

**Coping with Uncertain Dynamics in Visual Tracking:
Redundant State Models and Discrete Search Methods**

by

Leonid Taycher

Submitted to the Department of Electrical Engineering and Computer
Science

in partial fulfillment of the requirements for the degree of

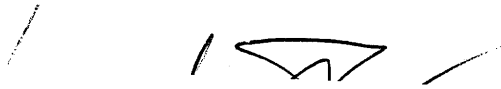
Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2006

© Massachusetts Institute of Technology 2006. All rights reserved.



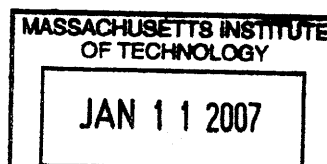
Author
Department of Electrical Engineering and Computer Science
July 27, 2006



Certified by
Trevor J. Darrell
Associate Professor
Thesis Supervisor



Accepted by
Arthur C. Smith
Chairman, Department Committee on Graduate Students



ARCHIVES

Coping with Uncertain Dynamics in Visual Tracking: Redundant State Models and Discrete Search Methods

by
Leonid Taycher

Submitted to the Department of Electrical Engineering and Computer Science
on July 27, 2006, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

A model of the world dynamics is a vital part of any tracking algorithm. The observed world can exhibit multiple complex dynamics at different spatio-temporal scales. Faithfully modeling all motion constraints in a computationally efficient manner may be too complicated or completely impossible. Resorting to use of approximate motion models complicates tracking by making it less robust to unmodeled noise and increasing running times.

We propose two complimentary approaches to tracking with approximate dynamic models in a probabilistic setting. The Redundant State Multi-Chain Model formalism described in the first part of the thesis allows combining multiple weak motion models, each representing a particular aspect of overall dynamic, in a cooperative manner to improve state estimates. This is applicable, in particular, to hierarchical machine vision systems that combine trackers at several spatio-temporal scales. In the second part of the dissertation, we propose supplementing exploration of the continuous likelihood surface with the discrete search in a fixed set of points distributed through the state space.

We demonstrate the utility of these approaches on a range of machine vision problems: adaptive background subtraction, structure from motion estimation, and articulated body tracking.

Thesis Supervisor: Trevor J. Darrell
Title: Associate Professor

Acknowledgments

There are many people without whom this thesis would have never been written. My advisor, Trevor Darrell, allowed me to seek my own way and, at the same time, helped me to avoid many obstacles I would not have seen myself. I would like to also thank him for his skill in showing program committees errors of their ways. Trevor gathered in his group some of the best students in the lab. These years would not have been the same without Gregory Shakhnarovich, David Demirdjian, Ariadna Quattoni, Mike Siracusa, and Ali Rahimi. I have greatly benefited from collaboration with Gregory and David and a significant part of this dissertation would not have happened without them. Talking and working with John Fisher was intellectually stimulating and trying to catch him in the lab provided me with some badly needed physical exercise.

Finally, I would like to thank my parents, Samuil and Liliya, who taught me to love learning and not to be seduced by paths of the least resistance. They, along with my sister Lena, and my friends, Leo, Marat, and Lena were instrumental in preserving my sanity in the last seven years.

Contents

1	Introduction	15
1.1	Tracking, Search and Memory of Things Past	16
1.2	Probabilistic Models for Tracking	17
1.2.1	Generative Models	18
1.2.2	Conditional Random Field Model	21
1.3	Probabilistic Tracking and Search	22
1.4	Redundant State Multi-Chain Models	23
1.5	Tracking Using Discrete Search	24
1.6	Summary of Contributions	25
1.7	Structure of the Dissertation	26
2	Background and Prior Work	27
2.1	Monolithic Generative Models	27
2.1.1	Kalman Filtering	27
2.1.2	Non-Parametric Methods	29
2.2	Tracking with Structured State Representations	31
2.3	Hierarchical Processing in Vision Applications	34
2.3.1	Incorporating Higher-level Knowledge into Feature Extractors	35
2.4	Cooperative Tracking Approaches	37
2.5	Classification and Product Models	38
2.6	Biological Vision Systems	39
2.7	Exemplar-based Sequence Analysis	40
2.8	Review of Articulated Tracking Methods	40
2.8.1	Pose-Observation Compatibility Models	41
2.9	Summary	43
3	Redundant State Multi-Chain Model	45
3.1	Graphical Model	46
3.2	Analyzing Approximation Validity	52
3.3	Inference in Redundant-State Models	55
3.4	Batch Optimization Algorithm	58
3.5	Relationship to Turbo Code Decoding Algorithm	59
3.6	Summary	61

4	Applications of RSMCM in Hierarchical Tracking Systems	63
4.1	Applying Redundant State Modeling to Adaptive Background Maintenance	64
4.1.1	Prior Approaches	65
4.1.2	Redundant Model Formulation	66
4.1.3	Implementation and Results	68
4.2	Structure from Motion Estimation	73
4.2.1	Prior Approaches to Structure from Motion Recovery	73
4.2.2	Factorization Algorithm Incorporating Temporal Coherence	74
4.2.3	Point Feature Tracking	75
4.2.4	Implementation	75
4.3	Summary	83
5	Likelihood Sampling and RSMCM for Articulated Body Tracking	85
5.1	Introduction	85
5.2	Tracking with Likelihood Sampling	86
5.2.1	Upper Body Model	86
5.2.2	Likelihood Sampling	89
5.2.3	Proposal Distribution	90
5.2.4	Kinematic Constraints	90
5.2.5	Single Frame Pose Estimation Implementation and Results	92
5.2.6	Pose Propagation Over Time	94
5.3	Tracking in a Redundant State Framework	95
5.4	Summary	96
6	Exploring Complex Distributions with Discrete Search	105
6.1	Introduction	105
6.2	Tracking by Exploring Likelihood Modes	106
6.2.1	Local Optimization	108
6.2.2	Temporal Integration	108
6.3	Tracking with Strong Temporal Constraints and Local Search	110
6.3.1	Probabilistic Model	111
6.3.2	Estimating Continuous Posterior Distribution with Grid Filtering	111
6.4	Implementation	113
6.5	Results	114
6.5.1	Experiments with synthetic data	114
6.5.2	Statistical Analysis of Results	118
6.5.3	Experiments with real data	119
6.5.4	Discussion	119
6.6	Summary	120
7	Conclusions	125
7.1	Contributions	125
7.2	Future Directions	126
A	Proofs of Analysis Theorems	127

List of Figures

1-1	Graphical model corresponding to N -th order Markov chain	19
1-2	First order Markov models with and without explicit instantaneous state variable	20
1-3	Chain-structured Conditional Random Field model	21
1-4	Redundant State Multi-Chain Model	23
2-1	First order dynamic model	28
2-2	A factorial dynamic model	31
2-3	Switching dynamic model	32
2-4	Dynamic model with overcomplete state	33
2-5	Graphical models used in hierarchical sequence processing	34
2-6	An example of the “sleeping man” problem in adaptive background subtraction	36
2-7	Coupled HMM tracking and classification model	37
2-8	A product of HMMs classification model	38
3-1	First order Markov models with an explicit instantaneous state variable	47
3-2	Redundant State Model	47
3-3	Synthetic example of the product model	49
3-4	Synthetic example of the failure of the product model	50
3-5	Variants of the redundant-state model	51
3-6	Graph structures used in inference algorithms in the dual-chain model	56
3-7	Messages exchanged during one approximate filtering iteration	59
3-8	Graphical model representation of turbocode decoding problem	60
3-9	Dual-chain representation of turbocode decoding problem	60
4-1	Graphical models used in hierarchical sequence processing	64
4-2	Combining background maintenance and object tracking models	66
4-3	Fixing “sleeping man” problem	69
4-4	Error types used for evaluating background subtraction algorithms	70
4-5	Background Subtraction Algorithms: Quantitative Comparison	70
4-6	Background Subtraction Algorithms: Qualitative Comparison	71
4-7	Background Subtraction Algorithms: Qualitative Comparison	72
4-8	Prediction errors in structure-from-motion estimation	77
4-9	Minimum eigenvalues of the indicator matrices	78

4-10	Improvement in feature tracking through iterations of coordinate ascent algorithm	79
4-11	Comparison of typical performance of four SFM algorithms	80
4-12	Quantitative comparison of four SFM algorithms	81
4-13	Qualitative performance evaluation of SFM algorithms on a real-life sequence	82
4-14	Module separation in RSMCM	83
5-1	Articulated upper body model	87
5-2	Generative model used for single-frame sampling	88
5-3	Generative appearance model used in defining a proposal distribution	89
5-4	Constraining arm configuration	91
5-5	Processing stages of articulated body likelihood sampling	94
5-6	Sample results of single-frame pose estimation	98
5-7	Single-frame pose estimation failure	98
5-8	Tracking sequence 1 with RSMCM	99
5-9	Tracking sequence 2 with RSMCM	100
5-10	Tracking sequence 3 with RSMCM	101
5-11	Comparing performance of four tracking algorithms on a sample sequence	103
6-1	High-level overview of the ELMO algorithm	107
6-2	Comparing algorithm performance on synthetic sequences	115
6-3	Error statistics for competing articulated tracking algorithms	116
6-4	Distributions of improvements in joint position estimates over competing algorithms	117
6-5	Qualitative performance evaluation on a salute-chest-azumi sequence	121
6-6	Qualitative performance evaluation on a gesture sequence 1.	122
6-7	Qualitative performance evaluation on a gesture sequence 2.	123
6-8	Qualitative performance evaluation on a gesture sequence 3.	124

List of Tables

3.1	Summary of random variables and conditional distributions used in this chapter	48
6.1	Average times required for algorithms tested to process a single frame.	118
6.2	Confidence intervals for median error reduction, with $p = 0.001$	119

List of Algorithms

1	Recursive Belief Propagation Algorithm for Filtering in a Redundant State Model	58
2	Coordinate Ascent for Batch Optimization in a Redundant State Model . . .	61
3	Sampling based articulated pose tracking	95
4	Tracking Articulated Body by Exploring Likelihood Modes	108

Chapter 1

Introduction

As computers are becoming a ubiquitous part of our environment, simplifying their interaction with us and the world around us becomes more and more important. Before acting upon the world, a computer (or robot) first needs to perceive it. Right now the world has to be severely instrumented with relatively noise-free devices such as keyboards, mice, and RFID tags, to facilitate a computer’s perception of it. This makes interaction with computers unnatural for humans¹, and impossible for non-instrumented parts of the world.

Humans, on the other hand, are well adapted for long-range perception of the surroundings via such “natural” (but noisy) senses as sight and hearing. We are adept at extracting the necessary bits from the wealth of information reaching the brain from the eyes and ears. What is most important is our ability to process *sequences* of observations. The world around us is not static, and seeing it in motion provides more information than any single one, or a collection of random images. Contemporary computers lack this ability: existing models of sufficient complexity are too hard to specify (or train); simpler models commonly used in practice are weak in a sense that they are able to characterize only few aspects of our knowledge about the world. This causes two major problems: significant computational resources are required to extract necessary information with sufficient accuracy; unmodeled behaviors may confuse the model and result in incorrect estimates.

In this dissertation we explore two approaches to improving tracking efficiency and precision. The first approach is motivated by biological visual systems and uses multiple simple models to track complex behaviors. Each weak model expressing a particular facet of behavior can be independently designed with a corresponding efficient inference algorithm. We describe how to combine multiple constraints encoded by weak models by incorporating them into a single redundant-state framework and propose efficient inference algorithms for the resulting model. These algorithms are based on the inference methods developed for constituent models with minimal overhead required for model fusion, making the overall inference efficient but are more robust to errors that are due to unmodeled behaviors. Our second approach is motivated by advances in the available computing power and our ability to assemble vast databases of images labeled with corresponding information of interest. We will demonstrate how such a database can be leveraged to improve running times of weak models’ inference algorithms by performing bulk of the

¹Though the definition of “natural” is changing fast

computations offline.

We demonstrate the performance of these two approaches in a wide range of applications from improving performance of adapting background subtraction algorithms to articulated body tracking.

1.1 Tracking, Search and Memory of Things Past

Visual tracking can be loosely defined as the task of extracting and maintaining some form of knowledge about the evolving world from sequences of images. It is rarely possible, and just as rarely necessary, to completely model the state of the observed world.² We would like to keep as little information as necessary to perform a specific task. For example, in a world consisting of vehicles moving on the road it is not necessary to spend memory and processing time modeling operation of the internal combustion engine in order to track the cars passing by. Having completely different models for red trucks and white cars might also be excessive, especially if we do not know what kinds of cars are going to appear. A relatively simple model of patches with slowly changing appearance passing across a slowly changing background might suffice in this case. On the other hand, having too little information is also not acceptable – tracking a person using only his position is not very useful for sign language gesture recognition tasks.

One can think of tracking as finding the sequence of instantaneous world states (*configurations*) of the model that best correspond to the sequence of observations. Even if we are interested in a very limited amount of information, it may be too complicated or even impossible to obtain these configurations from each image independently. It is often computationally simpler to combine information from multiple images. Many existing tracking approaches are optimization-based: at every new frame they search for the best configuration corresponding to the given frame starting at the configuration most likely based on previous observations.³ An extension to this approach is to search for a configuration that corresponds to the current observation and is compatible with state estimates at previous times.

Before such algorithms can be implemented, four questions have to be answered

- How is the configuration (state of the world) represented?
- How is the compatibility between a configuration and the observation computed?
- How is the new configuration (starting point for the search) predicted from previous observations?
- How is the search performed?

These questions cannot and should not be answered separately. The state representation should on the one hand be useful for higher-level inference and on the other make computing observation compatibility and forward prediction easier. The compatibility function

²The only complete model of the world is the world itself

³Here we conveniently ignore the question of initialization – finding the configuration at the first frame

should be simple enough to be computed efficiently, but robust to noise and to parts of the observation that are unmodeled by the state. The prediction function should be accurate, but not too complicated. The search should be robust to errors in compatibility and prediction functions but fast.

Each prediction function expresses some constraints on the configuration evolution in time, and different constraints may be easy to express using different configuration representations. For example representing a human body pose as a set of joint angles makes it very simple to express a constraint that connected segments of the body do not separate under normal circumstances. Expressing the tendency to move the hands linearly [72] is hard in terms of joint angles, but simple when the pose is represented by rigid poses of individual segments [23, 101, 112]. Unfortunately enforcing connectivity preservation is complicated for the set-of-rigid-poses representation.

If we are interested only in the final state estimate, it is not strictly required that the function being optimized is identical to the one we are interested in, as long as the locations of their extrema are the same. Commonly one is interested in computationally viable functions and commits to a single representation *and* wants prediction and compatibility functions to be simple to evaluate. This requirement often leads to severe approximations to both component functions, and can make the solutions to the final optimization problem sensitive not only on the system of interest but also to the unmodeled parts of the observations. The deterministic search algorithms may thus converge to incorrect solution, and propagating the wrong solution through time results in losing track. This has led the tracking community to adopt *probabilistic* approaches, which produce a *probability distribution* over possible values of the maximum of the objective function rather than a single value at every frame.

1.2 Probabilistic Models for Tracking

Probabilistic frameworks commonly used for tracking share the structure with the deterministic methods. They treat state variable as latent (unobserved) and images as observed variables. These models also specify the temporal relationship between state values at different time steps, and the compatibility between state and observation at the same time step. In contrast to the deterministic tracking, the state is a *random* variable in probabilistic models, and the constituent dependencies are stochastic functions. The primary goal is not computing the best value of the state variable but rather its full distribution function.

The models can be either learned or specified in an ad-hoc fashion, but lack of training data and general computational complexity they are always only approximations to true models of the world. Thus even when exact inference is performed in a probabilistic model, the resulting state distribution may not correspond to the true one. When using a particular inference algorithm in a particular probabilistic model, one has to distinguish between two sources of errors, those resulting from the approximations involved in specifying the model, and those that arise from the algorithm itself.

In this section we review two classes of dynamic probabilistic models commonly used in tracking, without specifying particular inference algorithms. The algorithm selection is strongly influenced by the application area and will be addressed in the next chapter.

1.2.1 Generative Models

The probabilistic *generative models*, are widely used in tracking applications as they explicitly describe how a state can be generated from the values of the state at previous time steps, and how an observation is generated from the internal state. An implicit assumption in this class of models is that the state variable contains all available information about the current observation that can be obtained from observations at other time steps. That is the current observation is independent from the rest conditioned on the value of the state.

Formally, a dynamic generative model is described by specifying two models:

- **Rendering (or emission, or forward) model**, which describes how an observation—an image in vision applications—can be generated from a particular value of the state.
- **Transition (or evolution) model**, which describes how the state evolves in time. This models allows us to take advantage of the fact that images we see come from a sequence and not a random collection.

Both models are almost never absolutely precise. Since the state does not contain all information about the world, specifying a complete forward model is impossible. It is also unnecessary, since even much simpler models can result in good state estimates. Transition functions are also not exact. We rarely know completely how to correctly predict the new value of the state from its previous values, and even when we do, it is not always computationally viable. As a result both transition and emission models are *stochastic* and state becomes a *random* rather than a deterministic variable. That is, we admit from the beginning that we can never know the exact value of the state, but can estimate its distribution, and then use it for reasoning about the world.

To use the generative models in computer programs we encode the state Θ^t , where superscript indicates the time, as a vector of numbers

$$\Theta^t = (\theta_1^t, \theta_2^t, \dots, \theta_N^t)^T.$$

Transition and emission models are encoded as stochastic functions

$$\begin{aligned} \Theta^t &= f(\Theta^0, \Theta^1, \dots, \Theta^{t-1}, \omega^t), & \omega^t &\sim p_\omega \\ I^t &= r(\Theta^t, \eta), & \eta^t &\sim p_\eta \end{aligned} \tag{1.1}$$

where $f(\cdot)$ is a transition function, $r(\cdot)$ is an emission (rendering) function, and ω^t and η^t are random variables that are included to account for incomplete modeling of the world.

