with $\mathbf{\Phi}$ a random measurement matrix (or more generally, incoherent with $\mathbf{B}_c$). The sparse part of the signal can now be recovered by solving the following problem:

$$\min_{\mathbf{s}_c} \|\mathbf{x}_c\|_1 \text{ s.t } \|\mathbf{y}_c - \mathbf{\Phi}\mathbf{B}_c\mathbf{s}_c - \mathbf{\Phi}\hat{\mathbf{x}}_p\|_2 \leq \epsilon \tag{7.7}$$

While seemingly intuitive, the two step process is only correct when the basis $\mathbf{B}_p$ and $\mathbf{\Phi}$ are orthogonal complements of each other. When this condition is not

satisfied, the measurements $\mathbf{y}_c = \mathbf{\Phi x}$ multiplex $\mathbf{s}_c$ as well as $\mathbf{s}_p$. Solving for $\mathbf{s}_c$ separately as defined in (7.7) ignores potential encoding of $\mathbf{s}_p$ in $\mathbf{y}_c$. In essence, the problem definition in (7.7) treats $\mathbf{x}_p$ as a static variable, inspite of its dependence on the variable.

A second problem of the above solution is that is lacks the *Universality* property of compressive sensing algorithms. Universality suggests that we do not need to know the sparsifying basis at the time of sensing[1]. The two-step approach outlined above depends strongly on subspace projection which violates the universality principle. A practical implication of this is that the algorithm will not be able to work with available compressive sensing data (such as those for images) that have been captured only with random measurements or under a different choice for the subspace $\mathbf{B}_p$.

However, inspite of this limitation the two step sensing approach reveals certain properties of the Hybrid Sensing paradigm. To begin with, if the signal were to truly obey the model, then the sparsity of $\mathbf{s} = [\mathbf{s}_p^T \mathbf{s}_c^T]^T$ is $N_1 + K$ and would require $O((N_1 + K)\log(N/(N_1 + K)))$ measurements (for solving the L_1 problem). In contrast, the two step procedure suggests that the total number of measurements required for the Hybrid Signal model is $N_1 + O(K \log(N_2/K)$. In essence, the gain of assuming the Hybrid signal model is that the location of N_1 components of the sparse vector is known a priori, and injecting the knowledge into both the sensing and reconstruction algorithm allows us to recover the signal with lesser measurements (or alternatively, get better SNR for the same number of measurements). This savings in measurements is extremely significant for high dimensional problems such as compressive imaging. For example, when sensing a 1 megapixel image, if $K = 5000$ and $N_1 = 2000$, then this reduces the number of measurements required from about 27000 compressive measurements to 15000 HSS measurements in order to obtain the same accuracy during reconstruction.

[1]Hence, the use of random measurements which are incoherent with any fixed basis.

7.2.2 Problem Formulation

The shortcomings of the two-step recovery algorithm discussed above can be fixed if we optimize for the subspace coefficients and the sparse coefficients simultaneously. With this we can formulate the problem:

$$(P3)\ \min_{\mathbf{s}} \|\mathbf{s}_p\|_2 + \|\mathbf{s}_c\|_1 s.t. \|\mathbf{y} - A_p\mathbf{s}_p - A_c\mathbf{s}_c\| \leq \epsilon \tag{7.8}$$

The cost function is made up of the sum of $\|\mathbf{s}_p\|_2$ and $\|\mathbf{s}_c\|_1$, the former leading to MMSE solution and the latter a sparse solution. In the absence of the $\|\mathbf{s}_c\|_1$ term, the solution of (7.8) would be the projection of data onto the span of $\mathbf{B}_p$. Alternately, removing $\|\mathbf{s}_p\|_2$ from the cost function leads to the standard L_1 recovery problem in compressive sensing.

It can also be shown that the solution to $(P3)$ ensures no overlap between the sparse part of the solution $\mathbf{x}_c = \mathbf{B}_c\mathbf{s}_c$ and the subspace defined by the columns of $\mathbf{B}_p$.

7.2.3 Recovery Algorithm

Problem $(P3)$ is a convex problem, and can be solved using any general purpose solver. However, there have been a significant work in designing faster and efficient methods towards solving the basic CS problem formulations (including $(P0)$, $(P1)$ and their variants). Reusing these algorithms to solve $(P3)$ is immensely beneficial.

Note the similarity between the definition of $(P2)$ and $(P3)$ in the basic formulation of the optimization problem as a convex cost, with quadratic constraints. However, $(P2)$ can be recast as a problem with *Linear* costs and quadratic constraints, making it a SOCP. Similarly, $(P3)$ also maps onto a SOCP by using the epigraph re-formulation [18].

Bayesian recovery algorithms work with various sparsity priors on $\mathbf{s}$, under Gaussian measurement noise. In particular, BCS [67] uses the RVM [129], frame-

work to solve for the posterior $p(\mathbf{s}|\mathbf{y})$. The problem definition of $(P3)$ can be easily mapped onto the RVM framework simply by re-deriving the basic theory of RVM with Gaussian priors on $\mathbf{s}_p$ and Gaussian with Gamma hyper-priors on $\mathbf{s}_c$. Alternatively, in practice, setting the hyper-prior parameters corresponding to $\mathbf{s}_c$ to zero (in practice, very small) would *coerce* the EM algorithm to allow arbitrary values for $\mathbf{s}_p$.

However, for the experiments performed, we use the iterative recovery approach called *CoSaMP* [96]. CoSaMP works on selecting and refining a sparsity support set for the signal $\mathbf{s}$. CoSaMP also allows for efficient implementations and has strong convergence guarantees [96]. Further, the CoSaMP algorithm is easily modified to solve $(P3)$, by forcing the support set to include $\mathbf{s}_p$, and in essence performing support selection over $\mathbf{s}_c$. We discuss these reformulations and detailed solution outline of the modified CoSaMP algorithm in Table 7.1.

7.2.4 Choice of Measurement Vectors

The two step solution suggests that for a given sparsity level K on $\mathbf{s}_c$, a total of $N_1 + 2K$ or $N_1 + O(K \log(N_2/K)$ measurements are required (depending on the inversion algorithm used). In this regard, an obvious choice of measurement matrix is using subspace sampling along with compressive measurements, i.e, setting $A = [\mathbf{B}_p^T \ \mathbf{\Phi}^T]^T$, such that $A_p = \mathbb{I}_{N_1}$ and $A_c = \mathbf{\Phi}\mathbf{B}_c$.

However, an interesting question to be posed is when the measurement matrix A is a completely random matrix. Analyzing the restricted isometry property under the hybrid subspace signal model is of future research interest. In our experiments, we show examples of reconstructions using only random measurements.