A graphical model corresponding to this generative model is shown in Figure 1-1. The shaded nodes (I^\bullet) are observed, clear nodes (Θ^\bullet) are latent, and the edges in the graph define the dependencies between the variables. General graphical model formalisms are described, for example, in [54].

The conditional distributions $p(\Theta^t | \Theta^0, \Theta^1, \dots, \Theta^{t-1})$ and $p(I^t | \Theta^t)$ can be derived from Equations 1.1, and the graphical model encodes a factorization of the joint probability

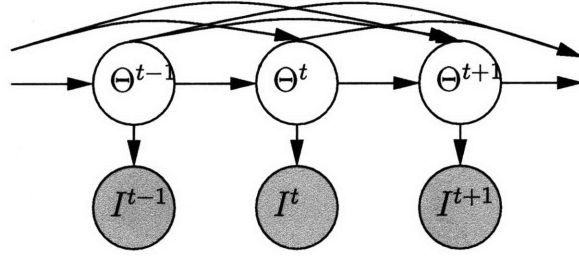


Figure 1-1: Probabilistic graphical model corresponding to the generative model in Equation 1.1. The observed image I^t is stochastically generated from the latent state Θ^t . The value of the state depends on previous values $\Theta^0, \Theta^1, \dots, \Theta^{t-1}$.

density

$$p(\Theta^0, \Theta^1, \dots, \Theta^t, I^0, I^1, \dots, I^t) = \prod_{\tau=1}^t p(I^\tau | \Theta^\tau) p(\Theta^\tau | \Theta^0, \Theta^1, \dots, \Theta^{\tau-1}) \quad (1.2)$$

Inferring the state of the world in this model amounts to computing the posterior distributions of Θ^t for every time step t . In classical tracking the posterior is conditioned on all images observed up to time t :

$$p(\Theta^t | I^0, I^1, \dots, I^t).$$

In batch applications (e.g. shape-from-motion) the state is conditioned on *all* available observations from time 0 to time T :

$$p(\Theta^t | I^0, I^1, \dots, I^T).$$

Intermediate algorithms, that are able to “look” a few time steps into the future have also been implemented. They amount to computing (for k -step look-ahead) a posterior distribution:

$$p(\Theta^t | I^0, I^1, \dots, I^{t+k}).$$

All of these distribution can be derived from Equation 1.2.

Inferring the distributions of latent nodes in this graph is made complicated by the density of the dependence structure. When the current value of the state depends only on a fixed number of the previous values,

$$f(\Theta^0, \Theta^1, \dots, \Theta^{t-1}, \omega^t) = f(\Theta^{t-N}, \Theta^{t-N+1}, \dots, \Theta^{t-1}, \omega^t)$$

then this model becomes an n -th order Markov chain, and can be converted to a first-order model shown in Figure 1-2(a) [2]. This involves introducing a “superstate” S^t , which contains all information about previous states necessary to generate Θ^t :

$$\begin{aligned} S^t &= F(S^{t-1}, \omega_S^t), & \omega_S^t &\sim p_{\omega_S} \\ \Theta^t &= G(S^t, \omega_\Theta^t), & \omega_\Theta^t &\sim p_{\omega_\Theta} \\ I^t &= r(\Theta^t, \eta^t), & \eta^t &\sim p_\eta \end{aligned} \quad (1.3)$$

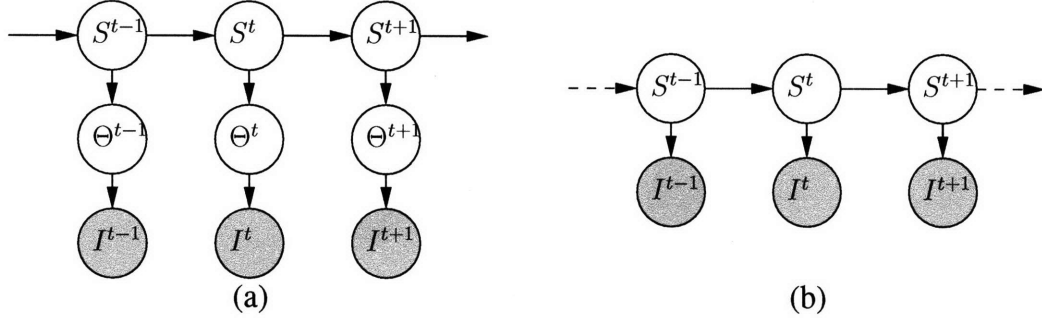


Figure 1-2: First order Markov models of the world evolution. (a) A two stage model corresponding to the Equation 1.3. The observed image I^t is stochastically generated from the latent instantaneous state Θ^t . The value of the instantaneous state is (stochastically) generated from the temporal state S^t , which depends on the it value S^{t-1} at the previous time step. (b) A variant of the model corresponding to Equation 1.4. The instantaneous state was integrated out according to Equation 1.5

where the evolution of S^t is encoded by the first order transition function $F(\cdot)$, and the $G(\cdot)$ is a function that extracts information about Θ^t from the superstate.

In the following discussion we refer to Θ as an *instantaneous* state and to S as a temporal state or, when there is no danger of misperception, as simply state. When performing inference in this model, we need to compute a joint probability distribution of S and Θ . In many models Θ is an element of S , or can be computed deterministically from it. In this case Θ can be completely removed from the model

$$\begin{aligned} S^t &= F(S^{t-1}, \omega_S^t), & \omega_S^t &\sim p_{\omega_S} \\ I^t &= R(S^t, \nu^t), & \nu^t &\sim p_{\nu} \end{aligned} \quad (1.4)$$

where $R(\cdot)$ incorporates $G(\cdot)$ and $r(\cdot)$. From the stochastic perspective it amounts to integrating Θ^t from the distribution $p(I^t, \Theta^t | S^t)$

$$p(I^t | S^t) = \int p(I^t, \Theta^t | S^t) d\Theta^t = \int p(I^t | \Theta^t) p(\Theta^t | S^t) d\Theta^t \quad (1.5)$$

The simplified model is shown in Figure 1-2(b).

Equation 1.4 defines two probability distributions $p(S^t | S^{t-1})$ and $p(I^t | S^t)$. In keeping with the standard notation, we will refer to $p(I^t | S^t)$ seen as a function of S^t as *likelihood*. The posterior distribution of the state at time t is proportional to the product of the likelihood and the *temporal prior*. The temporal prior is a distribution that encodes the knowledge about the current state available for the observations other than the current one. For the online (filtering) case, the temporal prior is defined by Chapman-Kolmogorov equation [79]

$$p(S^t | I^{1..t-1}) = \int p(S^t | S^{t-1}) p(S^{t-1} | I^{1..t-1}) dS^{t-1}. \quad (1.6)$$

Similar definitions exist for batch and look-ahead priors.

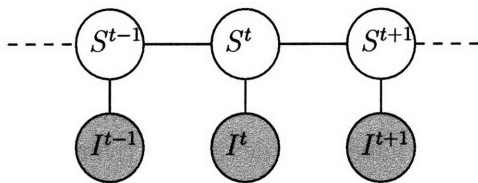


Figure 1-3: Chain-structured Conditional Random Field model. The state of the world at time t is specified by S^t , and the observed image by I^t . The model is described by motion compatibility (potential) function $\phi(S^t, S^{t-1})$ and the image compatibility function $\phi_o^t(S^t) = \phi(I^t, S^t)$.

1.2.2 Conditional Random Field Model

An alternative to a generative model is a Conditional Random Field (CRF) model that replaces transition and emission models by a more general *compatibility* functions. These functions can be more efficient to evaluate than likelihood or transition function, resulting in faster inference algorithms. While one could use techniques such as Gibbs Sampling to generate observations from a CRF, it is not a proper generative model since it does not explicitly state how the observations are generated from the hidden states.

A chain version of a CRF is shown in Figure 1-3. While, apart from the lack of arrows, it is quite similar to the generative model, the underlying computations are quite different. This model is specified by the motion potential $\phi(S^t, S^{t-1})$ and the observation potential $\phi_o^t(S^t) = \phi(I^t, S^t)$. The observation potential function is the measure of compatibility between the latent state and the observation. Of course, one choice for it might be the generative model's emission probability $p(I^t|S^t)$, but this does not have to be the case. It can be modeled by any function that is large when the latent state corresponds to the one observed in the image and small otherwise.

Rather than modeling the joint distribution of poses and observations, the CRF directly models the distribution of poses conditioned on observation,

$$p(S^{0..T}|I^{1..T}) = \frac{1}{Z} p(S^0) \prod_{t=1}^T [\phi(S^t, S^{t-1}) \phi_o^t(S^t)], \quad (1.7)$$

where Z is a normalization constant.

Once the observation potential is defined, a chain-structured CRF can be used to perform on-line tracking

$$p(S^t|I^{1..t}) \propto \phi_o^t(S^t) \int \phi(S^t, S^{t-1}) p(S^{t-1}|I^{1..t-1}) dS^{t-1}. \quad (1.8)$$

Similarly to generative model formulation, the expressions for conditioning on different sets of observations can be derived.

The main advantage of this model from the implementation standpoint is that the observation potential $\phi_o^t(S^t)$ may be significantly simpler to learn and faster to evaluate than the emission probability $p(I^t|S^t)$.

1.3 Probabilistic Tracking and Search

The models described in Equations 1.4 and 1.7 have been used in a large number of contexts and standard inference methods have been developed over the past several decades.

The particulars of vision problems define unique conditions on constituent elements of this model. Computing emission probabilities usually involves a comparison between a relatively large number of pixels generated by the rendering model and the observed image. This results in likelihoods that are sharply peaked, which for reasons described later, makes the shape of the posterior similar to the shape of the likelihood. Likelihoods are also often multimodal (cf. [102]) with large number of outliers due to approximations involved in the rendering function, and, in some cases, sensor noise.

On the other hand, the transition probability models involved in computing the temporal priors are quite uncertain due to the properties discussed in the previous section. In order for such models to be useful in tracking (i.e. for the posterior estimates to be close to the true underlying distributions), the temporal priors should assign significant probability to the regions that would be assigned significant probability by the true prior. This can be achieved by compensating the uncertainty in the transition function by inflating the dynamic noise, making the temporal prior wide (i.e. assigning significant probabilities to a large region in the state space that includes the true prior's coverage). In this case the prior does not provide a lot of smoothing, but rather serves in a data association function by identifying which peaks in the likelihood come from the actual observation and which are due to noise and/or unmodeled elements in the image generation process.

Taking a search perspective to state estimation, the goal in designing a transition function is twofold:

- Enforce as many constraints as possible to decrease the search region and decrease the probability of observing a spurious likelihood peak.
- Make computing the function as fast as possible to improve processing time.

The complexity of the transition function often depends on a particular parameterization of the state. The transition functions that are complicated in one parameterization can be made simpler in another. For example, a non-linear transition function can be linearized by a (non-linear) change in the state representation. Furthermore, there might not exist a single representation in which all constraints are easily describable.

Wide priors complicate tracking for two main reasons. First, posterior distribution estimation often involves searching for the likelihood peaks, and the likelihood (and the posterior) is close to zero everywhere else. The prior defines a search region and a wide prior implies longer search times. The second problem is that the likelihood peaks may be dense, and a wide prior may cover (i.e. assign significant probability) to more than one (correct) peak, thus making the prior's selectivity quite poor. Both of these issues may result in posterior estimates that differ significantly from the true posterior and consequent loss of track.

The first complication has more impact in applications with high-dimensional state spaces such as articulated body tracking and are less significant for simpler trackers since search in low-dimensional spaces can be made efficient. The second problem has more

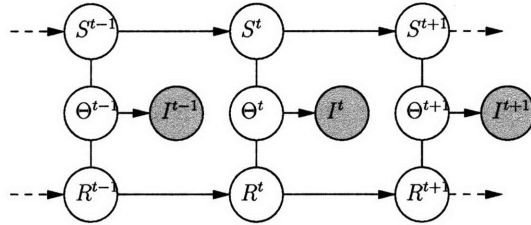


Figure 1-4: Redundant State Multi-Chain Model. This model is created by combining two generative models that describe the evolution of the instantaneous state Θ using different parameterizations of the state (S and R) and the propagation and instantaneous state generation functions.

bearing on the low- and mid-level vision modules, especially in hierarchical systems, that need to be fast and cannot expend time necessary for maintaining and propagating a multi-modal posterior.

1.4 Redundant State Multi-Chain Models

One can approach the problem of disambiguating likelihood peaks in different ways. Early methods chose the most likely mode and ignored the rest. Alternatively, one could preprocess the image in order to remove the extra likelihood peaks. The common practice over the last decade has been to defer the decision until the modes can be disambiguated [19, 36]. This is achieved by maintaining and propagating a multi-modal posterior distribution in the expectation that the correct peak can be detected from future observations. This strategy has a significant computational cost, as propagating multi-modal distributions is much more expensive than the ones with a single mode. It might not be possible to maintain this distribution indefinitely using finite resources [58].

In this thesis we explore a new approach to improving precision of the posterior estimates without explicit introduction of overly complex priors. The key to our method is the observation that it is not necessary to commit to a single representation. We propose maintaining multiple representations of the state simultaneously, propagating each one through an appropriate approximate transition function and reconciling them at every step with each other and with the observation.

If considered from the search point of view, each imprecise temporal prior is generated by a transition function which involves different approximations. In turn, every approximation requires dynamic noise with different properties in order to ensure that it fully covers the true region of interest. All imprecise priors will cover the true region, but if the approximations are sufficiently dissimilar, they will place the rest of the probability mass into distinct regions in the state space as we will show in section 3.2. It is therefore reasonable to perform the search only in the region to which *all* approximate temporal priors assign significant probability.

From a density estimation perspective, we would like to introduce a new prior distribution over the instantaneous states that assigns large probability to the region that is covered

by all individual priors and a small probability to the rest of the state space. This can be done by taking a product of the constituent priors and *scaling* it so that the new prior integrates to unity. Combining generative models into a single system by forming products of the temporal priors results in a model shown in Figure 1-4 which we call a *Redundant State Multi-Chain Model* (RSMCM) framework.

This approach of modeling a complex distribution as a product of simple ones is motivated by product models used for classification in [43] and [11]. In contrast to those methods we are not interested in *training* the product models, but rather in finding conditions under which combining existing models via a product leads to the improved posterior estimations and the efficient inference algorithms for the product models.

Combining models expressing similar constraints into a product model is ill advised, as it would emphasize these constraints more than necessary, making the overall model more certain and sharpening the prior in the incorrect place. In the following chapters we will quantify this intuition and make it more precise.

We will show that combining models that describe evolution of the world at different spatio-temporal scales into a RSMCM has an interpretation of introducing a feedback link into commonly used hierarchical tracking systems in a principled way. This feedback connection has been postulated (and anatomical evidence found) to be present in biological vision systems, and has been found to be beneficial in artificial ones as well.

We have designed and implemented inference algorithms for RSMCM that take full advantage of the inference algorithms that have been designed for individual models. We will demonstrate how a modular implementation of the practical systems is possible, allowing independent design of constituent subsystems.

1.5 Tracking Using Discrete Search

The method proposed in the previous section combines complimentary weak models to reduce the influence of “noise” likelihood peaks on the posterior estimates.

It does not address the second major problem of the probabilistic search – the sheer amount of computation that is needed to explore the region of the state space covered (i.e. assigned significant probability) by the wide temporal prior. In the second part of this dissertation we propose a solution to this problem – quickly exploring large regions of the state space by using a large, precomputed database of possible observations labeled with corresponding state values. We will refer to this database as a “grid” without making any claims about the regularity in the values of the labels.

The key to our approach is to (partly) replace slow exploration of the continuous pose-space used by state-of-the-art methods by fast search on the discrete grid. The straightforward approach to using the grid is to linearly search it for the pose that optimizes the objective function (matching the current observation and conforming to temporal constraints); it has two drawbacks. If the grid is large enough to cover the region of interest in pose space with sufficient density, linear search can still be slow. The search can also be imprecise if the distances between individual grid points is large so that no point lies near the peak of the objective function.

We propose complimentary approaches to overcoming these drawbacks, resulting in

two grid-based tracking algorithms described in this paper. The first algorithm we propose does not assume strong temporal constraints and searches the whole database at every time step by using an efficient *approximate* search algorithms [96]. The results are then refined by local, search in continuous pose space around the proposed grid points, resulting in a mixture-of-Gaussians approximation to the likelihood function. Finally, the approximate likelihood is combined with the temporal prior to obtain a posterior estimate. In addition to rectifying possible errors in the approximate search, the refinement step is convenient for dealing with sparseness of the grid – it would locate the peak of the compatibility function if the initial grid point lies in its general vicinity.

When the temporal constraints are strong, they can be used to restrict the number of grid points that need to be considered at every time step. In this case exact search can be performed on this limited subset yielding the sparse HMM framework given by our second algorithm. If the grid is dense enough, the pose can be estimated directly from results of the search. Otherwise an additional local refinement step can be added.

1.6 Summary of Contributions

In this dissertation we propose two methods for exploiting weak temporal models for tracking complex scenes.