7.3 Reflectance Field Acquisition

Bidirectional Reflectance Distribution Function (BRDF) is a 4-dimensional function that describes how an opaque surface point reflects incoming light [97]. The

$\mathbf{s}$= CoSaMP_Modified($\mathbf{\Phi}$, $\mathbf{y}$, K, M_1)
Input: $\mathbf{\Phi}$: $(M \times N)$ sensing matrix $\mathbf{y}$: $(M \times 1)$ measurement vector K : Sparsity level of compressive part of signal M_1 : No. of Subspace measurements **Output:** $\mathbf{s}$: Output vector. **Notation:** $A^\dagger$: Pseudo-inverse of matrix A $A_{\|T}$: Columns of matrix A corresponding to index set T $\mathbf{s}_{\|T}$: The components of vector $\mathbf{s}$ indexed by T $\text{supp}(\mathbf{u}; 1 : M, K)$: Support of K non-subspace elements of $\mathbf{u}$ that have maximum strength, along with the subspace component support denoted as $1 : M$.
$T = \{1, \ldots, M_1\}$ $\mathbf{s}^0_{\|T} = (\mathbf{\Phi}_{\|T})^\dagger \mathbf{y}$ $\mathbf{s}^0_{\|T^c} = \mathbf{0}$ $k = 0$ while stopping criterion not satisfied $\mathbf{v} = \mathbf{y} - \mathbf{\Phi}\mathbf{s}^k$ $\mathbf{u} = \mathbf{\Phi}^T \mathbf{v}$ $\Omega = T \cup \text{supp}(\mathbf{u}; 1 : M_1, 2K)$ $\mathbf{r} = (\mathbf{\Phi}_{\|\Omega})^\dagger \mathbf{y}$ $T = \text{supp}(\mathbf{r}; 1 : M_1, K)$ $k = k + 1$ $\mathbf{s}^k_{\|T} = (\mathbf{\Phi}_{\|T})^\dagger \mathbf{y}$ $\mathbf{s}^k_{\|T^c} = \mathbf{0}$ end $\mathbf{s} = \mathbf{s}^k$

Table 7.1: The modified CoSaMP algorithm for Subspace Sparse signals

incoming lighting direction θ is a two-dimensional quantity (azimuth and elevation) while the outgoing lighting direction is another two-dimensional quantity making the BRDF 4-dimensional function $B(\theta;\phi)$. The slice of the BRDF corresponding to the viewing direction being fixed (as would be the case when the scene is being observed by a fixed camera), while the incident illumination changes freely, is called the reflectance field and is a two-dimensional function $R(\theta)$. Here we show that these reflectance fields $R(\theta)$ strongly follow the HSS signal model and therefore can be accurately measured using very few projections as predicted by the HSS sensing theory we discussed earlier.

7.3.1 Models for Reflectance Fields

The reflectance properties of a material surface is useful for accurately rendering the surface in graphics and relighting applications. Nevertheless the inherent high dimensionality of the reflectance data, together with its high frequency nature (especially at specular surfaces) imply that one has to sample the reflectance fields at very high resolutions in order to obtain accurate and realistic relit images. One way to tackle this data deluge is to discover appropriate basis for representing BRDF's. Spherical Harmonics [135], wavelets [77] and zernike polynomials [76] have all been used as appropriate basis in order to compress the huge amount of data generated by high resolution BRDF's. Another approach to tackle the data deluge is to come up with specific nonlinear models of reflectance functions. The parameters of the model then describe the reflectance functions and only these parameters have to be stored and retrieved. The physically inspired analytical reflection models such as the Phong model [104], Blinn model [14], Oren-Nayar [99] model and the Cook and Torrence model [35] capture the analytical properties of these reflectance fields to varying degrees of fidelity. The problem with such analytical models is that while they allow for exceptional compression in data storage they do not afford any compression in the data capture process.

7.3.2 Hybrid Subspace Sparsity of Reflectance Fields

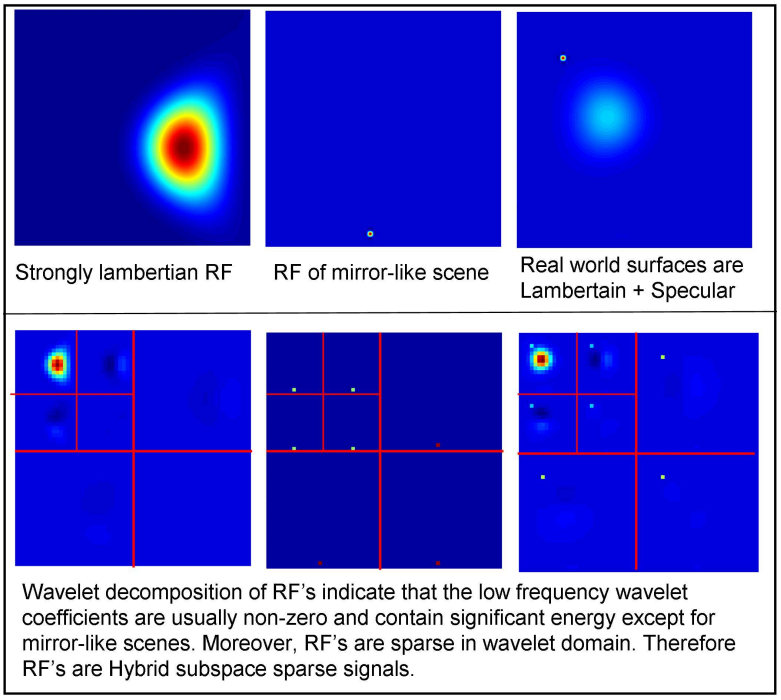


Figure 7.2: (Top) 64×64 reflectance fields of a strongly Lambert (face), strongly specular and a natural surface (Lambert+specular). (Bottom) Two level wavelet transforms of the RF's show that these signals are well approximated as Hybrid Subspace Sparse signals.