Our first contribution, Redundant State Multi-Chain model, reduces the effect of the noise likelihood peaks on the posterior estimates by pooling the behavior constraints expressed by multiple weak models into a unified probabilistic model. The approach is motivated by the observation that models describing the same set of observations implicitly marginalize over an intermediate feature representation between state and observation. By making the feature representation explicit we obtain a straight-forward and computationally efficient means of mediating between the constituent models. The resulting fused model has a clear probabilistic interpretation, reconciling multiple generative models that describe the same observations, each corresponding to a particular set of independence assumptions and dynamical model. An integrated model is enabled by the introduction of an explicit latent appearance model: this model is desirable for reasons of global consistency; however, exact inference on the resulting combined model is complicated by the introduction of loops. We propose two methods for adapting algorithms designed for constituent modules to operate in a combined system. Both algorithms are based on Belief Propagation updates on acyclic subgraphs of the loopy graph. The online filtering algorithm operates on single-frame slices of the temporal chain. The batch inference algorithm uses single-chain based subgraphs. We demonstrate advantages of the RSMC models on background maintenance [109] and structure-from-motion estimation tasks [39]. In both cases using RSMCM implementation resulted in significant reduction in errors over regular feed-forward systems.

Our second major contribution addresses the computational complexity of search constrained by a weak temporal prior such as in articulated body tracking. One approach to dealing with this problem is likelihood sampling, where a sample-based representation of the likelihood function is constructed from low-level features extracted from each frame. Tracking is performed using late temporal integration of single-frame pose estimates. This method is successful when the constituent feature detectors (and/or trackers) are available.

To deal with more general cases we propose using large predefined sets (grids) of likely observations labeled with corresponding state values to efficiently explore state-observation compatibility surfaces in large volumes of state space. This is achieved by supplementing optimization in continuous space by search in the grids. We propose two complimentary methods of utilizing discrete search for density estimation. The first method is useful in the presence of very weak dynamic models and searches the whole grid at every time step with an efficient *approximate* search algorithm [96]. The results are then refined by local continuous-space search around grid points, and temporal integration is done as the last step in the algorithm [25]. The second method uses stronger dynamic model to determine a region of the grid where a linear search is performed. The state distribution can then be estimated directly from the search results [118]. When applied to the task of articulated body pose estimation, both algorithms produce accurate results outperforming state-of-the-art methods. The local search algorithm, furthermore, can be implemented to operate in real time.

1.7 Structure of the Dissertation

The rest of this thesis is structured as follows: in Chapter 2 we discuss previous probabilistic approaches to visual tracking including both models and inference methods. We derive the Redundant State Multi-Chain Model, describe the conditions under which different models should be combined into RSMCM, and present on-line and batch inference algorithms in Chapter 3. We then describe applications of RSMCM to real-world vision tasks: improving performance of hierarchical vision systems such as object tracking with adaptive background subtraction and structure-from-motion-estimation from feature tracks in Chapter 4.

Chapter 5 introduces likelihood sampling method with applications to articulated body tracking. This method is restricted to cases when body parts (in particular head and hands) can be detected or tracked in every frame. We show how pose estimation can be made more robust by incorporating body pose and constituent part trackers into a redundant state model.

We turn to grid-based techniques in Chapter 6 and describe two tracking algorithms based on discrete search: Exploring Likelihood MOdes (ELMO) and Conditional Random People (CRP). We present results of these algorithms on articulated body tracking without any part trackers necessary as a preprocessing step.

Discussion of the contributions' impact and outline of the possible future developments are presented in chapter 7.

Chapter 2

Background and Prior Work

In this chapter we review existing probabilistic approaches to state-space sequence processing, with examples drawn (when possible) from applications in computer vision. We first describe models that do not assume any structure of the state and the generic inference methods that have been proposed for these models. We then turn to structured generative models that make specific assumptions about the state space. These assumptions may lead to decreasing computational requirements, or to better precision in state estimation, or both.

In discussing generative models, we will focus on three distinguishing characteristics – the state probabilistic density representation, possible transition functions, and the applicable inference methods.

2.1 Monolithic Generative Models

Monolithic models represent the full joint distribution of all parameters, and are used in cases when the state vector and the transition model do not have an identifiable structure. The graphical model for such a system is shown in the Figure 2-1. The choice among monolithic systems can be narrowed down further depending on what is known about the likelihood function. If it is known to be unimodal, then, generally, the posterior state distribution would also be unimodal. In this case parametric models are preferable, as they have closed-form solutions and can be efficiently updated. When the likelihood function can be multimodal, maintaining parametric representation of the posterior can quickly become unmanageable, and the semi- or non-parametric methods are used.

2.1.1 Kalman Filtering

Simple approximations have been shown to be sufficient for many applications where the state distribution and evolution models have low complexity. If it is reasonable to assume that the transition function can be approximated as linear and the transition noise is Gaussian,

$$S^t = AS^{t-1} + \omega, \quad \omega \sim N(0, \Sigma_\omega)$$

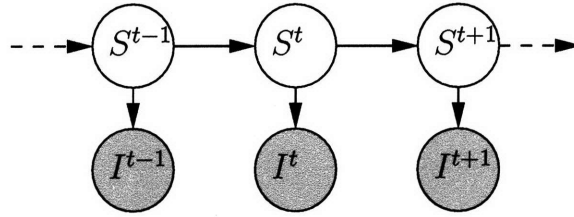


Figure 2-1: First order dynamic model with monolithic state. The model is specified by the transition probability $p(S^t|S^{t-1})$ and emission probability $p(I^t|S^t)$

then the state distribution can also be modeled with a single Gaussian.¹ Under these assumptions closed-form solutions are available for both filtering, known as Kalman Filter [56], and smoothing – Rauch-Tung-Striebel (RTS) algorithm [89].

These algorithms have been used for vision applications such as feature-point tracking [91], tracking independently moving multiple objects [137], and in early approaches to articulated pose tracking [92].

The simplicity of observation and dynamic models that allows for efficient implementation of a Kalman filter are also its main drawbacks when used in tracking applications. The linear approximation to the dynamics can be inappropriate for some tasks. A large number of tracking problems also require modeling multimodal posterior distributions that cannot be represented with a single Gaussian.

Non-Linear Gaussian Filtering

In many applications, while the state distribution may be reasonably approximated with a normal distribution, the (constant) linear approximation to transition function is grossly invalid. Several extensions to Kalman filtering concepts have been proposed to deal with these situations. In the Extended Kalman filter [6] F is linearized around the current state estimate to obtain a (linear) approximation $A(S^{t-1})$, which is then used in the standard Kalman update.

$$S^{t-1} \approx A(S^{t-1})S^{t-1} + \omega, \quad A(S^{t-1}) = J(f)|_{S^{t-1}, \omega} \sim N(\omega; 0; \Sigma_\omega)$$

This method has been extensively used for articulated tracking purposes [127, 71] and in structure from motion estimation [52, 6].

An alternative method for propagating Gaussian densities through non-linear functions was proposed in [55]. Rather than approximating the transition function, the Unscented Kalman filter deterministically samples points from the distribution, propagates them through the full transition function and re-estimates the output Gaussian distribution from propagated samples. The propagated density is then used in the standard Kalman update. This algorithm has been applied to articulated body tracking in [110].

While these methods allow modeling non-linear dynamics, they still use a unimodal

¹This is true only when the prior on the initial state is Gaussian and the emission model is linear-Gaussian, but suitable approximations can usually be made for unimodal but non-Gaussian likelihood models

Gaussian approximation to the posterior distribution. This makes them inappropriate for more complicated tracking tasks that require propagating multimodal posterior distributions.

2.1.2 Non-Parametric Methods

When it can no longer be assumed that the posterior distribution would remain unimodal, simple models, such as a Gaussian, become inappropriate. In general, an arbitrary distribution can be approximated to a good precision with a mixture-of-Gaussians, but the number of components in the mixture may be large. The mixtures can be easily propagated through the transition function with methods similar to those in the previous sections. Unfortunately, the number of Gaussians in the mixture has a capacity to grow geometrically with every update. While they can be combined, and their number reduced after the update [13], this approach has been generally discarded in favor of non-parametric sample-based methods [66].

The sample-based methods are based on the fact that expectation of any function can be computed using a large number of samples drawn from the distribution.

$$E_x[f(x)] = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=1}^N f(x_i), \quad \text{where } x_i \sim p(x) \text{ i.i.d} \quad (2.1)$$

At the heart of any non-parametric density propagation approach lies a particular method of maintaining and propagating a fair sample from the state posterior distribution at every time step.

Particle Filtering

Particle filtering makes use of the importance sampling technique to create a sample-based representation of the posterior distribution at every time step. Rather than drawing samples directly from the posterior, the samples are drawn from a simpler *proposal* distribution. Each sample (particle) is then assigned a weight equal to the ratio of the posterior and the proposal distributions evaluated at it.

The simplest implementation of this idea is the CONDENSATION algorithm [48], which has been extensively used for visual tracking tasks [48, 50, 113, 99, 98]. In CONDENSATION the temporal prior is used as a proposal distribution. The posterior at the previous time-step is sampled with replacement according to the particles' weights, each sample is then propagated through the transition function, and dynamic noise is added. Each particle is then assigned a weight equal to its likelihood (proportional to the ratio of the posterior and the prior).

The advantages of this method are its simplicity and ability to use any of the transition functions. Its major disadvantage is the deterioration of performance in high dimensions [104]. Approximate knowledge of the transition function requires inflating dynamic noise in order to ensure that the temporal prior (the proposal distribution) sufficiently covers the region where the true prior would be high. This also ensures that the temporal prior covers

a large region of the state space which requires a large number of samples to make computation with finite number of samples approximate that in Equation 2.1. CONDENSATION can also suffer from problems related to the use of pseudo-random number generators for sampling [58]. The combination of these issues can lead to sample impoverishment, where a large number of samples are concentrated at a single mode of the posterior and do not represent the full distribution. Using a large number of particles increases running time of the algorithm, since particle filtering requires evaluating likelihood for every particle in the set.

When the high-dimensional state evolves in a low-dimensional subspace, it is possible to use CONDENSATION -style algorithms with better effect by sampling in the subspace rather than in the full state space [129].

More complicated particle filtering algorithms [49, 102, 104] use likelihood as well as temporal priors to create a proposal distribution that better approximates the posterior. These algorithms have better theoretical properties, allow representation of the posterior with a smaller number of particles, and have better running times. They are, however, more difficult to implement.

Problems with pseudo-random number generators have been addressed by using deterministic, quasi-random sampling [82, 78]. These methods do not suffer from the degeneracies faced by the regular particle filters, but can still be subject to sample impoverishment as they do not account for the multiple samples corresponding to the same point in state space.

Markov-Chain Monte Carlo

A more general, albeit generally slower approach to tracking using sample-based distribution is Markov-chain Monte Carlo (MCMC) family of algorithms [66]. MCMC methods construct a Markov chain that has the target distribution as its stationary distribution. This chain is then simulated until it reaches the equilibrium distribution (in reality, this is only done for a fixed number of steps). States generated by the Markov Chain after this “burn-in period” are fair samples from the target distribution. The transition probabilities in the Markov chain can be implemented as a random walk using Metropolis-Hastings algorithm [66], or take into account gradient of the target density (so-called Hybrid Monte Carlo [16]).

Well designed MCMC methods tend to be more accurate than particle filters such as CONDENSATION allowing for better exploration of sample space and are not susceptible to sample impoverishment. However, they are also relatively slow and harder to use (e.g. detecting when the Markov chain has reached the equilibrium is not easy). While MCMC-based methods have been applied to articulated pose estimation [107, 16], multi-body tracking [57], and shape-from-motion estimation [22], they are not yet widely used in vision community due to difficulties in implementation and computational complexity.

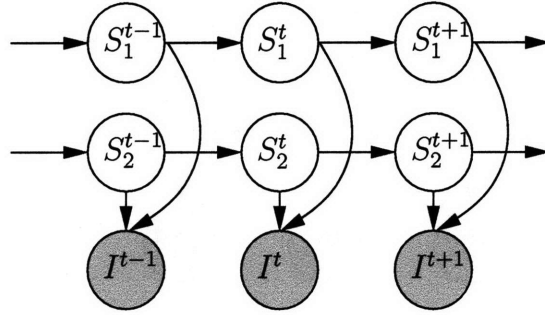


Figure 2-2: A first order factorial dynamic model. The state S^t is partitioned into subsets S_1^t and S_2^t , with factored evolution model $p(S_1^t, S_2^t | S_1^{t-1}, S_2^{t-1}) = p(S_1^t | S_1^{t-1})p(S_2^t | S_2^{t-1})$, and the observation is generated according to emission probability is $p(I^t | S_1^t, S_2^t)$

2.2 Tracking with Structured State Representations

The posterior estimates may be made more tractable and transition models more precise by introducing structure into the state. The resulting graphical model no longer has a simple chain structure of an HMM, but becomes a more general Dynamic Bayesian Network (DBN) (cf. Figures 2-2 and 2-3). The increase in the complexity of the model (graph) structure is compensated for by reducing complexity of the individual conditional distributions. The main advantage of the structured models from the computational point of view is the ability to approximate the full posterior distribution of the state by the product of marginal distributions of the partitions. This reduces the effective dimensionality of the space where the distributions have to be maintained. For sample based models this reduces the necessary number of samples to be evaluated.

Factorial Models

Several different names have been assigned to the technique of partitioning the state into non-interacting substates that combine to generate the observation (Figure 2-2). When the state is discrete the approach is known as a Factorial Hidden Markov Model [37]. The fully sample-based method has been presented as Partitioned Sampling [65, 27]. The hybrid models where part of the state is represented with a parametric distribution and part with sample-based are known as Rao-Blackwellized Particle filters [29].

In this model the transition probability is approximated by the product of transition probabilities for each substate, but the observation is still generated based on all of the subsets.

$$\begin{aligned} p(S^t | S^{t-1}) &= p(S_1^t, S_2^t | S_1^{t-1}, S_2^{t-1}) = p(S_1^t | S_1^{t-1})p(S_2^t | S_2^{t-1}) \\ p(I^t | S^t) &= p(I^t | S_1^t, S_2^t) \end{aligned} \quad (2.2)$$

Even though the transition probability can be factored, the individual substates are coupled through the observed image and are *not* independent. The key to success of the factorial models is the fact that the factored approximation to the posterior can indeed be maintained without inflating the errors caused by this approximation [8]. Factorial mod-

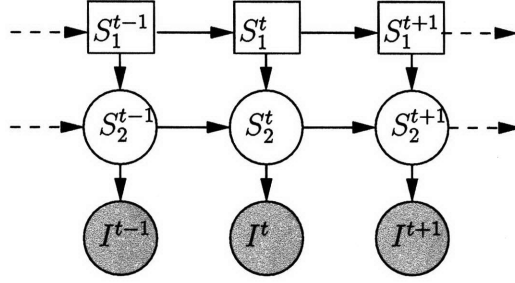


Figure 2-3: A first order switching dynamic model. The state S^t is partitioned into a discrete “activity” meta-variable S_1^t and continuous pose S_2^t . The activity variable evolves independently, $p(S_1^t|S_1^{t-1}, S_2^{t-1}) = p(S_1^t|S_1^{t-1})$, and the pose evolution depends on activity. The full transition probability is $p(S_1^t, S_2^t|S_1^{t-1}, S_2^{t-1}) = p(S_1^t|S_1^{t-1})p(S_2^t|S_1^t, S_2^{t-1})$, and the observation is generated only based on the pose $p(I^t|S_1^t, S_2^t) = p(I^t|S_2^t)$

els have been used for articulated body [65] and multi-object [51] tracking applications in computer vision. The multi-object tracking application has been successful, since the main assumption that observed objects move independently is borne out in practice. This assumption is questionable in human body tracking, since evolution of joint angles in the whole body is often strongly correlated.

Switching and Mixture Models

An alternative to fixed dynamic models used in the above models is to incorporate some kind of meta-information into the state, and use different dynamics based on this meta-information (Figure 2-3). The approach proposed in [80] partitions the state S^t into “activity” (S_1^t) and “pose” (S_2^t). The activity evolves independently, while the pose transition model depends on the current activity. The resulting model is

$$p(S_1^t, S_2^t|S_1^{t-1}, S_2^{t-1}) = p(S_1^t|S_1^{t-1})p(S_2^t|S_1^t, S_2^{t-1}). \quad (2.3)$$

A marginal distribution of the pose,

$$p(S_2^t|S_1^{t-1}, S_2^{t-1}) = \sum_i p(S_1^t = i, S_2^t|S_1^{t-1}, S_2^{t-1}) = \sum_i p(S_1^t = i|S_1^{t-1})p(S_2^t|S_1^t = i, S_2^{t-1}),$$

is a mixture of different behaviors where mixture coefficients are the probabilities of the corresponding activities.

A similar model has been used in [1]. In that case the discrete variable determined the “class” of a pose, and depended not on the previous class as in Figure 2-3 but rather on the previous pose.

These models are at their best when they need to represent a small number of simple possible behaviors. The behavior models compete instead of cooperating, so more complicated motions can be difficult to model.

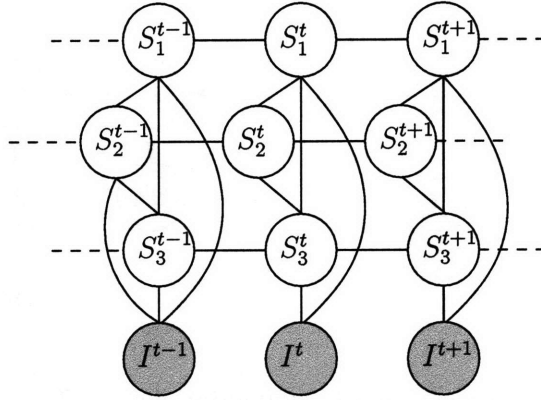


Figure 2-4: Dynamic model with overcomplete state. The state has an overcomplete representation $S^t = (S_1^t, S_2^t, S_3^t)$. Dependencies between subsets are introduced in order to ensure that the state is valid. The graphical model is undirected in order to correctly represent a dependency structure. The transition probability can be factored into the product of individual substate transitions $p(S_1^t, S_2^t, S_3^t | S_1^{t-1}, S_2^{t-1}, S_3^{t-1}) = p(S_1^t | S_1^{t-1})p(S_2^t | S_2^{t-1})p(S_3^t | S_3^{t-1})$

Overcomplete Models and Non-parametric Belief Propagation

In overcomplete models the state is represented with a set of dependent random variables whose number is greater than the actual number of degrees of freedom. These models have been used for articulated body tracking with each variable representing a rigid pose of a particular limb [112, 101]. Compared to the standard approach of representing body poses with a set of joint angles, this representation allows using simpler emission models – each limb can be rendered individually, although occlusions have to be accounted for. The relationships between sub-states (poses) have to become more complicated, in order to ensure that they represent valid articulated poses (e.g., the forearms do not become detached from the shoulders). The graphical model for this type of system is shown in Figure 2-4.

Inference in these models is complicated due to the presence of loops, even for online (filtering) tasks. The key observation that allows for tractable inference is that we can approximate the joint posterior distribution of the nodes by the product of marginals

$$p(S^t | I^{1..t}) \approx \prod_{i=1}^N p(S_i^t | I^{1..t}).$$

The same approximation was used for inference in the factorial models.

An important tool for computing marginal posterior distribution in the graphical models is Belief Propagation algorithm [81]. The marginals are computed in a distributed fashion by passing messages that contain partial *beliefs* between the nodes in the graph. It has been proven that Belief Propagation can compute exact distributions on tree-shaped graphs and both empirical evidence and theoretical results [134] suggest that it can produce good results on loopy graphs as well.

When distributions at the nodes are Gaussian, and the dependencies are linear-Gaussian,

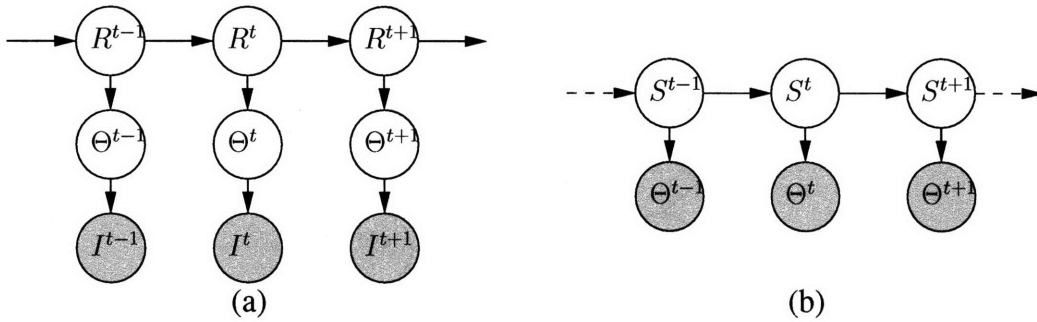


Figure 2-5: Hierarchical processing of sequential data. The marginal distributions $p(\Theta^t | I^{0..t})$ are computed using the low-level feature extractor (a). These marginals are then used as input to high-level algorithm (b).

then the messages can be computed in closed-form. As we have seen, this is not the case for many real-life applications, and so a non-parametric version of Belief Propagation [111, 47] have been used in articulated body tracking applications [112, 101].

2.3 Hierarchical Processing in Vision Applications

The generative models described in the previous sections are all defined as generating the image I^t . This is convenient for defining models, but in reality complete rendering is rarely used for modeling emission probabilities. High level models, e.g., articulated and multi-body tracking and structure from motion estimation, generate “feature” representations, such as edge pixels, feature point locations, foreground labels, etc. These features are then compared to those extracted from the observed images and the likelihood is computed using some kind of similarity metric.

This approach can be mapped to the proposed probabilistic model in a straight-forward way if the features are extracted for each individual frame independently (e.g., edges, color segmentations, etc). When the feature extractor is itself a sequential processor, for example an adaptive background subtraction algorithm [84, 40, 109], the inference procedure should be described using two models shown in Figure 2-5.

These models define, in effect, two different dynamics on the intermediate “feature” variables. One can be derived from the feature extractor, and is usually local. The other is from the high-level global model. The fact that both of them are used, makes it clear that the independence assumptions encoded in each model (i.e., that the features are independent, conditioned on the states) are false.

Low-level algorithms tend to ignore global spatial relationships by modeling the evolution of each image patch (in feature extraction [109, 120]) or object (in object tracking [77]) as independent, and compensating for it with restrictive assumptions about the local behavior of the scene. For example, feature-point trackers usually assume that the image patch about the point of interest has a relatively stable appearance, and adaptive background subtraction modules typically assume that the foreground object does not remain stationary for extended periods of time. When these assumptions are violated, the resulting errors

are passed to higher-level modules, and these are not always able to correct them (e.g., the so-called “sleeping man problem”, Figure 2-6).

While in general the posterior estimates produced by a high-level model are still correct, it is advantageous to incorporate both dynamics into a single probabilistic model. Several approaches have been proposed where the full posterior distribution over features is considered in the high-level model [46, 122]. These methods use the uncertainty about a particular feature estimate to decrease its influence on the high-level estimates.

Ad-hoc methods of incorporating feedback from high-level to low-level models have also been proposed [70, 128, 52]. These methods generally use the high-level estimates to validate the output of low-level tracking. The high-level estimates are used to gate feature tracking output in [70]. In [52] feature position is deterministically selected from those predicted by the global and local models but no fusion is performed. The main drawback of all of these methods is the deterministic nature of the feedback mechanism. They do not allow for soft decision making (e.g. deferring the final decision to the future time steps) and are sensitive to the thresholds used to trigger the feedback.

2.3.1 Incorporating Higher-level Knowledge into Feature Extractors

Rather than using feedback from the high-level tracking modules for regularizing feature extraction, one could incorporate higher-level constraints into feature tracking itself. Instead of considering individual feature trackers to be completely independent, which can lead to merged tracks, techniques such as Multiple Hypothesis Tracking (MHT, [19]) or Joint Probabilistic Data Association Filter (JPDAF, [36]) can be incorporated into low-level tracking process [119]. The goal of both of these techniques is to solve data association problem (i.e., which observation corresponds to which tracked point). They assume that individual motions are independent, which is completely incorrect in feature-extraction applications, since features are coherently generated by a single high-level motion.

Several methods have been proposed in the context of background maintenance for motion-independent high-level analysis [41, 123]. The use global (or region) based analysis at a single time point to detect sudden illumination changes and be more robust to sensor noise. These approaches do not use motion information, and thus cannot deal with “sleeping-man”-type problems. This problem is usually approached with ad-hoc heuristics-based methods that disable the background model adaptation in the general vicinity of the detected object. If the “vicinity” broadly defined (e.g., as a bounding box of the detected object), these methods may disable correct adaptation. If, on the other hand, an object shape is used to define non-adapting region, then using adaptive shape models is complicated.

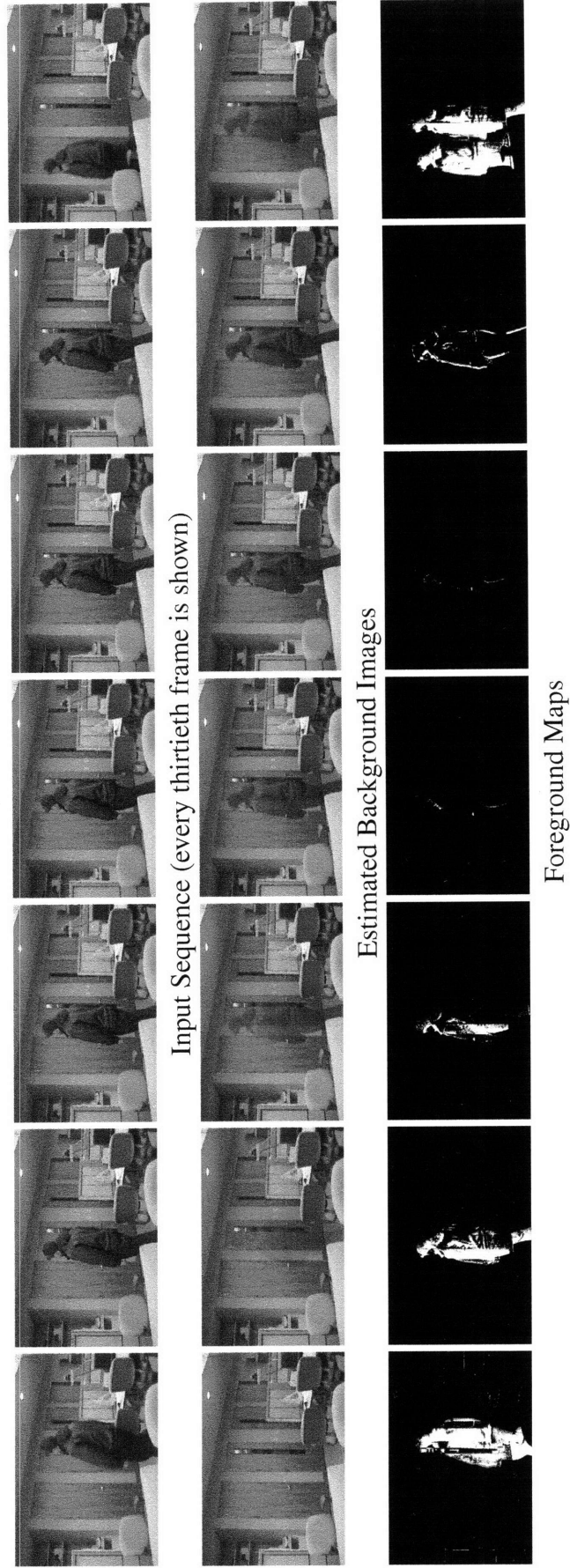


Figure 2-6: An example of the “sleeping man” problem in adaptive background subtraction. Adaptive background maintenance systems make an implicit assumption that foreground objects do not remain stationary. When this is not the case (as in the sequence shown in the top row), the background model (middle row) adapts to motionless foreground objects which then “fade-away” (the computed foreground maps are in the bottom row).

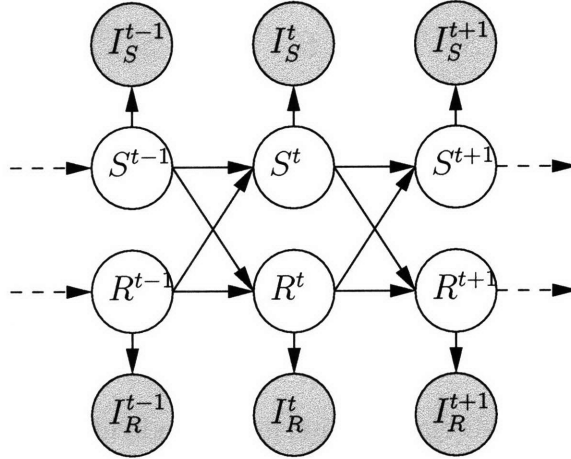


Figure 2-7: Coupled HMM tracking and classification model. States S^t and R^t independently generate observations I_S^t and I_R^t , but the transitions are coupled: $p(S^t, R^t | S^{t-1}, R^{t-1}) = p(S^t | S^{t-1}, R^{t-1})p(R^t | S^{t-1}, R^{t-1})$

2.4 Cooperative Tracking Approaches

Multiple methods have been proposed for improving robustness of tracking results with respect to image noise. One such approach is to use more than one type of features derived from the observed images. In tracking heads, for example, one can take advantage of the edge maps, flesh color detection and optical flow measurements. One could use all of these “independent” measurements simultaneously, by devising an appropriate emission model. Alternatively, a separate tracker can be run using each available modality, with state beliefs reinforcing each other at every step [34, 64, 131, 132].

These approaches are characterized by assuming that features (e.g. color and edges) are independent when conditioned on the state values, and can be tracked using different state representations. The graphical model corresponding to this assumption is shown in Figure 2-7, and is equivalent to the Coupled HMM model proposed by [9]. The output nodes I_1^t and I_2^t , corresponding to different modalities, are assumed to be independent in this model although the reality is that they are tightly coupled. Indeed, they come from the same observation.

Another feature of all of these approaches is that the information exchange between individual trackers is performed on the temporal state probability distribution level. In particular, the scaled products of posterior distributions are formed at every time step in [64, 34]. The ability to form such products relies on the access to “translators”—functions that map between distributions in different state spaces. This requirement complicates the resulting algorithms and makes them less general, since such functions are not always available.

Cue integration approaches differ significantly from the redundant state model proposed in Section 1.4 and described in detail in Chapter 3. The RSMCM is a method for combining multiple dynamic models to interpret a *single* stream of data. The information exchange in this framework is performed at the *shared* instantaneous state level, removing the necessity

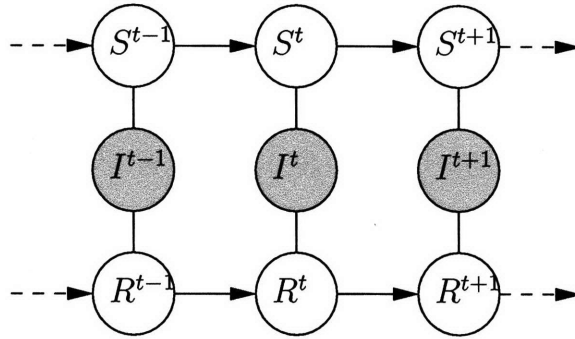


Figure 2-8: A product of HMMs classification model. Individual models are specified, each modeling a particular aspect of the class distribution. The observed sequence is assigned a probability equal to the scaled product of probabilities assigned by each chain.

for explicit translation functions.

2.5 Classification and Product Models

In previous sections we have discussed using probabilistic dynamical models to infer the distributions of the hidden states at every time step. Similar models have been applied to the task of *classifying* sequences. They are trained to assign to each observed sequence a probability of belonging to a particular class.

Most of the models described in the previous sections have been used for classification. Hidden Markov Models (HMMs), in particular, are used extensively in language [67], speech [86] and vision [126]. Parameters of these models are *learned* from the data belonging to a certain class, and are then used to assign a probability of belonging to this class to novel sequences. Classifiers based on individual chain models have also been extended to mixtures of chains, each trained to detect a specific subclass with mixture coefficients set depending on the relative frequency of each subclass.

An alternative method of combining densities to model complicated distributions is to form their (scaled) products. This approach is appealing since it makes inference (classification) easy – as component models are usually simple. A product of experts model was proposed for individual observations' classification [43]. It has been extended to sequence models in Product of HMMs [11] (PoHMMs).

The product model consists of a set of HMMs that are trained jointly, but inference is performed independently. Each model can be thought of as representing a single aspect of the sequence behavior, and their product assigns a large probability only to sequences exhibiting *all* of the individual behaviors. The graphical model for PoHMMs model is shown in Figure 2-8. The joint distribution of an observed sequence and the hidden state sequence for the top chain is

$$p_S(I^{1..T}, S^{0..T}) = p(S^0) \prod_{t=1}^T (p(S^t|S^{t-1})p(I^t|S^t)).$$

By integrating out the state sequence we obtain a marginal density assigned to a sequence $I^{1..T} = (I^1, I^2, \dots, I^T)$

$$p_S(I^{1..T}) = \int p(S^0) \prod_{t=1}^T (p(S^t|S^{t-1})p(I^t|S^t)) dS^0 dS^1 \dots dS^T$$

The observation probability for the bottom chain, $p_R(I^{1..T})$ can be obtained in a similar manner. Combining the probabilities obtained from both chains, the final probability of a sequence is

$$p(I^{1..T}) = \frac{1}{Z} p_S(I^{1..T}) p_R(I^{1..T}) \quad (2.4)$$

where Z is the scaling constant necessary for $p(I^{1..T})$ to integrate to unity.

While inference in the PoHMMs model is straight-forward, training is complicated. The models should be trained to be “independent”, since taking product of similar densities (representing similar constraints) does not add new information to the system, but rather makes the result tighter.

The PoHMMs model cannot be used for tracking directly in a way a factorial model can. In factorial models the individual states (parts of the global state) interact with each other, which allows the model to maintain consistency. In PoHMMs the states are decoupled at run-time and without this interaction their estimates can easily diverge, making the overall estimate meaningless.

2.6 Biological Vision Systems

Early theories of motion processing in the brain [68] proposed a purely feed-forward interpretation similar in spirit to hierarchical bottom-up models described in section 2.3. More recently, however, anatomical evidence of feedback connections in the brain has emerged [108]. A model that simultaneously maintains and propagates multiple coherent representations of motion at different levels in the brain has been proposed in [136].

Motion processing in the brain has been described as inference in a directed hierarchical generative model [87, 88]. Online inference was described as loopy belief propagation but with no symmetry between messages (due to the structure of the model). A computational model of pattern recognition in the brain has been proposed in [63]. While this model describes analyzing static patterns, it does postulate that the feed-forward/feedback nature of the multi-scale processing may be described by message passing in an undirected model.

Our motivation for the redundant state model comes partially from hierarchical feedback models of [136, 63]. In particular the Belief-Propagation based inference algorithm of [63] is quite similar to the one that we will propose for performing on-line state estimation in the redundant state models.

2.7 Exemplar-based Sequence Analysis

The tracking approaches discussed above rely on the analytic representation of the relationship between the state (pose) and the observation. This is a natural point of view in the context of generative models. It is also efficient from the coding standpoint – it takes less memory to store a sequence of states and the observation generating function than to store images themselves. However by using the analytic generation function, a tracking method becomes a subject to the approximation errors in its specification.