Reflectance fields turn out to be ideal candidates for the Hybrid Subspace Sparse signal model. Real world materials are typically either strongly Lambert, strongly specular or somewhere between the two. Let us consider the statistical properties of the reflectance fields of naturally available materials when these RF's are projected to a canonical basis such as the wavelet or the discrete cosine transform (DCT) basis. Shown in figure 7.2 are shown the 64×64 RFs of a single pixel from the MERL Face dataset and the a single pixel from the specular MERL BRDF dataset. Shown below are the wavelet transform domain coefficients. It can clearly be seen while the transform domain coefficients of the specular pixel are truly sparse, the transform domain characteristics of the face pixel is strongly

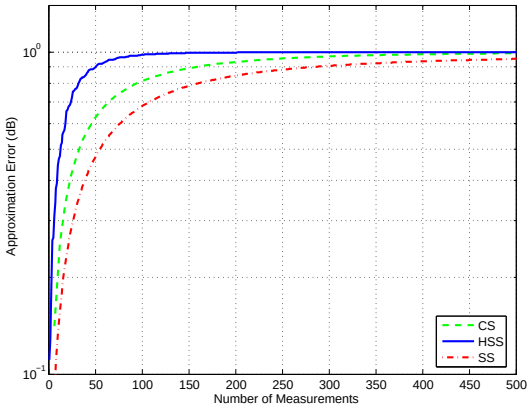


Figure 7.3: Comparison of the energy compaction (or approximation error) Vs number of measurements required for 64×64 reflectance fields for the MERL Face BRDF dataset. The Haar wavelet basis was used for all three sampling schemes. Notice that the HSS signal model captures significantly higher signal energy for the same number of measurements.

subspace sparse, i.e., it contains (a) significant energy in the low frequency coefficients (b) the high frequency terms in the transform domain are sparse. Thus, we see that real-world reflectance fields follow the HSS signal model. Figure 7.3 shows the energy compaction of the reflectance fields for the MERL Face dataset characterizing the BRDF for various regions of faces. Nevertheless, notice that under the Haar wavelet basis provides efficient energy compaction enabling capture of RF's by either subspace sampling [101] or compressive sampling [102, 122] or Hybrid subspace sparse sampling proposed here. Further notice that the HSS signal model better approximates reflectance fields than either the subspace or the sparse signal model and this would lead to significant reduction in the number of measurement required.

7.3.3 Compression Ratios and Angular Frequency

The use of HSS or the CS signal model for reflectance field acquisition becomes more and more relevant as the angular sampling along each direction of the incoming light increases. Let N_m be number of samples that we want along each angular

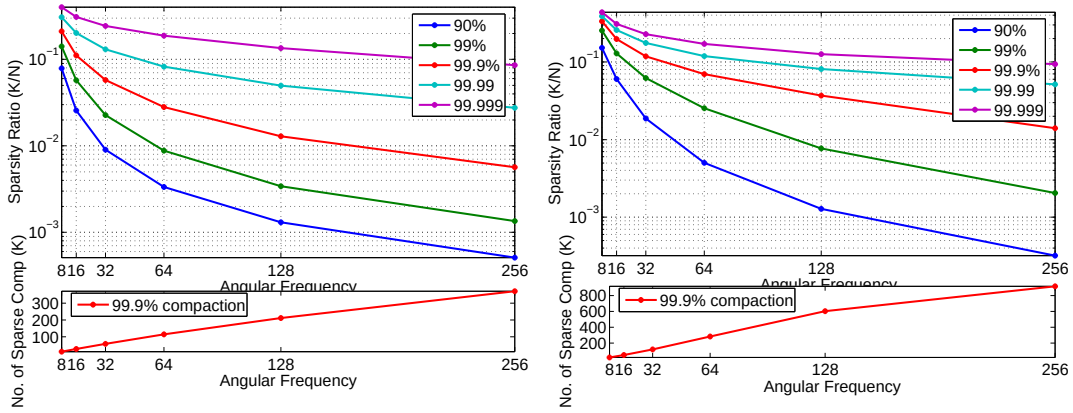


Figure 7.4: Average sparsity levels of reflectance fields in the MERL specular object database [91] (left) and MERL Face BRDF dataset (right) for various sampling frequencies along the angular directions of incoming light. (top) The ratio K/N reflects the inherent compressibility of the signals. It is seen that this ratio decreases linearly (in log-space) as the angular sampling increasing, implying significant reduction in capture time as the resolution of capture time increases. The various plots show compression ratios at different energy compaction levels. (bottom) The values for sparsity level K for various angular resolutions increases linearly.

dimension of the incoming light direction. Hence, the total dimensionality of the problem $N = N_m * N_m$. As N_m increases, we capture higher frequency components of the light fields leading to capture of small nuances in the lighting orientations. However, as a consequence, N increases (quadratically in N_m), thereby leading to larger capture time for a specific capture SNR. This often becomes a limiting factor in the acquisition of high-resolution reflectance map of a scene.

However, the number of sparse components does not increase quadratically when the angular resolution changes. To verify this, we computed sparsity statistic over thousands of reflectance fields from the MERL Reflective surface databases and the MERL face BRDF database. Figure 7.4 show that for a given energy compaction level, the number of sparse components increase linearly. The achievable compression ratio, which is proportional to $N/K = N_m^2/K$, hence, increases linearly with N_m. This is also reflected in the linear decrease (in log scale) in the sparsity ratio plots. This implies that the capture process for a given output SNR

increases linearly with increasing angular resolution, as opposed to quadratic to Hadamard based multiplexing [121]. The sparsity structure of highly specular surfaces is very different from the sparsity structure of strongly Lambertian scenes (like face). While strongly Lambertian scenes such as faces have significant energy in low frequency components, if traditional compressive sensing based random measurements of the signal are obtained, these measurements are not sufficient to reconstruct the fine shading details that are necessary to capture the RF's of such surfaces. For strongly specular scenes, projection onto any fixed basis (i.e., traditional subspace sensing) is highly inadequate since such surfaces have very high frequency components in their RF's. The HSS signal model can adequately model both scenes and lead to accurate reconstruction of RF's for almost any real surface.

7.3.4 HSCS Acquisition of RF

In order to evaluate the HSCS acquisition of reflectance fields, we first need to measure linear projections of the reflectance fields that account for both subspace and random measurements required for compressive sampling. If the reflectance field of a single pixel is represented by x, then the measurement can be modeled as,

$$y = Ax; \tag{7.9}$$

where A is made up of two components $A1$ and $A2$. $A1$ accounts for the subspace components and consists of the lower frequency discrete cosine transform basis coefficients since we know that all RF's have significant energy in these components. $A2$ accounts for random binary (0,1) measurements required for sparse signal reconstruction. In order to experimentally evaluate the presented design we implement a setup very similar to the setup described in [121]. A projector projects random binary patterns on to wall. The reflection from the wall acts as the secondary light source illuminating the scene. Images are captured using