The state-observation correspondence function may be alternatively specified as a set of exemplars – image-pose pairs. Several methods that use a database of such exemplars have been proposed. Using exemplars removes the rendering step from the tracking process, making computing observation compatibility computation more efficient and potentially more precise.

Exemplars have been used for tracking by specifying possible transitions between keyframe states [124]. This method requires augmenting the pose-observation compatibility encoded by the exemplars by an analytic transformation function, which limited its usefulness for generic tracking.

Keyframes have also been used as a part of larger system for recovering 3D structure from 2D observations in [28]. In this work the keyframes have provided “anchor points” to which the pose sequence corresponding to observations was fitted.

While designed originally for single-frame pose estimation, nearest neighbors search [96] can also be used for tracking by smoothing the resulting pose estimates.

Rather than operating on single labeled frames and then enforcing temporal continuity, full sequence exemplars have been used for pose estimation and activity recognition in [30].

2.8 Review of Articulated Tracking Methods

In this section we move away from the general problem of state estimation in visual tracking and concentrate on methods used for articulated body tracking. This particular application has been instrumental in bringing a large number of probabilistic techniques to the attention of machine vision community, since it involves operating in a very high-dimensional state space with large number of unobserved degrees of freedom and complicated dynamics.

Tracking in complicated state spaces (e.g., human pose) is naturally approached as a search or optimization problem. The deterministic algorithms optimize a model-based pose-observation correspondence function using local search methods (e.g., gradient descent) [83, 21, 24]. The search is performed at every frame in the sequence starting with the previous frame’s optimum. These methods are computationally efficient, but suffer from local minima and observation ambiguities. They are able to handle relatively small interframe motion and can fail when fast motion or occlusion occurs.

The probabilistic paradigm was introduced to articulated tracking when the need to maintain uncertainty about estimates was recognized. Many of the models described in Chapter 2 were used in the body tracking applications. Early probabilistic human tracking approaches (e.g., [92]) used a Kalman Filtering framework, thus implicitly modeling all

constituent distributions with Gaussians. These assumptions have been later relaxed to account for nonlinear mono-modal dynamics (while still using Gaussian state model) by using Extended [53] or Unscented [110] Kalman Filters. Switched linear systems [80] were proposed to describe arbitrary learned dynamics.

Unimodality assumptions needed to be removed in order for the body trackers to recover from errors caused by projective ambiguities causing multiple peaks in the objective function. This led to introduction of semi- and non-parametric density propagation algorithms such as Multi-Hypothesis Tracking [13] and particle filtering [99]. These approaches use pseudo-random sampling to explore the pose space and evaluate the pose-observation compatibility. The sampling process also incorporates dynamic coherence constraints absent from the deterministic methods, but require large numbers of samples to sufficiently cover the high-dimensional pose space.

While knowledge about the local motion behavior can be incorporated into the sampling process to guide the search [102, 129], local search around each sample is still necessary to improve performance [103, 14]. Gradient information can also be directly incorporated into the sampling process [16]. An alternative approach to search in probabilistic setting is to use simulated annealing where the search is initialized by sampling the temporal prior derived from the results of the search at the previous time step [26]. Partitioned [65] and Layered [113] sampling use factored state dynamics, which allows semi-independent propagation of parts of the state vector.

One could also search for the optima at every frame independently, and incorporate temporal information later. This approach has been implemented in a probabilistic setting by [114, 100]. Bottom-up cues have also been used in [61] in the context of Rao-Blackwellized particle filtering to reduce the number of parameters that need to be sampled. Recently, several approaches have been proposed to learn the (possibly multi-valued) function directly mapping the previous pose estimate and the observation to the new pose estimate [1, 106, 105]. These approaches have been demonstrated to perform well on a limited range of motion but require large amounts of training data.

Content-based retrieval methods using databases of images or image sequences labeled with pose information have been used to perform single-frame pose estimation [96] and action recognition [30]. These algorithms compute descriptors for all labeled exemplars and observations, and use (approximate) k-nearest-neighbors search based on this descriptor to retrieve examples from the database that are then interpreted according to the task.

2.8.1 Pose-Observation Compatibility Models

An important part of any articulated-body tracking system is a observation compatibility function that computes how well a particular pose matches an observed image. Such functions are usually based on *rendering* an image of a human in a specified pose and then comparing it to the image. While the comparison can be performed using raw pixel values [13], features such as edges [44, 26], silhouettes [83, 106], results of color segmentation [127], or learned filter responses [98] are usually employed.

All features described above are inherently image-to-image comparison features. Their usefulness for computing pose-image similarity is limited by the sophistication of the rendering function, and the sensitivity of a particular feature to the pose variation. The ren-

dering function is supposed to be computed for every state value where the likelihood has to be evaluated and the state-of-the-art non-parametric methods require large number of such evaluations; therefore, in order for algorithms to run in reasonable time the rendering function cannot be very sophisticated. Additionally, the features may be relatively insensitive to pose changes: for example, silhouettes do not preserve much information about self-occlusion.

Similarity Sensitive Embedding

Instead of choosing features on an ad-hoc basis, one could *learn* the features that are most sensitive to pose similarity. One method for learning such features is Similarity Sensitive Embedding (SSE) introduced in [96]. The general approach it takes is to use two sets of image pairs – one with images depicting objects (in our case humans) in similar poses, and the other with objects in dissimilar poses – to select features that are likely to have the same value for similar examples and different values for dissimilar ones. The SSE algorithm as described in [96] and used in this thesis operates on a set binary features that are obtained by thresholding bins of edge direction histograms computed over all possible windows in the input image.

The output of the learning stage of the algorithm is a function that maps images into fixed-length binary strings, such that the images depicting people in similar poses are mapped into binary strings that are close in Hamming distance, and images of people in dissimilar poses are mapped to strings that are far. Formally, the algorithm learns a function H ,

$$H : I \rightarrow \{0, 1\}^N.$$

such that with high probability

$$|H(\mathbf{I}(\theta_1)) - H(\mathbf{I}(\theta_2))| \text{ is } \begin{cases} \text{small} & \text{if } \theta_1 \text{ is similar to } \theta_2 \\ \text{large} & \text{otherwise} \end{cases}$$

where $\mathbf{I}(\theta)$ is the image of a person in pose θ . The similarity measured in the embedding space may be different from the distance in the image space which is often dominated by the factors irrelevant to the difference in the parameters of interest.

The similarity sensitive encoding can be used for approximate search in the database, using, e.g., locality sensitive hashing as proposed in [96]. It can also be incorporated into a conventional pose-observation compatibility function by using it as a parameter in the exponential distribution

$$\phi(I, \theta) = \exp\{-\lambda |H(\mathbf{I}(\theta)) - H(I)|\} \quad (2.5)$$

that is large for compatible image/pose pairs and small otherwise.

The distances in the learned embedding space strongly depend on the training image pairs. If the image pairs labeled as similar contain images with large variations in some of the parameters, then the distance will not be sensitive to them. This behavior can occur if the similar pairs are selected automatically using a distance function that that weighs some parameters more than others. For example if the mean distance between selected

joint positions is used (as in [96, 118]), then the distance in the embedding space would be more sensitive to the upper-body pose than to the lower body.

2.9 Summary

In this chapter we have given an overview of probabilistic models that are used for visual tracking. The main source of complexity of the generative models is the lack of precise knowledge about transition probabilities. This necessitates maintaining potentially complex posterior distributions. These distributions can be simplified by introducing extra independence assumptions or by the use of preprocessing techniques that can decrease influence of spurious peaks in the likelihood. In the absence of feedback from high-level modules, simple preprocessing models can themselves be confused by noise and produce incorrect outputs resulting in the erroneous high-level estimates. Feedback connections have been shown to be present in the biological visual systems, and useful in computer vision applications.

Inference methods used in tracking algorithms range from closed-form formulas for Kalman filter variants through non-parametric density propagation in tree-shaped structures (particle filtering and MCMC tracking) to non-parametric Belief Propagation in loopy graphs.

Existing product models are appropriate for classification tasks but are not very useful for tracking since they do not allow information exchange between individual chains at run-time.

Chapter 3

Redundant State Multi-Chain Model

True models of the world’s evolution are seldom fully known and computationally viable true models are almost non-existent. The observed world can be approximately described by many different models at various levels of abstraction and using various representations. For example feature points moving in two dimensions can be thought of as moving independently or as parts of a three dimensional object observed by a moving camera. Cars on the road are autonomous, but their collections (especially in a traffic jam) have properties of a fluid and can be modeled as such.

In this chapter we introduce and describe in detail the first main contribution of this thesis– the Redundant State Multi-Chain Model. This framework allows us to combine existing models of dynamics in a principled way to take advantage of differences in their characterization of observed behavior and of different constraints on the motion they represent. We propose inference methods that take full advantage of efficient algorithms designed to perform inference in constituent models.

Our approach is based on the fact that the evolution of the same set of observations can be tracked using several models that have different states and/or transition models but share an instantaneous state (defined in section 1.2). In this chapter we will assume that each individual model can be used for tracking independently. That is, each would produce posterior estimates that are very similar to the true underlying one in the presence of very low noise in the likelihood function. The noise-free likelihood function in vision tasks is sharply peaked and unimodal under these conditions, and the shape of the posterior is dominated by the shape of the likelihood. The temporal prior serves as a “gating” function (Section 1.3), and since the approximate models are capable of tracking noise-free observations, their temporal priors should always assign non-negligible probability (“cover”) the location of the likelihood peak. It is reasonable to expect that the true underlying prior would cover only the possible locations of the likelihood peak, but approximate priors would assign some probability mass to other regions. In the noise-free case the true and approximate posteriors will be essentially the same. The differences between true and approximate models becomes significant in the case of noisy observations (this is made precise in Section 3.2). The sensor noise (and/or, approximations to the likelihood function) can result in introducing extra peaks into the likelihood mode and/or broadening the true one. Since the approximate priors “cover” larger regions of the state space than the true one, they are more likely to assign non-negligible probability to the region of the noise-related

peak in the likelihood resulting in the posterior estimate that is significantly different from the true posterior. Using the true prior should also produce better estimates in case of the broadened likelihood peak.

An important observation that we will exploit in this chapter is that while all temporal priors cover (i.e. assign significant probability mass to) a region of the state space which the true prior would cover, they are likely to place the rest of the probability mass to *different* regions if the approximations involved in formulating them are sufficiently dissimilar. By taking a product of the priors, the probability mass should be concentrated in the region covered by the true prior, and result in better posterior estimates for the instantaneous state as well as the states of constituent models. We will quantify this intuition later in the chapter. The sharing of the latent instantaneous state variable between the models constitutes the main advantage of RSMCMs over the product models described in section 2.5. It allows information exchange between temporal states of constituent models at every time step and prevents posterior estimates from diverging. Individual chains in the product models are, on the other hand, decoupled during inference making interaction impossible.

For the ease of the argument we will concentrate on the two-chain models in the following discussion, although it will be shown that more than two approximate models can be combined. The rest of this chapter is structured as follows: in section 3.1 we describe RSMCM graphical model and how it can be created based on the individual single-chain dynamic models. We then go on to describe the conditions under which RSMCM produces better results than any of the constituent models. We present the on-line filtering algorithm in Section 3.3 and batch optimization algorithms in Section 3.4. We conclude by discussing the relationship of RSMCM to turbo codes and turbo code decoding algorithm.

3.1 Graphical Model

We wish to use multiple single-chain dynamic models of the form shown in the Figure 3-1 (same as one described in the section 1.2) to define a redundant state multi-chain model. All individual single-chain systems share the same instantaneous state and, consequently, a model for generating observations from the instantaneous state.

The multi-chain system shown in Figure 3-2 is formed from two such models. It consists of two temporal state chains S and R corresponding to each constituent systems' transition models, shared instantaneous states Θ^t , and observations I^t . In contrast to the single-chain models, the multi-chain model is *undirected*¹. The intuition for this is that we would like for the temporal chains to be *independent* if the instantaneous states are known. The potentials of the edges are equal to the conditional probability densities from the original models (e.g., $\phi(S^t, S^{t-1}) = p(S^t|S^{t-1})$, $\phi(\Theta^t, R^t) = p(\Theta^t|R^t)$, etc.). The ambiguity in choosing the potential $\phi(I^t, \Theta^t)$ does not arise, since the conditional density $p(I^t|\Theta^t)$ is conceptually the same in both models. The random variables and conditional distributions used in this chapter are summarized in Table 3.1. To complete the description of the model we need to specify the distribution $p(S^0, R^0)$. We choose to model it as a product $p(S^0)p(R^0)$, where the marginal distributions are the same as those used in the

¹In keeping with notation of [11] we preserve the edge directionality of the single-chain models when it does not affect the independence properties.

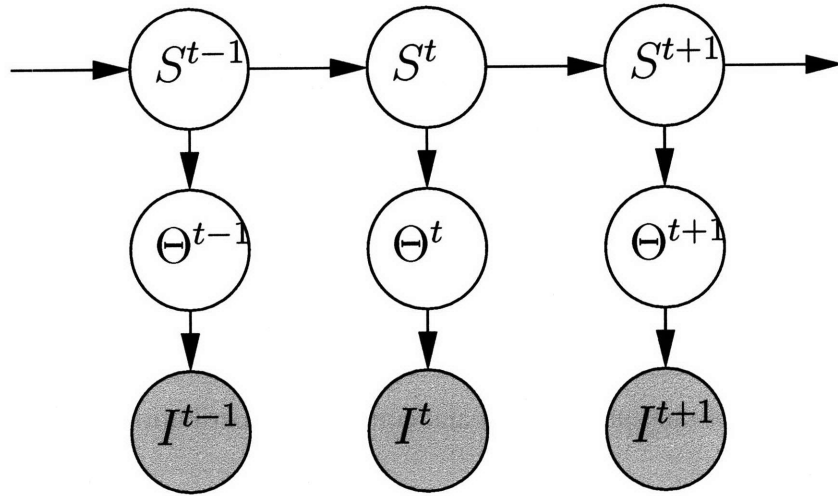


Figure 3-1: A two stage model corresponding to the Equation 1.3. The observed image I^t is stochastically generated from the latent instantaneous state Θ^t . The value of the instantaneous state is (stochastically) generated from the temporal state S^t , which depends on the it value S^{t-1} at the previous time step.

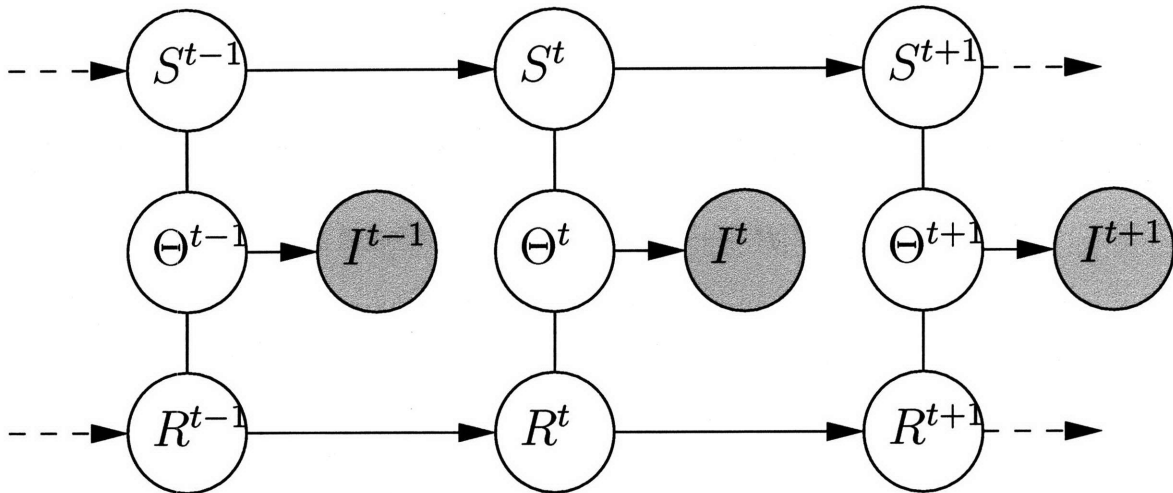


Figure 3-2: A Redundant State Model that incorporates the models with structure shown in Figure 3-1. The model is presented using directed/undirected formalism of [11]. The model is undirected, but the arrows are preserved when they do not affect the conditional independence assumptions. The potential in this undirected model correspond to the conditional probabilities in the individual models (e.g., $\phi(R^t, R^{t-1}) = p(R^t|R^{t-1})$, etc.).

t	- Time index
I^t	- Image observed at time t
I^\bullet	- All images used for inference at time t
$I^{\bullet \setminus t}$	- All images used for inference at time t other than I^t
S^t, R^t	- Temporal states of the constituent chains at time t
$p(S^t S^{t-1}), p(R^t R^{t-1})$	- State evolution models of constituent models
Θ^t	- Latent instantaneous description of the world used by both constituent models
$p(\Theta^t S^t), p(\Theta^t R^t)$	- The instantaneous state generation models in constituent models
$p(I^t \Theta^t)$	- Observation generation model

Table 3.1: Summary of random variables and conditional distributions used in this chapter

constituent models. While R^0 and S^0 are in reality tightly coupled, we chose to make this approximation to simplify inference algorithms.

Based on our choice of potentials, in the absence of all but one of the temporal chains the model reduces back to a standard single-chain structure. The only interaction between the chains can be through the instantaneous state nodes. We will show that these properties of the model enable us to partially utilize single-chain inference methods for inference in RSMC models.