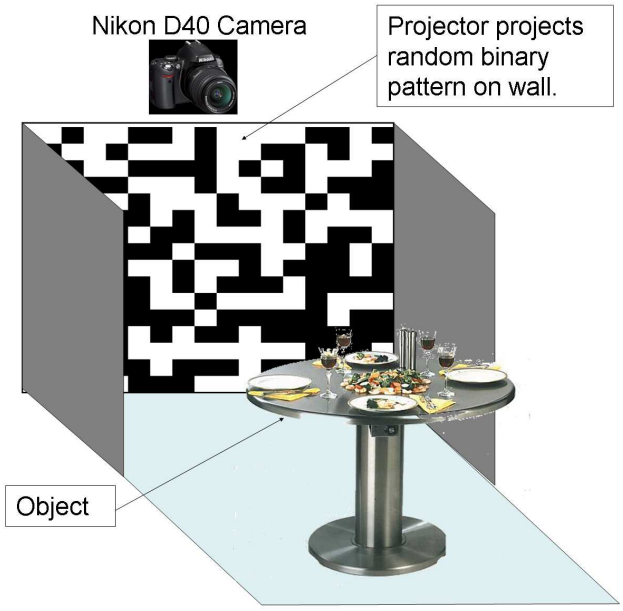


Figure 7.5: Illustration showing the experimental setup. A projector projects random binary patterns on to wall. The reflection from the wall acts as the secondary light source illuminating the scene. Images are captured using a Nikon D40 camera with 50mm f1.8 lens.

a Nikon D40 camera with 50mm f1.8 lens. The setup is illustrated in Figure 7.5. Once the measurements are obtained we follow the reconstruction algorithm presented in 7.2, in order to reconstruct the RF's.

7.4 Experiments

7.4.1 Evaluation SNR vs compression rate

In Section 7.3.2, we showed that reflectance fields of common objects are indeed extremely sparse and compressible in various basis. In particular, the Haar wavelet basis led to significant energy compaction of these reflectance fields. Note, that even though almost all reflectance fields have all their energy in very few of the Haar wavelet components, the exact components that contain this energy are

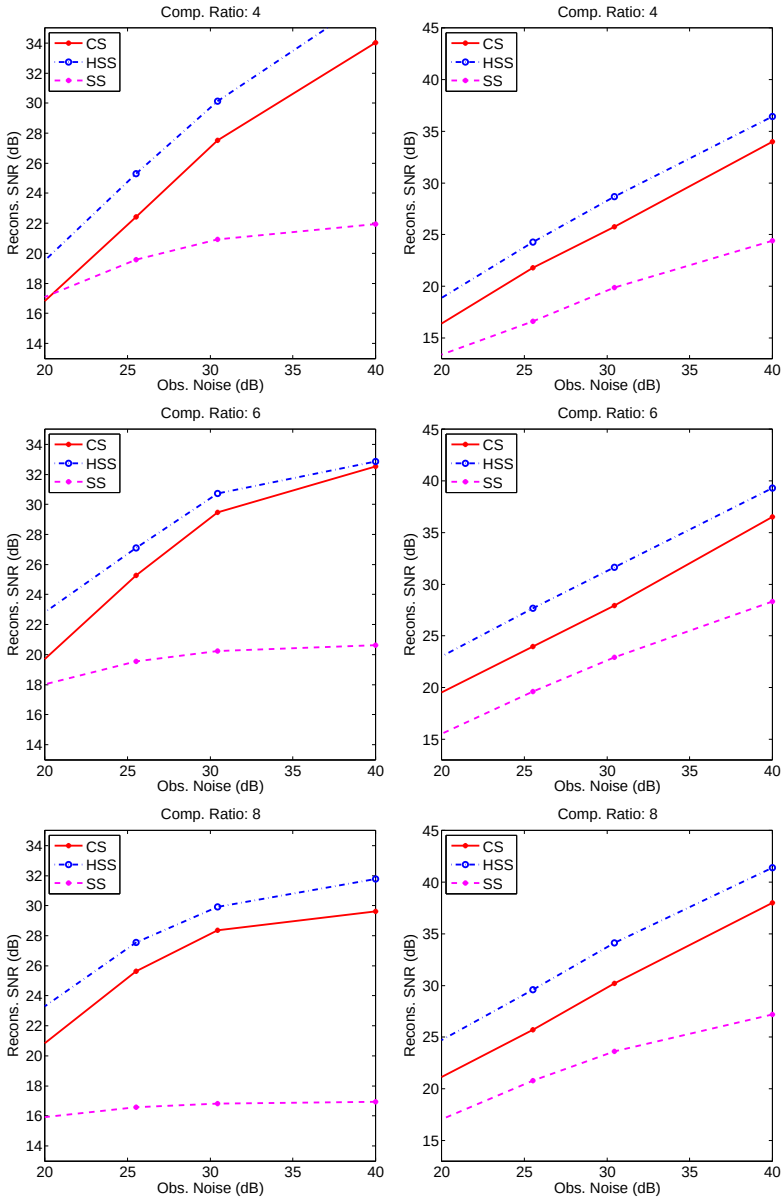


Figure 7.6: Reconstruction error of subspace sampling, compressive sampling and the proposed Hybrid subspace sparse sampling scheme over two publicly available reflectance field datasets. (Left col.) MERL Face dataset (Right col) MERL specular objects dataset. Notice that even though the two databases have very different RF characteristics (face - smooth, specular object - peaky) the proposed HSCS method performs better than traditional methods on both databases.

different for different surfaces (and their orientation with respect to the lighting direction). Therefore, just measuring the projections of these reflectance fields onto a fixed Haar basis and inversion would not lead to significant compression. Instead, in order to exploit the sparsity of the RF's in Haar wavelet basis, we perform an experiment using 100 reflectance fields from the MERL BRDF dataset. Each reflectance field is a 64×64 matrix therefore containing a total of 4096 unknowns. We measure the M independent projections of these RF's under random binary lighting. The number of such measurements M is varied. The noise level in the observation is also systematically varied. We then assume that the RF is sparse in the Haar wavelet basis and perform inversion using Bayesian Compressive Sensing. Shown in Figure 7.6 are the signal to noise ratios of the recovered reflectance fields for various compression factors (compression factor $= N/M$). Notice that one can obtain a compression factor of about 16-32, with significant signal to noise ratio. Also, notice that the signal to noise ratio of the recovered RF is greater than the input observation SNR for compression factors that are less than 8. This shows that in the presence of moderate noise and low compression ratios, the compressive inversion algorithm additionally performs smoothing thereby improving the SNR.

7.4.2 Real Experiments on capture of reflectance fields

We used the experimental setup described in Section 7.3.4 in order to capture the reflectance fields of several objects. Figures 7.7 and 7.8 show reconstruction results over a 16×16 and a 32×32 problem respectively. The *Robot* dataset was acquired using $0-1$ Bernoulli random measurements[2]. Reconstruction was done with 128 measurements, using the Haar basis with the lower $M_1 = 64$ coefficients for $\mathbf{B}_p$. The *Nemo* dataset was collected using a mix of subspace projection and compressive measurements. A total of 256 measurements were taken, of which

[2]All images were obtained in HD by taking multiple images at different exposures and averaging them

Figure 7.7: Reconstructed 16×16 RF of a robot exhibiting both Lamberts and Specular shading from 128 compressive measurements using a HSS model. The top 64 Haar coefficients were chosen as the subspace B_p.