To understand how the individual chains influence each other, it is first instructive to consider what is going on at the instantaneous state level. The posterior distribution of Θ^t is proportional to the product of three functions: the likelihood, and predictions from the top (S) and bottom (R) chains. Let $p_S^t(S^t|I^{\bullet \setminus t})$ be a temporal prior at S^t , where the distribution is conditioned on the observation set appropriate for the particular inference task (e.g. $I^{1..t-1}$ for filtering and $I^{1..t-1, t+1..t+k}$ for look-ahead), and $p_R^t(R^t|I^{\bullet \setminus t})$ be similarly defined. Two corresponding temporal priors for the instantaneous state are then

$$p_S^t(\Theta^t|I^{\bullet \setminus t}) = \int p(\Theta^t|S^t)p_S^t(S^t|I^{\bullet \setminus t})dS^t \text{ and} \quad (3.1)$$

$$p_R^t(\Theta^t|I^{\bullet \setminus t}) = \int p(\Theta^t|R^t)p_R^t(R^t|I^{\bullet \setminus t})dR^t. \quad (3.2)$$

We denote the product of these distributions, rescaled to integrate to unity as the *product temporal prior*,

$$p_*^t(\Theta^t|I^{\bullet \setminus t}) \propto p_S^t(\Theta^t|I^{\bullet \setminus t})p_R^t(\Theta^t|I^{\bullet \setminus t}). \quad (3.3)$$

That is S^t and R^t jointly serve as redundant memory about the past (and/or future) observations. This is very similar to the Product of HMMs [11] formulation, with one extremely important difference – Θ^t is a *latent* rather than observed variable. This allows information to be shared between individual temporal chains during inference, in contrast to the product models in which inference in the component chains is independent. The posterior distribution of the latent instantaneous state is proportional to the product of the likelihood and product prior.

$$p_*(\Theta^t|I^\bullet) \propto p(I^t|\Theta^t)p_*^t(\Theta^t|I^{\bullet \setminus t}) \quad (3.4)$$

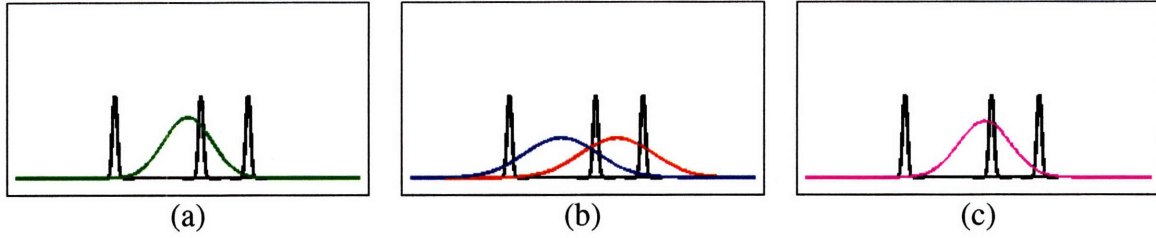


Figure 3-3: Synthetic example of the product model. Each pane shows the noisy likelihood function overlaid with (a) true temporal prior distribution; (b) two approximate temporal priors; and (c) product prior. The product prior is more similar to the true prior than any of its factors. Using this prior instead of either or the factors during inference should improve the posterior estimate.

By using the product prior RSMCM places most of the probability mass in the region covered by both of the constituent priors resulting in better selectivity, and, as discussed in section 1.2, improved posterior estimates. An illustration of this behavior is shown in Figure 3-3.

The reason that the true prior, $\bar{p}^t(\Theta^t|I^{\bullet t})$, can be better approximated by a product prior rather than any of its factors lies in the assumptions involved in designing constituent models of the RSMCM. Each of these models assumes that the instantaneous state nodes are independent when conditioned on the values of the temporal states. This assumption is rendered invalid by their use of the approximate evolution functions (and the transition distributions). Under very general conditions, the true prior can be represented as a product of an approximate prior and a “correcting function” (not necessarily a distribution),

$$\bar{p}^t(\Theta^t|I^{\bullet t}) = q_S(\Theta^t; I^{\bullet t})p_S^t(\Theta^t|I^{\bullet t}) = q_R(\Theta^t; I^{\bullet t})p_R^t(\Theta^t|I^{\bullet t}). \quad (3.5)$$

While q_S has a trivial definition,

$$q_S(\Theta^t; I^{\bullet t}) \equiv \frac{\bar{p}^t(\Theta^t|I^{\bullet t})}{p_S^t(\Theta^t|I^{\bullet t})}, \quad (3.6)$$

when the prediction errors in individual models are sufficiently large *and independent of each other*, the approximations

$$q_S(\Theta^t; I^{\bullet t}) \propto p_R^t(\Theta^t|I^{\bullet t}) \quad (3.7)$$

$$q_R(\Theta^t; I^{\bullet t}) \propto p_S^t(\Theta^t|I^{\bullet t}) \quad (3.8)$$

can be used for the reasons given in the previous section. This leads directly to our product model.

The effect of the RSMCM on the posterior distribution estimates of the temporal states S and R is more subtle, but similar to the one described above. Consider the joint temporal prior of Θ^t and S^t from the top chain (Figure 1-2(a)) only,

$$p_S^t(\Theta^t, S^t|I^{\bullet t}) = p(\Theta^t|S^t)p_S^t(S^t|I^{\bullet t})$$

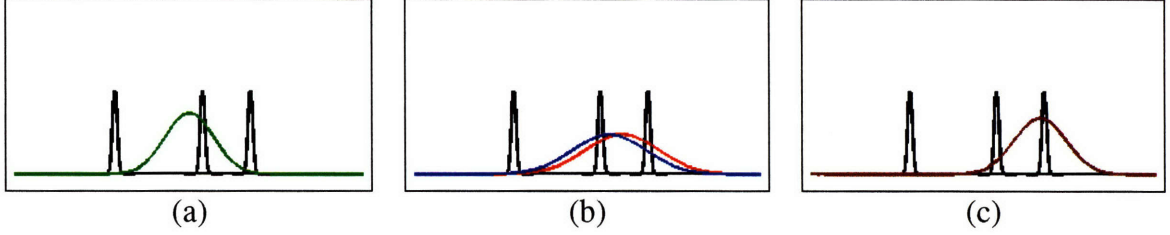


Figure 3-4: Synthetic example of the failure of the product model. Each pane shows the noisy likelihood function overlaid with (a) true temporal prior distribution; (b) two approximate temporal priors; and (c) product prior. The product prior is less similar to the true prior than any of its factors. Taking a product incorrectly increases confidence of the model about the wrong mean.

Again, the independence assumptions encoded by the single-chain graphical model are incorrect, since it does not use the true evolution function. Instead, in RSMCM we use the approximation

$$\bar{p}^t(\Theta^t, S^t | I^{\bullet t}) \approx p_R^t(\Theta^t | I^{\bullet t}) p_S^t(\Theta^t, S^t | I^{\bullet t}) \quad (3.9)$$

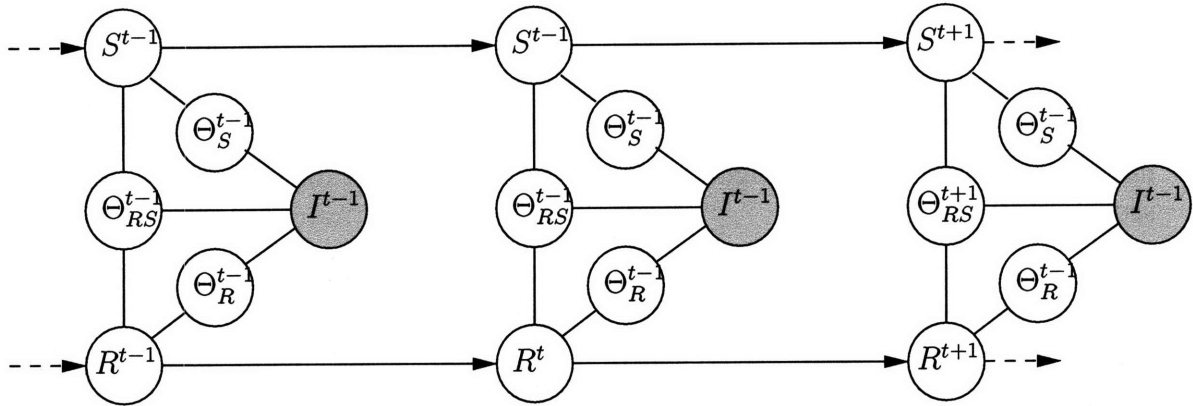
The product prior has the effect of decreasing the relative probability density of any configuration (Θ^t, S^t) that is likely under the approximate dynamic defined by the top chain such that Θ^t is *unlikely* under the temporal prior derived from the bottom model. The reverse relationship is

$$\bar{p}^t(\Theta^t, S^t | I^{\bullet t}) \approx p_S^t(\Theta^t | I^{\bullet t}) p_R^t(\Theta^t, R^t | I^{\bullet t}) \quad (3.10)$$

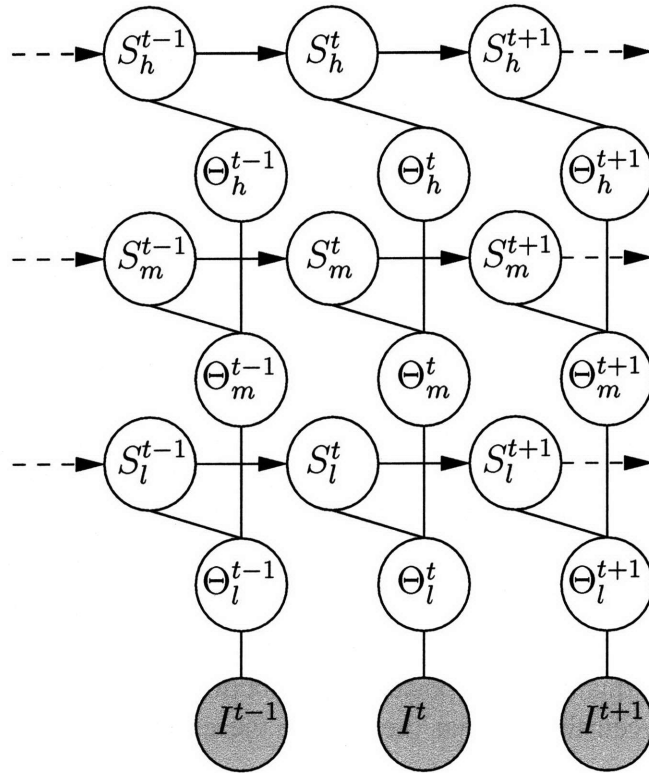
One-way information transfer (i.e. using equation 3.9 without the companion equation 3.10) is used in the hierarchical systems described in Section 2.3 (e.g., [109, 46]). For example the background model (i.e. the state of the low-level background subtraction) affects the high-level tracker state through the foreground labeling. The high-level knowledge, however, is not propagated to the background model. This can lead to mis-estimation of the model and will affect subsequent foreground estimates and the tracking performance. This case is discussed in detail in Chapter 4.1.

Although we have so far discussed the case when individual models use the same latent appearance features, it is possible to combine models with intersecting feature sets (the graphical representation of such a model is presented in Figure 3-5(a)). In that case, the combined feature model would be the union of individual feature sets, and the likelihood potentials are extended to produce uniform likelihoods for features that are not part of the original submodel. In general, when the feature sets are disjoint, the model would reduce to a PoHMMs model with non-interacting chains. Since we are interested in combining models that correspond to interacting stages of a feed-forward algorithm, we do not consider such cases.

It is also possible to define a *hierarchical* RSMCM. In such a model, rather than using a single instantaneous state representation, a set of such representations (e.g. at different levels of abstraction) is used, and separate approximate dynamic models are used for each of them. An example of this model is shown in Figure 3-5(b)



(a)



(b)

Figure 3-5: Variants of the redundant-state model. (a) RSMC model with intersecting feature sets. The instantaneous state is partitioned into 3 sub-states: Θ_{RS} is shared by both chains, Θ_R depends only on R , and Θ_S depends on S . This variant is appropriate for combining models that use partially intersecting feature sets. (b) A hierarchical RSMCM. This model combines three models describing evolution on the world at three different levels of abstraction, from highest Θ_h^t to lowest Θ_l^t .

3.2 Analyzing Approximation Validity

The redundant-state model described above is quite general, in that it allows combining any two probabilistic dynamics models sharing the same “feature” or instantaneous state representation. Since the errors involved in defining constituent models are stochastic in nature, both cases shown in Figures 3-3 and 3-4 may happen for particular observation sequences.

It is clear that there are cases when one of the constituent models would produce better results than RSMCM. For example if R and S are defined over the same state space, and share the same dynamics, then the product model would amplify the errors rather than decrease them! Even taking the product of the true temporal prior with itself results in a prior that is *more certain* (has smaller variance) and thus may assign very low probability to correct likelihood peaks. A somewhat less extreme case is shown in Figure 3-4. While the Product of HMMs [11] model may suffer from the same drawback, it is specifically trained to reduce the correlation between individual models and reduce the probability of being overconfident. Rather than learning the individual chains, we consider the models predefined and concentrate on determining when combination of the existing single-chain models into a RSMCM results in the improvement in posterior estimates.

The RSMCM combines stochastic models, and both cases shown in Figures 3-3 and 3-4 may happen for particular observation sequences. The following analysis is focused on determining which of these cases is more likely. Intuitively, we would expect that taking products of the independent approximations to the true prior would result in a better posterior estimates, and combining strongly correlated approximations would result in worse estimates. This is, as we will show, indeed true but with some caveats. When two priors are combined via a scaled product, the product is sharper than either of them, so if the approximations are not significantly wider than the true prior, there is a danger of the product prior to become overconfident, reducing the quality of the posterior estimates. Thus it is not possible to continuously improve the performance by combining more and more independent tracking models.

We will quantify the quality of the posterior distribution estimates as an expected value of KL-divergence between the optimal (i.e. using the correct model) and approximate posteriors.²

If $p(x|y)$ is the true posterior, and $q(x|q)$ is an approximation, then

$$C(q) = E_y [D_{KL}(p(x|y)||q(x|y))] \quad (3.11)$$

Since KL-divergence is an expectation over x in this case, the cost is an average error in posterior density computed over all possible state-observation pairs. If both true and approximate models use the same emission model $p_{y|x}(y|x)$ and differ only in the prior

²KL-divergence is, for reasons detailed in [18], a natural way to measure differences between distributions

$p_x(x)$ vs. $q_x(x)$, then

$$\begin{aligned}
E_{y \sim p_y(y)} [D_{KL}(p_{x|y}(x|y) || q_{x|y}(x|y))] &= D_{KL}(p_x(x) || q_x(x)) \\
&\quad - D_{KL}(p_y(y) || q_y(y)) \\
&\text{where} \\
p_y(y) &= \int p(y|x)p_x(x)dx \\
q_y(y) &= \int p(y|x)q_x(x)dx
\end{aligned} \tag{3.12}$$

The proof is given in the Appendix. The fact that using an approximate prior that is non-negligible anywhere the true prior is non-negligible incurs no cost when the observations are noise-free follows from this equation. Indeed, if $p(y|x) = \delta(y-x)$, then $p_y(y) = p_x(y)$ and $q_y(y) = q_x(y)$ and $D_{KL}(p_x(x) || q_x(x)) = D_{KL}(p_y(y) || q_y(y))$.

In order to demonstrate the properties of the RSMCM we analyze a case where the underlying and both approximate models are linear-Gaussian, since closed-form analysis of the RSMCM cannot be performed for general priors, However, this case is directly useful and provides intuition about more complicated cases.

We consider the system that is described by the following equations:

$$\begin{cases} \Theta^t = g(\Theta^{t-1}) + \omega_0^t, & \omega_0^t \sim N(0, \Sigma_0) \\ I^t = \Theta^t + \nu^t, & \nu^t \sim N(0, \Sigma_\nu), \end{cases} \tag{3.13}$$

where $N(\cdot; \mu, \Sigma)$ is a multi-variate Gaussian distribution with mean μ and covariance Σ . The approximate models are described by

$$\begin{cases} S^t = \hat{g}_1(S^{t-1}) + \omega_1^t, & \omega_1^t \sim N(0, \Sigma_1) \\ \Theta^t = S^t \\ I^t = \Theta^t + \nu^t, & \nu^t \sim N(0, \Sigma_\nu), \end{cases} \tag{3.14}$$

and

$$\begin{cases} R^t = \hat{g}_2(R^{t-1}) + \omega_2^t, & \omega_2^t \sim N(0, \Sigma_2) \\ \Theta^t = R^t \\ I^t = \Theta^t + \nu^t, & \nu^t \sim N(0, \Sigma_\nu), \end{cases} \tag{3.15}$$

Both approximate models share the emission (image generation) equations with the true model, but incorporate approximate evolution functions $\hat{g}_1(\cdot)$ and $\hat{g}_2(\cdot)$ rather than the true function $g(\cdot)$. All functions are modeled as linear. We denote $\mu_1 = \hat{g}_1(\Theta^{t-1}) - g(\Theta^{t-1})$ and $\mu_2 = \hat{g}_2(\Theta^{t-1}) - g(\Theta^{t-1})$.

For ease of analysis we assume that both approximate estimators are unbiased, that is

$$E_{\Theta^{t-1}} [\mu_1] = E_{\Theta^{t-1}} [\mu_2] = 0 \tag{3.16}$$

and have the covariance structure

$$E_{\Theta^{t-1}} \left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} (\mu_1 \ \mu_2) \right] = \begin{pmatrix} P_1 & P_{12} \\ P_{12}^T & P_2 \end{pmatrix} \quad (3.17)$$

with the expectation taken with respect to the marginal distribution $p(\Theta^{t-1})$.

Evolution equations of each model can be described via conditional distributions

$$p(\Theta^t | \Theta^{t-1}) = N(g(\Theta^{t-1}), \Sigma_0) \quad (3.18)$$

$$q_1(\Theta^t | \Theta^{t-1}) = N(g(\Theta^{t-1}), \Sigma_1) \quad (3.19)$$

$$q_2(\Theta^t | \Theta^{t-1}) = N(g(\Theta^{t-1}), \Sigma_2) \quad (3.20)$$

by using the property $\Theta^t = S^t$ and $\Theta^t = R^t$ of approximate models. All models share the same emission model

$$p(I^t | \Theta^t) = N(\Theta^t, \Sigma_\nu) \quad (3.21)$$

Using these conditional distributions we can define posterior distributions

$$\begin{aligned} p(\Theta^t | I^t, \Theta^{t-1}) &\propto p(I^t | \Theta^t) p(\Theta^t | \Theta^{t-1}), \\ q_1(\Theta^t | I^t, \Theta^{t-1}) &\propto p(I^t | \Theta^t) q_1(\Theta^t | \Theta^{t-1}), \text{ and} \\ q_2(\Theta^t | I^t, \Theta^{t-1}) &\propto p(I^t | \Theta^t) q_2(\Theta^t | \Theta^{t-1}). \end{aligned}$$

We assume that single-chain models combined into a RSMCM are optimal, in the sense that they use noise distributions that would, on average, result in the best posterior estimates. Lemma 1 describes the conditions under which $C(q_1)$ and $C(q_2)$ are optimal.

Lemma 1. $C(q_1)$ and $C(q_2)$ are minimized by setting $\Sigma_1 = P_1 + \Sigma_0$ and $\Sigma_2 = P_2 + \Sigma_0$.

It is worth noting that for the optimal performance, the dynamic noise in the approximate models needs to be inflated exactly by the error covariance of the transition function — using smaller noise covariance results in a prior that is too sharp (i.e. does not assign enough probability to the region covered by the true priors). Using larger noise covariance results in a prior that is overly broad.

Theorem 1 describes *sufficient* conditions under which the product approximation that uses the conditional distribution

$$q_*(\Theta^t | \Theta^{t-1}) \propto q_1(\Theta^t | \Theta^{t-1}) q_2(\Theta^t | \Theta^{t-1})$$

has cost $C(q_*)$ that is less than the cost of each of the constituent models.

Theorem 1. $C(q_*) < C(q_1)$ and $C(q_*) < C(q_2)$ if the respective indicator matrices

$$\begin{aligned} \Gamma_1 &= Q_{1\eta} P_1 - (Q_{1\eta} P_{12} + (Q_{1\eta} P_{12})^T + (Q_{1\eta} \Sigma_0)^T) \text{ and} \\ \Gamma_2 &= Q_{2\eta} P_2 - (Q_{2\eta} P_{12}^T + (Q_{2\eta} P_{12}^T)^T + (Q_{2\eta} \Sigma_0)^T) \end{aligned}$$

are positive semidefinite when

$$\begin{aligned} Q_{1\eta} &= (I + (\Sigma_0 + P_1)\Sigma_\nu^{-1})^{-1} \\ Q_{2\eta} &= (I + (\Sigma_0 + P_2)\Sigma_\nu^{-1})^{-1} \end{aligned}$$

The proofs of this theorem and Lemma 1 can be found in the Appendix.

Theorem 1 confirms our intuition that the models combined into RSMCM should be decorrelated. In the extreme case where the models are perfectly correlated, $P_1 = P_{12}$ and $\Gamma_1 = - (Q_{1\eta}^T (P_{12}^T + \Sigma_0^T))^T$ is not positive semidefinite.

While it is well understood that unbiased estimators, whose errors are uncorrelated, can be coherently combined to produce an improved estimate, the previous analysis is more specific. For the Gaussian case, Theorem 1 describes the degree of correlation in the estimation errors which can be tolerated and still produce an improved using an RSMCM. It is instructive to consider a one-dimensional case when all constituent matrices become scalars. The sufficient conditions then reduce to

$$\begin{aligned} p_1 &\geq 2p_{12} + \sigma_0^2 \text{ and} \\ p_2 &\geq 2p_{12} + \sigma_0^2 \end{aligned} \tag{3.22}$$

That is each of the diagonal terms on the covariance matrix of the estimators should be greater than the sum of the off-diagonal terms and the noise variance of the underlying model. The off-diagonal terms in this case are equal to $\sqrt{p_1 p_2} \rho_{12}$ where ρ_{12} is the correlation coefficient. For the above conditions to be satisfied, it is necessary for the correlation coefficient to be less than 0.5.

Another observation that can be made from this theorem is that it is possible for RSMCM combination to improve posterior estimates of one chain and worsen those of the other. This will happen if the estimators are not perfectly uncorrelated and one of them has significantly higher variance than the other. In this case the off-diagonal terms (P_{12} and P_{21}) would dominate one of the diagonal ones (P_1 or P_2) making one of the indicator matrices not positive semidefinite. This behavior is illustrated in Section 4.2.4.

3.3 Inference in Redundant-State Models

Single-chain models are popular because there exist efficient algorithms for performing inference in them. While our proposed multi-chain model is loopy (Figure 3-6(a)), making inference complicated in general, we take advantage of the fact that we are interested only in marginal distributions for the state nodes and propose a set of algorithms based on Belief Propagation [81] for inference in RSMCM. The standard BP techniques for loopy graphs involve simultaneous exchange of messages between all nodes; approaches that use a different *message schedule* have also been proposed [135]. In this section we propose a message schedule that allows for efficient *filtering* in the redundant state model.

Consider the model in Figure 3-6(a). At time $t = 1$, we are concerned with nodes with superscripts (times) $t \leq 1$. If the initial states S^0 and R^0 are independent (as shown), then the resulting subgraph is a tree, and we can use standard Belief Propagation techniques to

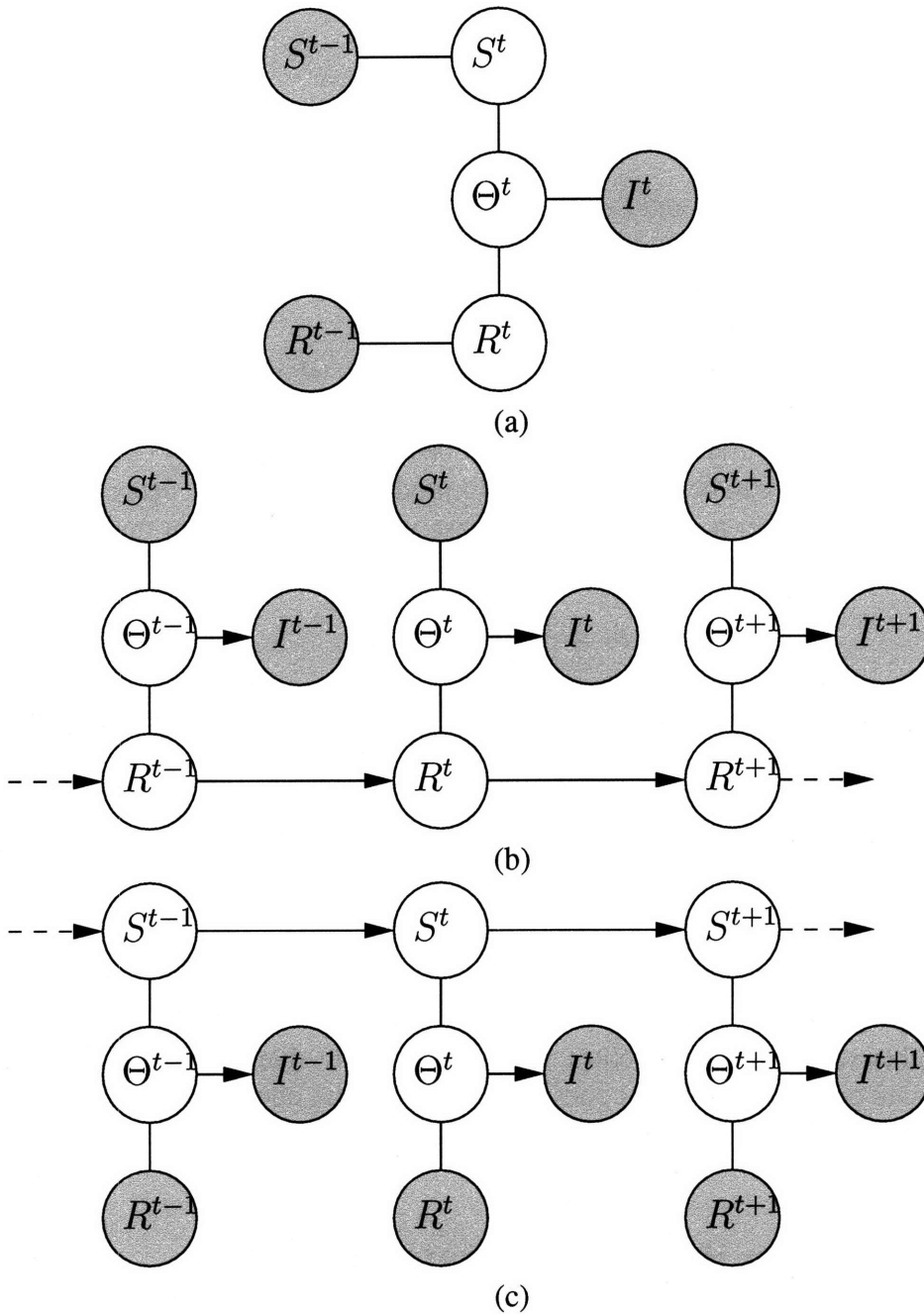


Figure 3-6: Graph structures used in inference algorithms in the dual-chain model. (a) A tree-shaped subgraph on which a single step of approximate filtering is performed. The marginal distributions, $p(S^{t-1}|I^{0..t-1})$ and $p(R^{t-1}|I^{0..t-1})$, have been computed at the previous iteration, and are not modified; I^t is observed. (b, c) Subgraphs for coordinate ascent in the dual-chain model. By fixing values of states $S^{0..T}$, the structure is reduced to the single-chain model shown in (b). Existing feature-extraction algorithms may be adapted to perform inference in this model with relatively little modifications. When $R^{0..T}$ are fixed (c) an existing high-level optimization algorithm can be applied.

compute exact marginal distributions at state nodes S^1 and R^1 .

$$p(S^1|I^1) = \frac{1}{Z} \left[\int \phi(S^1, S^0) p(S^0) dS^0 \right] \left[\int \phi(\Theta^1) \phi(\Theta^1, S^1) \int \phi(\Theta^1, R^1) \int \phi(R^1, R^0) p(R^0) dR^0 dR^1 d\Theta^1 \right], \quad (3.23)$$

where $\phi(\Theta^1) \equiv \phi(I^1, \Theta^1)$. The expression for $p(R^1|I^1)$ can be similarly derived.

Filtering at the next time step ($t = 2$) is more complex since the model now contains loops and the exact inference would require representing the joint $p(S^1, R^1|I^1)$:

$$p(S^2|I^1, I^2) = \frac{1}{Z} \int \phi(\Theta^2) \phi(\Theta^2, S^2) \int \phi(\Theta^2, R^2) \iint \phi(S^2, S^1) \phi(R^2, R^1) p(S^1, R^1|I^1) dR^1 dS^1 dR^2 d\Theta^2. \quad (3.24)$$

In order to simplify computations, we approximate the joint distribution, $p(S^1, R^1|I^1)$ with a product, $q(S^1)q(R^1)$. It can be easily shown that the best such approximation (in the KL-divergence sense) is the product of marginal distributions, $p(S^1|I^1)$ and $p(R^1|I^1)$. Substituting $p(S^1|I^1)p(R^1|I^1)$ for $p(S^1, R^1|I^1)$ in Equation 3.24, we obtain an approximate inference equation:

$$p(S^2|I^2) = \frac{1}{Z} \int \phi(S^2, S^1) p(S^1) dS^1 \int \phi(\Theta^2) \phi(\Theta^2, S^2) \int \phi(\Theta^2, R^2) \int \phi(R^2, R^1) p(R^1) dR^1 dR^2 d\Theta^2. \quad (3.25)$$

The similarity between Equations (3.23) and (3.25) suggests an approximate filtering algorithm that estimates marginal distributions of the state variables by recursively applying Belief Propagation to acyclic subgraphs of the form shown in Figure 3-6(a), using the marginal state distribution obtained at time $t - 1$ as priors at time t .

It can be shown that this approximation preserves the main property of the exact model: the appearance features that are assigned zero probability under *any* of the constituent models are assigned zero probability in the computation of *all* of the marginal distributions.

The messages exchanged between nodes during Belief Propagation are computed as described in Algorithm 1. An illustration of messages exchanged during one iteration of the algorithm in a system similar to one discussed in Section 3.2 is shown in Figure 3-7. Note that computations required for the prediction and update steps, as well as for part of the feature estimation step, are the same as those of individual object tracking and feature extraction algorithms.

If inference on constituent Markov chains were performed individually, it would still involve steps analogous to the prediction, update, and to part of the feature prediction steps of the approximate algorithm; consequently, combining models introduces very little additional complexity to the inference process.

At the first glance, the proposed algorithm appears to be very similar to one proposed by Boyen and Koller in [8] for factorial models. The differences are, in fact, quite significant.

Algorithm 1 Recursive Belief Propagation Algorithm for Filtering in a Redundant State Model

 INPUTS $p(S^0)$ and $p(R^0)$
for all $t > 0$ **do**

PREDICT the current states of the constituent models by computing messages:

$$\mu_{S^{t-1} \rightarrow S^t} = \int dS^{t-1} \phi(S^t, S^{t-1}) p(S^{t-1} | I^{1..t-1}) \text{ and}$$

$$\mu_{R^{t-1} \rightarrow R^t} = \int dR^{t-1} \phi(R^t, R^{t-1}) p(R^{t-1} | I^{1..t-1}).$$

ESTIMATE Instantaneous state distributions based on predicted states and current observations, compute messages:

$$\mu_{S^t \rightarrow \Theta^t} = \int dS^t \phi(\Theta^t, S^t) \mu_{S^{t-1} \rightarrow S^t},$$

$$\mu_{R^t \rightarrow \Theta^t} = \int dR^t \phi(\Theta^t, R^t) \mu_{R^{t-1} \rightarrow R^t},$$

$$\mu_{\Theta^t \rightarrow S^t} = \int d\Theta^t \mu_{R^t \rightarrow \Theta^t} \phi(I^t, \Theta^t), \text{ and}$$

$$\mu_{\Theta^t \rightarrow R^t} = \int d\Theta^t \mu_{S^t \rightarrow \Theta^t} \phi(I^t, \Theta^t).$$

UPDATE individual model state distributions by computing message products:

$$p(S^t | I^{0..t}) \propto \mu_{S^{t-1} \rightarrow S^t} \mu_{\Theta^t \rightarrow S^t}$$

$$p(R^t | I^{0..t}) \propto \mu_{R^{t-1} \rightarrow R^t} \mu_{\Theta^t \rightarrow R^t}.$$

 OUTPUT $p(S^t | I^{1..t})$ and $p(R^t | I^{1..t})$
end for

Our algorithm maintains and updates multiple redundant representations of the posterior and the interaction between component temporal chains is quite strong, since they all model evolution of the same process. The B&K algorithm propagates a non-redundant *factored* representation and the main underlying assumption is that individual chains are interacting weakly, if at all. Furthermore, in our filtering algorithm information exchange between constituent chains is moderated by the instantaneous state nodes and the need for junction-tree based computation never arises. The major difference between our algorithm and algorithms proposed in [34, 64, 131, 132] is that in our case both chains are processing the same stream of features, and the interaction between the chains is performed at the shared latent feature level. The other algorithms process parallel streams of features, and the interaction is performed at the temporal state level.

3.4 Batch Optimization Algorithm

While filtering is appropriate for online tasks, some object-tracking problems are formulated as global optimizations in single-chain models such as the one in Figure 3-1. For example, in structure-from-motion estimation we may be interested in computing the shape of the object based on *all* observed data, that is computing $\arg \max_{S^{0..T}} p(F^{1..T} | S^{0..T})$. Once again, the algorithms developed for single-chain models need to be modified to be of use in the dual-chain setting.