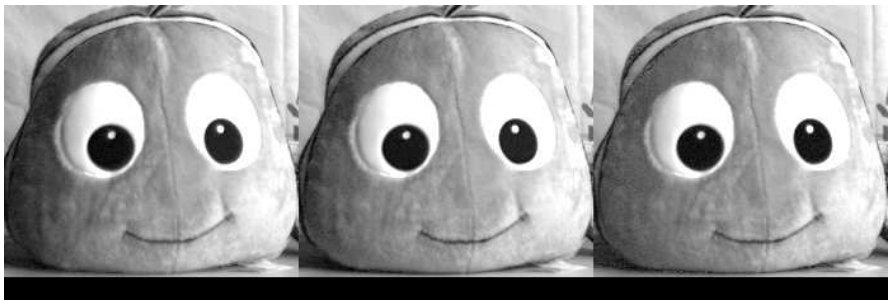


Figure 7.8: Reconstructed 32×32 RF of a soft toy (Nemo) exhibiting Lamberts shading from 256 measurements using a HSS model. The top 128 Haar coefficients were chosen as the subspace $\mathbf{B}_p$.

128 were the top Haar basis elements. Reconstruction was performed again with the Haar basis, with the top 128 coefficients chosen for $\mathbf{B}_p$. Note how the results capture the gentle shading on the sides of the subject. Relighting results are shown in Figure 7.9.

Figure 7.9: Relighting experiment on a 16×16 acquisition problem using the lighting shown in the inset figure. The RF was acquired from 192 measurements. The bottom row shows the blending of lighting in detail.

Chapter 8

Future Directions

There are multiple avenues in theory as well as applications for extending the findings of this dissertation. We discuss a few below.

8.1 Projective Geometry and Grassmann Manifold

Much of the analysis in the dissertation is done under the assumption of replacing projective transformation by the direct linear transform (DLT). For the applications discussed in the dissertation this serves as a reasonable approximation. However, the DLT does not explain what happens at the Line at Infinity and leaves a interesting open problem.

An interesting avenue for pursuing this lies in choosing an appropriate model for points that lie on projective spaces. Recall that $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in \mathbb{P}^n$ can be represented in the homogeneous coordinates. In the homogeneous coordinates, $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ are identical if $\tilde{\mathbf{x}} = \lambda\tilde{\mathbf{y}}$ or equivalently they lie on a line connecting them to the origin. Equivalently, a point on $\mathbb{P}^n$ is identical to the one dimensional subspace generated by it. In such a viewpoint, points on the projective space can be mapped onto a well studied manifold called the Grassmann Manifold. The Grassmann manifold is defined as the space of all linear subspaces of a vector space. There has been a lot of work on developing appropriate theory and tools for the geometry [1] as well as the statistics [33] on the Grassmann manifold. A study in understanding how projective transformations affect statistics on these manifolds might help in formulating a theory that address the shortcomings of existing literature.

8.2 Steering PTZ cameras for improved inference

Pan-Tilt-Zoom (PTZ) cameras allow for steering of the camera view keeping the camera center (pinhole) in place. Such a rotating (pan and tilt) and zooming camera produces views that satisfy elegant geometric properties [123] [128] [125]. The advantage of using PTZ cameras is that one may now control the specific settings of the PTZ cameras so as to improve tracking and recognition performance. Thus while one or few 'master cameras' observe a wide sensing field of view, their tracking results in turn guide the PTZ controls for other slave cameras that can then zoom into objects of interest in order to obtain high resolution imagery of these objects. In general, such a problem may be posed as an optimization of some desirable cost functional over the steering controls of the cameras. One example of such a cost functional would be the average tracking error over the whole scene. Here, we want to obtain new views (using the PTZ controls) that minimize the desired objective. We envision steering algorithms that operate some of the cameras in the slave mode, tethered with inputs from other cameras that sense the entire region of interest.

8.3 Distributed particle Filtering

This dissertation address the challenges in reducing the computational complexity of particle filters by making the computations inherently parallel. However, in a smart camera network, the sensors as well as the processing power is distributed. In order to make the filter truly distributed and enable its implementation on huge camera networks containing hundreds of cameras one needs to pay attention to methods that enable these particle filter-based estimates to be performed in a distributed manner. This can be achieved either using *Synchronized Particle Filtering* or by the more general means of *Distributed function estimation*.

8.3.1 Synchronized Particle Filtering

One way to *decentralize* the filter operations is to replicate it identically at each node. For particle filtering, this can be done easily if the random number generators are made identical across nodes. Such a scheme is referred to as *synchronized particle filtering* [34]. By initializing the random number generator with the same seed, all nodes can be made to generate the same particles, which in turn makes fusion of the associated weights simpler. The communication costs are then limited to the transmission of the associated weights across the whole network.

The immense flexibility of this approach allows for it to be effective in any particle filtering algorithm. However, this freedom in generality comes with associated drawbacks. For one, the stability of the algorithm depends critically on the requirement of synchronized random numbers, which requires that the hardware at each node be the same. Further, this particular way of decentralization, does not efficiently use the processing power of the nodes, as in the end the same computations are performed *identically* at each node.

8.3.2 Distributed function estimation

However, we can relax the need to make our distributed inference algorithm to be *identical* to the centralized one. There are a host of methods that allow for the computation of average mean through explicit global communication or through local consensus [137] [138].

An alternative to the concept of synchronous filtering can be by approximating the inference at each camera with a Gaussian mixture model [124, 133, 55] or in general, any parametric density family. The parameters can then be transmitted to all nodes in the sensor network, each of which locally updates their densities.

8.4 Compressive Sensing

In this dissertation, we argued for the need for models that go beyond simple subspace or sparse models. Towards this end, we proposed the hybrid subspace sparse model, that build on knowledge of a subspace that is representative of the signal we want to sense, and an over-complete dictionary for the residue. However, there is still a disconnect between both data-driven as well as analytic models for visual signals and the ones used in sensing and recovery of signals. As an example, it is well known that natural images exhibit a gradient distribution that is peaked at zero with heavy non Gaussian tails [106]. The Fields of Experts (FOE) [114] model learns filters for a classes of images (or any data) and uses them for image restoration. For reflectance fields, non-linear analytical models such as the Blinn-Phong and Terrance-Cook models have been used to parametrize and compress reflectance fields. Towards this end, signal recovery (from compressive measurements) can benefit significantly by clever use of such models.

One potential way of applying these statistical priors on signals is by modeling the recovery problem as one of Bayesian inference. Given (compressive) measurements $\mathbf{y} = \mathbf{\Phi x} + \mathbf{n}$ and a prior model for the signal $p(\mathbf{x})$, we can aim to recover the posterior,

$$p(\mathbf{x}|\mathbf{y}, \mathbf{\Phi}) \propto p(\mathbf{y}|\mathbf{x}, \mathbf{\Phi})p(\mathbf{x}) \tag{8.1}$$

The Relevance Vector Machine (RVM) [46] computes the posterior efficiently when $\mathbf{x}$ is modeled to be sparse. This forms the basis of much of Bayesian methods for recovery of signals from compressive measurements [67]. The key problem is extending this to alternate signal models such as one of non-Gaussian gradient distribution lies in the inference problem which becomes intractable. Further, the high dimensionality of visual signals makes Monte-Carlo based inference infeasible.

Bibliography

[1] P. Absil, R. Mahony, and R. Sepulchre, "Riemannian geometry of Grassmann manifolds with a view on algorithmic computation," *Acta Applicandae Mathematicae: An International Survey Journal on Applying Mathematics and Mathematical Applications*, vol. 80, no. 2, pp. 199–220, 2004.

[2] S. M. Ali and S. Silvey, "A general class of coefficients of divergence of one distribution from another," *Journal of the Royal Statistical Society*, vol. 28, pp. 131–142, 1966.

[3] K. Åström, "Invariancy methods for points, curves and surfaces in computational vision," Ph.D. dissertation, Department of mathematics, Lund University, Sweden, 1996.

[4] A. Athalye, M. Bolic, S. Hong, and P. M. Djuric, "Generic hardware architectures for sampling and resampling in particle filters," *EURASIP Journal of Applied Signal Processing*, vol. 17, pp. 2888–2902, 2005.

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

[6] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "A simple proof of the restricted isometry property for random matrices," *Constructive Approximation*, vol. 28, no. 3, pp. 253–263, 2008.

[7] R. Baraniuk, "Compressive sensing," *IEEE Signal Processing Magazine*, vol. 24, no. 4, p. 118, 2007.

[8] R. Baraniuk, V. Cevher, M. Duarte, and C. Hegde, "Model-based compressive sensing," *Submitted to IEEE Transactions on Information Theory*, 2008.

[9] R. Basri and D. Jacobs, "Lambertian reflectance and linear subspaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 218–233, February 2003.

[10] E. Begelfor and M. Werman, "How to put probabilities on homographies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1666–1670, 2005.

[11] P. Belhumeur and D. Kriegman, "What Is the Set of Images of an Object Under All Possible Illumination Conditions?" *International Journal of Computer Vision*, vol. 28, no. 3, pp. 245–260, 1998.

[12] A. Ben-Artzi, R. Overbeck, and R. Ramamoorthi, "Real-time BRDF editing in complex lighting," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 945–954, 2006.

[13] J. Bernardo, "Reference posterior distributions for Bayesian inference," *Journal of the Royal Statistical Society. Series B*, vol. 41, pp. 113–147, 1979.

[14] J. Blinn, "Models of light reflection for computer synthesized pictures," *ACM SIGGRAPH Computer Graphics*, vol. 11, no. 2, pp. 192–198, 1977.

[15] M. Bolic, "Architectures for efficient implementation of particle filters," Ph.D. dissertation, Dept. of Electrical Engineering, State University of New York at Stony Brook, August 2004.

[16] M. Bolic, P. M. Djuric, and S. Hong, "Resampling algorithms for particle filters: A computational complexity perspective," *EURASIP Journal of Applied Signal Processing*, no. 15, pp. 2267–2277, 2004.

[17] ——, "Resampling algorithms and architectures for distributed particle filters," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2442–2450, July 2005.

[18] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[19] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[20] E. Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes rendus-Mathématique*, vol. 346, no. 9-10, pp. 589–592, 2008.

[21] E. Candes, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, p. 1207, 2006.

[22] G. Casella and R. Berger, *Statistical inference*. Duxbury Press Belmont, Calif, 2002.

[23] A. Cedilnik, K. Košmelj, and A. Blejec, "The distribution of the ratio of jointly normal variables," *Metodološki zvezki*, vol. 1, no. 1, pp. 99–108, 2004.

[24] ——, "Ratio of two random variables: A note on the existence of its moments," *Metodološki zvezki*, vol. 3, no. 1, pp. 1–7, 2006.

[25] V. Cevher and J. H. McClellan, "General direction-of-arrival tracking with acoustic nodes," *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 1–12, 2005.

[26] ——, "An acoustic multiple target tracker," in *IEEE Statistical Signal Processing Workshop*, Bordeaux, FR, July, 2005.

[27] ——, "Proposal strategies for joint state-space tracking with particle filters," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Philadelphia, PA, March 2005.

[28] ——, "Fast initialization of particle filters using a modified Metropolis-Hastings algorithm: Mode-Hungry approach," in *IEEE International Conference on Acoustic Speech and Signal Processing*, Montreal, CA, May 2004.

[29] V. Cevher, A. C. Sankaranarayanan, J. H. McClellan, and R. Chellappa, "Target tracking using a joint acoustic video system," *IEEE Transactions on Multimedia*, vol. 9, no. 4, pp. 715–727, June 2007.

[30] N. Checka, K. Wilson, M. Siracusa, and T. Darrell, "Multiple person and speaker activity tracking with a particle filter," in *Proceedings of the International Conference of Acoustics, Speech and Signal Processing*, vol. 5, Orlando, FL, May 2004, pp. 881–884.

[31] R. Chellappa, G. Qian, and Q. Zheng, "Vehicle detection and tracking using acoustic and video sensors," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Montreal, CA, May 2004.

[32] S. Chib and E. Greenberg, "Understanding the metropolis hastings algorithm," *American Statistician*, vol. 49, pp. 327–335, 1995.

[33] Y. Chikuse, *Statistics on special manifolds*. Springer Verlag, 2003.

[34] M. Coates, "Distributed particle filters for sensor networks," in *Proceedings of the International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, April 2004, pp. 99–107.

[35] R. Cook and K. Torrance, "A reflectance model for computer graphics," *Proceedings of the 8th annual conference on Computer graphics and interactive techniques*, pp. 307–316, 1981.

[36] A. Criminisi, *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. Springer, 2001.

[37] A. Criminisi, I. Reid, and A. Zisserman, "Single view metrology," *Int. J. Comput. Vision*, vol. 40, no. 2, pp. 123–148, 2000.

[38] A. P. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of Royal Statistical Society, Series B*, vol. 39, pp. 1–38, 1977.

[39] D. Donoho, "Compressed Sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[40] ——, "For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 6, p. 797, 2006.

[41] A. Doucet, "On sequential simulation-based methods for bayesian filtering," Department of Engineering, University of Cambridge, Tech. Rep. CUED/F-INFENF/TR.310, 1998.

[42] A. Doucet, N. Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[43] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and Computing*, vol. 10, pp. 197–208, 2000.

[44] M. Duarte, M. Davenport, D. Takbar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk, "Single-Pixel Imaging via Compressive Sampling [Building simpler, smaller, and less-expensive digital cameras]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[45] Y. Eldar and M. Mishali, "Robust recovery of signals from a union of subspaces," *preprint*, 2008.

[46] A. Faul and M. Tipping, "Analysis of sparse Bayesian learning," *Advances in Neural Information Processing Systems*, vol. 1, pp. 383–390, 2002.

[47] H. Fillbrandt and K. H. Kraiss, "Tracking people on the ground plane of a cluttered scene with a single camera," *WSEAS Transactions on Information Science and Applications*, vol. 2, pp. 1302–1311, 2005.

[48] F. Fleuret, J. Berclaz, and R. Lengagne, "Multi-camera people tracking with a probabilistic occupancy map," *Technical Report EPFL/CVLAB2006.07*, July 2006.

[49] G. D. Forney Jr, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[50] W. Forstner, "Uncertainty and Projective Geometry," *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics, Springer*, 2004.

[51] D. Gatica-Perez, G. Lathoud, I. McCowan, J.-M. Odobez, and D. Moore, "Audio-visual speaker tracking with importance particle filters," in *Proceedings of the International Conference on Image Processing*, Barcelona, Spain, September 2003.

[52] R. C. Geary, "The frequency distribution of the quotient of two normal variates," *Journal of the Royal Statistical Society*, vol. 93, no. 3, pp. 442–446, 1930.

[53] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions and bayesian restoration of images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[54] A. Georghiades, D. Kriegman, and P. Belhumeur, "Illumination Cones for Recognition under Variable Lighting: Faces," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 52–59, 1998.

[55] Z. Ghahramani and M. Beal, "Variational inference for Bayesian mixtures of factor analysers," *Advances in Neural Information Processing Systems*, vol. 12, pp. 449–455, 2000.

[56] N. Gordon, D. Salmon, and A. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *IEE Proceedings of Radar and Signal Processing*, vol. 140, pp. 107–113, 1993.

[57] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A high-performance, portable implementation of the MPI message passing interface standard," *Parallel Computing*, vol. 22, no. 6, pp. 789–828, Sep. 1996.

[58] W. D. Gropp and E. Lusk, *User's Guide for* `mpich`*, a Portable Implementation of MPI*, Mathematics and Computer Science Division, Argonne National Laboratory, 1996, aNL-96/6.

[59] G. Haas, L. Bain, and C. Antle, "Inference for the cauchy distribution based on maximum likelihood estimators," *Biometrika*, vol. 57, no. 2, pp. 403–408, August 1970.

[60] R. Hartley and A. Zisserman, *Multiple View Geometry in computer vision*. Cambridge University Press, 2003.

[61] R. Hartley and P. Sturm, "Triangulation," *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146–157, 1997.

[62] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[63] P. J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, pp. 73–101, March 1964.

[64] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *European Conference on Computer Vision*, Cambridge, UK, April 1996, pp. 343–356.

[65] ——, *Active Contours*. Springer, 2000.

[66] A. D. Jepson, D. J. Fleet, and T. El-Maraghi, "Robust online appearance model for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1296–1311, Oct. 1998.

[67] S. Ji, Y. Xue, and L. Carin, "Bayesian Compressive Sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[68] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing: Concepts and Techniques*. Prentice Hall, 1993.

[69] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–481, March 2000.

[70] S. V. P. Kabal, "Detection of signals by information theoretic criteria," *IEEE Trans. on Signal Processing*, vol. 52, pp. 1171–1178, May 2004.

[71] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[72] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier Science Inc. New York, NY, USA, 1996.

[73] S. M. Khan and M. Shah, "A multi-view approach to tracking people in crowded scenes using a planar homography constraint," *European Conference on Computer Vision*, vol. 4, pp. 133–146, May 2006.

[74] Z. Khan, T. Balch, and F. Dellaert, "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, 2005.

[75] K. Kim and L. S. Davis, "Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering," *European Conference on Computer Vision*, vol. 3, pp. 98–109, May 2006.

[76] J. Koenderink, A. Van Doorn, and M. Stavridi, "Bidirectional reflection distribution function expressed in terms of surface scattering modes," *Lecture Notes in Computer Science*, pp. 28–39, 1996.

[77] P. Lalonde and A. Fournier, "A wavelet representation of reflectance functions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 4, pp. 329–336, 1997.

[78] J. Lawrence, A. Ben-Artzi, C. DeCoro, W. Matusik, H. Pfister, R. Ramamoorthi, and S. Rusinkiewicz, "Inverse shade trees for non-parametric material representation and editing," *Proceedings of ACM SIGGRAPH 2006*, vol. 25, no. 3, pp. 735–745, 2006.

[79] K. Lee, J. Ho, and D. Kriegman, "Nine Points of Light: Acquiring Subspaces for Face Recognition under Variable Lighting," in *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, HI, December 2001.

[80] I. Leichter, M. Lindenbaum, and E. Rivlin, "A probabilistic framework for combining tracking algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, June 2004.

[81] T. K. Leung and J. Malik, "Detecting, localizing and grouping repeated scene elements from an image," in *European Conference on Computer Vision*, Cambridge, UK, April 1996, pp. 546–555.

[82] B. Li and R. Chellappa, "A generic approach to simultaneous tracking and verification in video," *IEEE Transactions on Image Processing*, vol. 11, pp. 530–544, May 2002.

[83] M. R. Liggings, C. Y. Chong, I. Kadar, M. G. Alford, V. Vannicola, and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, vol. 85, pp. 95–107, Jan. 1997.

[84] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, September 1998.

[85] J. S. Liu, R. Chen, and T. Logvinenko, "A theoretical framework for sequential importance sampling with resampling," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. Springer-Verlag New York, 2001.

[86] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An invitation to 3D vision, from images to models*. Springer Verlag, 2003.

[87] S. Mallat and S. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[88] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[89] G. Marsaglia, "Ratio of normal variables and ratios of sums of uniform variables," *Journal of American Statistical Association*, vol. 60, no. 309, pp. 193–204, March 1965.

[90] ——, "Ratio of normal variables," *Journal of Statistical Software*, vol. 16, no. 4, May 2006.

[91] W. Matusik, H. Pfister, M. Brand, and L. McMillan, "A data-driven reflectance model," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 759–769, 2003.

[92] S. J. Maybank, "Probabilistic analysis of the application of the cross ratio to model based vision," *International Journal of Computer Vision*, vol. 16, no. 1, pp. 5–33, 1995.

[93] ——, "The fisher-rao metric for projective transformations of the line," *International Journal on Computer Vision*, vol. 63, no. 3, pp. 191–206, 2005.

[94] K. L. Mengerson and R. L. Tweedie, "Rates of convergence of the hastings and metropolis algorithms," *The Annals of Statistics*, vol. 24, no. 1, pp. 101–121, February 1996.

[95] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1091, 1953.

[96] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[97] F. Nicodemus, "Directional reflectance and emissivity of an opaque surface," *Applied optics*, vol. 4, no. 7, pp. 767–773, 1965.

[98] B. Ochoa and S. Belongie, "Covariance Propagation for Guided Matching," *Proceedings of the Workshop on Statistical Methods in Multi-Image and Video Processing*, May 2006.

[99] M. Oren and S. Nayar, "Generalization of the Lambertian model and implications for machine vision," *International Journal of Computer Vision*, vol. 14, no. 3, pp. 227–251, 1995.

[100] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation withapplications to wavelet decomposition," in *Asilomar Conference on Signals, Systems and Computers*, 1993, pp. 40–44.

[101] P. Peers and P. Dutré, "Inferring reflectance functions from wavelet noise," *Proceedings of Eurographics Symposium on Rendering*, pp. 173–182, June 2005.

[102] P. Peers, D. Mahajan, B. Lamond, A. Ghosh, W. Matusik, R. Ramamoorthi, and P. Debevec, "Compressive light transport sensing," *ACM Transactions on Graphics*, vol. 28, no. 1, pp. 1–18, 2009.

[103] T. Pham-Gia, N. Turkkan, and E. Marchand, "Density of the Ratio of Two Normal Random Variables and Applications," *Communications in Statistics: Theory and Methods*, vol. 35, no. 9, pp. 1569–1591, 2006.

[104] B. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.

[105] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–591, 1999.

[106] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli, "Image denoising using scale mixtures of Gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.

[107] G. Qian and R. Chellappa, "Structure from motion using sequential monte carlo methods," *International Journal of Computer Vision*, vol. 50, no. 1, pp. 5–31, 2004.

[108] G. Qian, R. Chellappa, and Q. Zheng, "Spatial Self-Calibration of Distributed Cameras," *Proceedings of Collaborative Technology Alliances Conference - Sensors*, 2003.

[109] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE Signal Processing Magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[110] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications inspeech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[111] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26, pp. 195–239, April 1984.

[112] B. D. Ripley, *Stochastic Simulation*. John Wiley & Sons Inc., 1987.

[113] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer-Verlag New York, 1999.

[114] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, San Diego, CA, 2005.

[115] A. C. Sankaranarayanan and R. Chellappa, "Optimal multi-view fusion of object locations," Copper Mountain, CO, January 2008.

[116] A. C. Sankaranarayanan, R. Chellappa, and A. Srivastava, "Algorithmic and architectural design methodology for particle filters in hardware," in *International Conference on Computer Design*, San Jose, CA, October 2005, pp. 275–280.

[117] A. C. Sankaranarayanan, R. Patro, P. Turaga, A. Varshney, and R. Chellappa, "Modeling and Visualization of Human Activities for Multi-Camera Networks," *EURASIP Journal on Image and Video Processing*, (to appear).

[118] A. C. Sankaranarayanan, A. Srivastava, and R. Chellappa, "Algorithmic and Architectural Optimizations for Computationally Efficient Particle Filtering," *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 737–748, 2008.

[119] A. C. Sankaranarayanan, A. Veeraraghavan, and R. Chellappa, "Object Detection, Tracking and Recognition for Multiple Smart Cameras," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1606–1624, 2008.

[120] F. Schaffalitzky and A. Zisserman, "Geometric grouping of repeated elements within images," in *Shape, Contour and Grouping in Computer Vision*, 1999, pp. 165–181.

[121] Y. Y. Schechner, S. K. Nayar, and P. N. Belhumeur, "Multiplexing for optimal lighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1339–1354, 2007.

[122] P. Sen and S. Darabi, "Compressive Dual Photography," in *Computer Graphics Forum*, vol. 28, no. 2, 2009, pp. 609–618.

[123] A. W. Senior, A. Hampapur, and M. Lu, "Acquiring multi-scale images by pan-tilt-zoom control and automatic multi-camera calibration," in *Workshop on Application of Computer Vision*, Berkenridge, CO, January 2005, pp. 433–438.

[124] X. Sheng, Y. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," in *Proceedings of the International Symposium on Information Processing in Sensor Networks*, Los Angeles, CA, April 2005.

[125] S. N. Sinha and M. Pollefeys, "Pan-tilt-zoom camera calibration and high-resolution mosaic generation," *Computer Vision and Image Understanding*, vol. 103, no. 3, pp. 170–183, 2006.

[126] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous Robot Vehicles*, vol. 1, pp. 167–193, 1990.

[127] R. Smith and P. Cheeseman, "On the Representation and Estimation of Spatial Uncertainty," *The International Journal of Robotics Research*, vol. 5, no. 4, p. 56, 1986.

[128] C. Stauffer and K. Tieu, "Automated multi-camera planar tracking correspondence modeling," in *IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, June 2003.

[129] M. Tipping, "Sparse bayesian learning and the relevance vector machine," *The Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[130] H. Tjelmeland, "Using all metropolis-hastings proposals to estimate mean values." Department of Mathematical Sciences, Norwegian University of Science and Technology,, Tech. Rep., 2004.

[131] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan, "The unscented particle filter," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR380, August 2000.

[132] J. Vermaak, M. Gangnet, A. Blake, and P. Perez, "Sequential monte carlo fusion of sound and vision for speaker tracking," in *International Conference on Computer Vision*, Vancouver, Canada, July 2001.

[133] M. Wainwright, *Graphical Models, Exponential Families, and Variational Inference.* Now Publishers, 2008.

[134] M. Wax and T. Kaliath, "Detection of signals by information theoretic criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, pp. 387–392, April 1985.

[135] S. Westin, J. Arvo, and K. Torrance, "Predicting reflectance functions from complex surfaces," *Proceedings of the Conference on Computer graphics and Interactive Techniques*, pp. 255–264, July 1992.

[136] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[137] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[138] L. Xiao, S. Boyd, and S. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[139] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 1208–1221, 2004.

[140] ——, "Tracking multiple humans in crowded environment," *IEEE Conference on Computer Vision*, vol. 2, pp. 406–413, June 2004.

[141] S. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing*, vol. 11, pp. 1434–1456, November 2004.

[142] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing*, vol. 13, pp. 1491–1506, November 2004.

[143] Y. Zhou, P. Yip, and H. Leung, "Tracking the direction-of-arrival of multiple moving targets by passive arrays: Algorithm," *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2655–2666, October 1999.