We base our optimization approach on a coordinate ascent algorithm that alternates between optimizing one set of states (either $R^{0..T}$ or $S^{0..T}$) while keeping the other one fixed. The dual-chain structure, with latent feature nodes separating states, naturally lends itself to this algorithm. Fixing one set of states reduces the problem to a single-chain

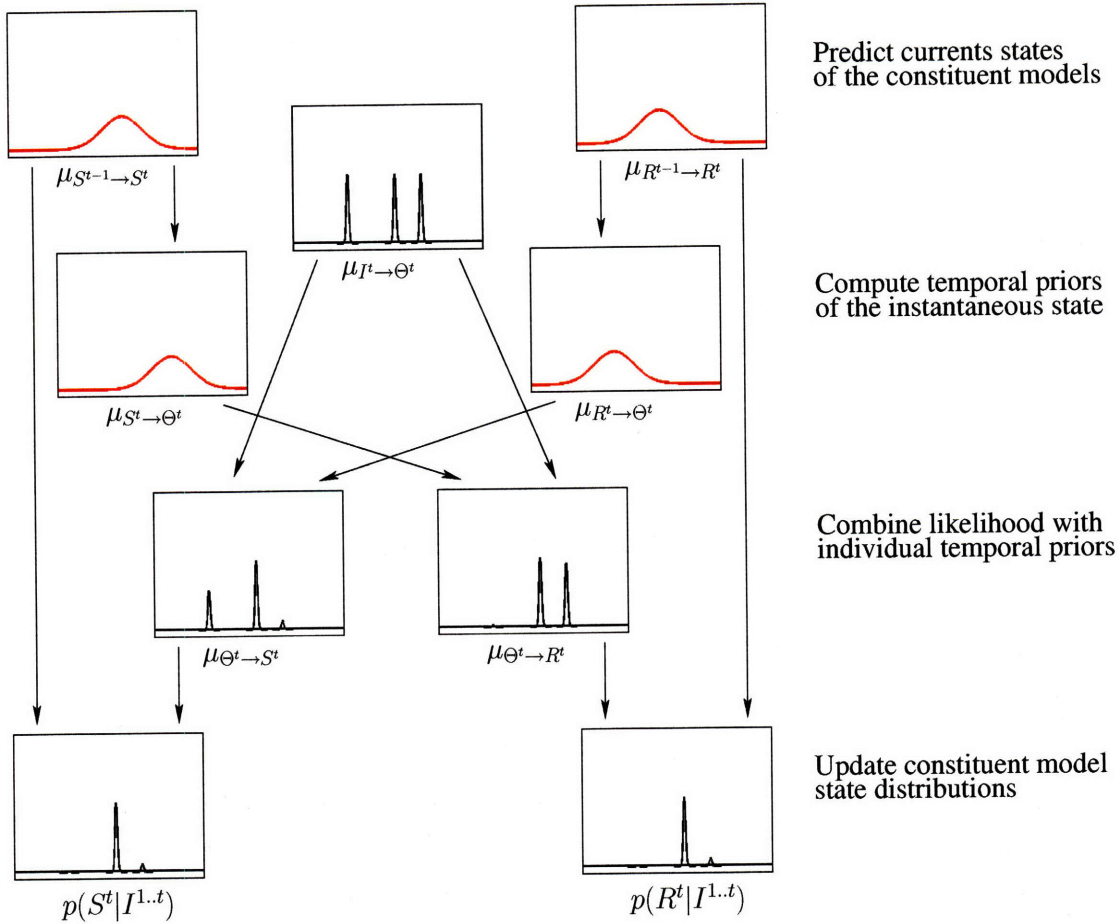


Figure 3-7: Messages exchanged during one approximate filtering iteration described in Algorithm 1. The arrows indicate dependencies between individual messages.

optimization that can be performed with available algorithms, (cf. Figures 3-6(b, c)). The summary of our method is presented in Algorithm 2.

3.5 Relationship to Turbo Code Decoding Algorithm

It is instructive to consider the similarity between the redundant state models for tracking and powerful error correcting turbo codes [7] for data communication. Both are based on combining multiple redundant weak sequence models to obtain a stronger model.

Turbo code decoding (1/3 code with 1 bit memory) can be viewed as inference in a graphical model shown in Figure 3-8. The input codeword $X = (X^{t+1}, X^{t+2}, X^{t+3}, X^{t+4})$ is transmitted directly, as well as being passed as input to two convolution encoders in the original and permuted orders. The output of convolution encoders (in this case just parities Y_i^j between neighboring bits in the encoder inputs) is also transmitted. The decoding is equivalent to computing marginal distributions of $X^{t+1} \dots X^{t+4}$ from the noisy observations $\hat{X}^{t+1} \dots \hat{X}^{t+4}, \hat{Y}_1^{t+1} \dots \hat{Y}_1^{t+4}, \hat{Y}_2^{t+1} \dots \hat{Y}_2^{t+4}$.

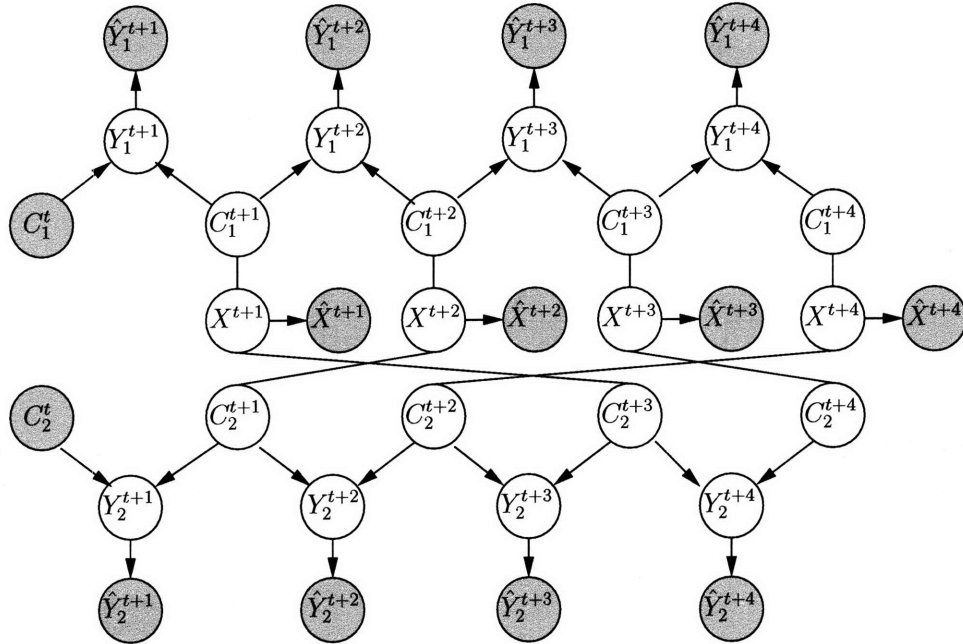


Figure 3-8: Graphical model representation of turbocode decoding problem. The codeword bits $X^{t+1..t+4}$ contaminated by noise are observed as $\hat{X}^{t+1..t+4}$. The codeword also serves as input to convolution encoder as direct copy $(C_1^{t+1}, C_1^{t+2}, C_1^{t+3}, C_1^{t+4}) = (X^{t+1}, X^{t+2}, X^{t+3}, X^{t+4})$ and permuted copy $(C_2^{t+1}, C_2^{t+2}, C_2^{t+3}, C_2^{t+4}) = (X^{t+2}, X^{t+4}, X^{t+1}, X^{t+3})$. The parity bits Y_i^j are computed by the convolution decoders as $Y_i^j = C_i^{j-1} \oplus C_i^j$, and are observed contaminated by noise as \hat{Y}_i^j .

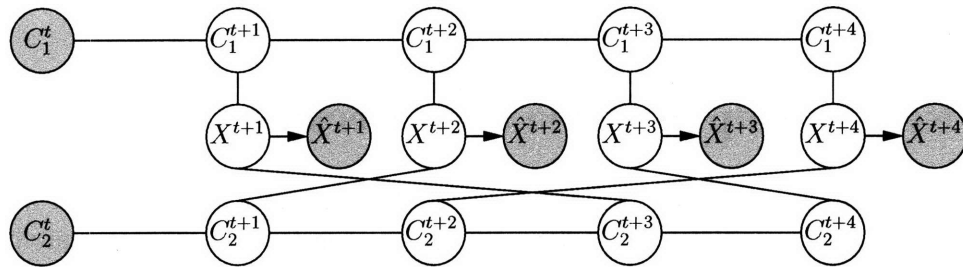


Figure 3-9: Dual-chain representation of turbocode decoding problem. This model can be obtained from one shown in Figure 3-8 by marginalizing over parity bits Y_i^j .

Algorithm 2 Coordinate Ascent for Batch Optimization in a Redundant State Model

INITIALIZE marginal distributions $p^0(S^t)$ to uniform
 $k \leftarrow 0$
while not converged **do**
 $k \leftarrow k + 1$
 PERFORM exact inference on the graph shown in Figure 3-6(b), using marginal distributions $p^{k-1}(S^t)$
 INITIALIZE marginal distributions: $p^k(R^t) \leftarrow p(R^t|I^{0..T})$
 PERFORM exact inference on the graph shown in Figure 3-6(c), using marginal distributions $p^k(R^t)$
 INITIALIZE marginal distributions: $p^k(S^t) \leftarrow p(S^t|I^{0..T})$
end while
OUTPUT marginal distributions: $p(S^t|I^{0..T}) \leftarrow p^k(S^t)$ and $p(R^t|I^{0..T}) \leftarrow p^k(R^t)$

The original graphical model can be transformed to one shown in Figure 3-9 by marginalizing over the values of Y_i^j . The potentials between neighboring elements of C_i sequences then become

$$\phi(C_i^{t-1}, C_i^t) = \begin{cases} p(Y_i^t = 1|\hat{Y}_i^t) & C_i^{t-1} \neq C_i^t \\ p(Y_i^t = 0|\hat{Y}_i^t) & C_i^{t-1} = C_i^t \end{cases}$$

The particular algorithm proposed in [7] has been shown in [69] to be analogous to Belief Propagation with a particular schedule: iterating optimization of marginal distributions of states in one chain while keeping the marginals of the second chain's states fixed. This is exactly the schedule proposed in Algorithm 2.

Turbo codes as described in [7] can be seen as an instance of the RSMCM, with redundancy provided by using different permutations of the codeword bits as input to the identical convolution encoders. The turbo coding algorithm is thus inherently block-oriented, with larger blocks resulting in better performance. RSMCM, in contrast, is able to combine different dynamic models making possible online inference (e.g. using Algorithm 1).

3.6 Summary

We have proposed a methodology for combining simple dynamical models with redundant representations as a way of modeling more complex dynamical structures. The approach was motivated by the simple observation that nearly all generative-model based tracking algorithms for tracking complex structures implicitly marginalize over an intermediate feature representation between state and observation. By making the feature representation explicit in our approach we obtained a straightforward means of mediating between simpler models as a means of capturing more complex behavior.

Exact inference on the resulting structure is complicated due to the introduction of loops in the graphical structure representing the combined models. However, we have proposed two methods for adapting algorithms designed for constituent modules to operate in a com-

bined system. An approximate inference method based on sequential inference on acyclic subgraphs provides a suitable alternative to exact inference appropriate for online tracking (filtering). A coordinate-ascent based algorithm has been designed for the batch inference case. Both approximations have the important property that infeasible configurations in *any* of the naive models precluded an infeasible configuration in *all* of the others. This property may be both an advantage and a disadvantage of the system framework. On one hand it allows restricting each model's search only to those configurations that are possible under other models, but on the other it requires careful design of the constituent systems, so that they never assign zero probability to feasible configurations.

We have also shown a connection between RSMCM and turbo codes, where the turbo decoding procedure is an instance of the batch RSMCM inference algorithm.

Chapter 4

Applications of RSMCM in Hierarchical Tracking Systems

Motion analysis algorithms are often structured in a multistage fashion, with each stage operating at a particular spatio-temporal scale and exploiting a different model of scene dynamics. Systems of this type are usually more computationally efficient than monolithic ones that jointly model local and global dynamics. They also have the advantage of modularity, as algorithms at each stage can be designed independently. Rather than using raw pixel data, high-level (large scale) stages treat the output of early, low-level ones as observations. For example, an algorithm may start by extracting local features (e.g. foreground/background labels or feature point tracks) from incoming frames, use these features to determine poses of the objects moving in the scene, and then analyze object interaction based on the individual objects' poses. High-level algorithms use models that are often too coarse and/or approximate for local motion estimation, but take into account global spatial relationships.

Low-level algorithms ignore global spatial relationships by modeling the evolution of each image patch (in feature extraction [109, 120]) or object (in object tracking [77]) independently, and compensating for it with restrictive assumptions about the local behavior of the scene. Feature-point trackers usually assume that the image patch about the point of interest has a relatively stable appearance. Adaptive background subtraction modules typically assume that foreground objects do not remain stationary for extended periods of time. When these assumptions are violated, the resulting errors (e.g. so-called “sleeping man problem”, Figure 2-6), are propagated to higher-level modules, and those are not always able to correct them.

While algorithms operating at each stage are often formulated as inference in probabilistic generative models, most existing multi-stage systems are formed in an ad-hoc fashion and do not have a consistent probabilistic interpretation—e.g., the uncertainty information is propagated only in one direction, from low- to high-level models. One alternative is to use more sophisticated algorithms (e.g. [119]) to introduce more high-level information directly into feature extraction process to reduce its sensitivity to locality assumptions. This makes the design of low-level modules complicated and substantially increases their running times. A more attractive solution would be to introduce a feedback connection into the hierarchical framework, in which knowledge available to high-level algorithms is prop-

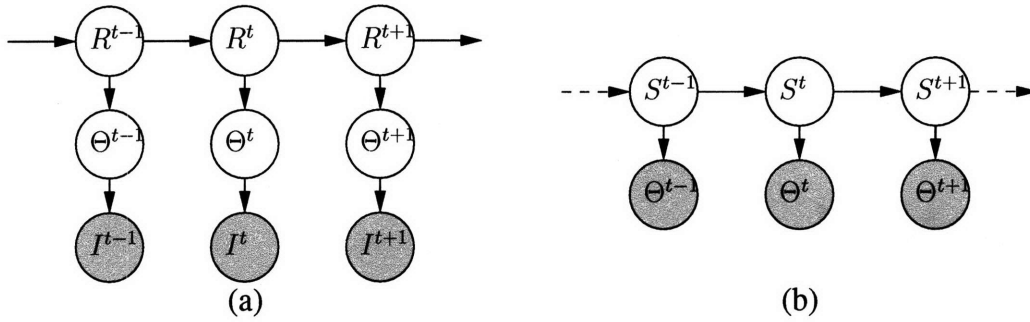


Figure 4-1: Hierarchical processing of sequential data (repeating Figure 2-5). The marginal distributions $p(\Theta^t | I^{0:t})$ are computed using the low-level feature extractor (a). These marginals are then used as input to high-level algorithm (b).

agated to low-level processing modules making them more robust to local perturbations.

This need to incorporate a feedback mechanism into multistage systems has long been recognized [91, 6]. There are three desirable criteria for a viable feedback framework. First, it should preserve existing modularity i.e., not be reduced to a monolithic model. Second, it should allow the use of existing algorithms with minimal modifications. Finally, it must consistently propagate uncertainty from high- to low-level processing. While the first two requirements are satisfied by the ad-hoc feedback mechanisms, they lack the consistency property. The feedback connection introduced in the redundant state model meets all three criteria.

In this chapter we explore applications of redundant state models for robust tracking in hierarchical systems. The hierarchical models (reviewed, e.g., in Section 2.3) lend themselves very well to conversion into RSMCMs. The low- and high-level models (Figure 4-1, repeating Figure 2-5) naturally share the feature values Θ that can be treated as an instantaneous state. While high-level module (Figure 4-1(b)) treats features as observed and does not explicitly define an image generation model, it is not strictly necessary since it is implicitly assumed to be shared with the feature extraction.

We demonstrate the performance of RSMCM-based hierarchical systems on two specific applications: adaptive background subtraction and structure from motion estimation. In both applications we demonstrate very high level of performance even though very simple feature extraction modules are used. The work presented in this section has been published in [117] and [116].

4.1 Applying Redundant State Modeling to Adaptive Background Maintenance

Background subtraction is a first step in many object tracking applications. It is used to determine likely locations of objects of interest (foreground objects) by comparing a newly acquired frame with an internally maintained model of the scene without objects of interest (background). One of the most popular classes of background maintenance systems are

the so called adaptive models [109, 42, 123, 40]. Such models are able to adjust to scene changes due to causes other than objects of interest (e.g., lighting variations).

Background models are usually designed to be task independent, and this often means that they can use very little high-level information. While region-based reasoning may be utilized at every individual frame [109, 123], temporal consistency is usually exploited only on a per-pixel basis. This limitation can cause the scene model to adapt to foreground objects that remain stationary for extended periods of time. After these objects “fade” into the background, their locations are no longer considered as regions of interest.

Several approaches to incorporating information about foreground objects into background maintenance have been proposed. They may be broadly grouped into two categories: probabilistic frameworks that *jointly* model scene and foreground object evolution [128], and systems consisting of separate modules for scene modeling and high-level inference (e.g., object tracking) [109, 42, 123]. Adjustments to the background model in modular systems depend on heuristic-based feedback from the higher-level modules.

We use the redundant state model to implement the first (to our knowledge) approach that incorporates background modeling and object tracking in a unified statistical framework, while still enabling an efficient modular implementation. Our approach is based on the observation that both background maintenance and object tracking may be formulated as state estimation in dynamic Bayesian networks representing generative models (see section 4.1.2). Each generative model is approximate. The background model models the underlying scene but is agnostic about pixels generated by the moving objects. On the other hand, the object tracker models foreground pixels but not the rest of the image.

These models have different failure modes: the background model fails when foreground pixel values are close enough to the expected background, and the tracker fails when the background contains patterns similar to the ones expected for the objects being tracked. We were able to use the RSMCM to pool the knowledge from both models and improve background model estimation, segmentation and tracking. Combining individual modules into the redundant-state framework improves the results without incurring a significant computational cost in comparison to the feed-forward system.

4.1.1 Prior Approaches

While many approaches to adaptive background modeling have been proposed, it remains an active research area. Several methods have been proposed that incorporate background estimation and object tracking in a single monolithic system [51, 128], but most systems take a modular approach that allows using a single background subtraction subsystem in different applications.

Stand-alone background subtraction algorithms assign background/foreground labels based on the history of the local measurements in a particular location. Popular modeling techniques may be separated into two broad classes, parametric and non-parametric [15]. Non-parametric models [123, 137] use previously observed frames directly, and consider a pixel to belong to the foreground if its value is different from a sufficient number of stored values. Parametric models maintain a representation of pixel value probability distribution (such as a mixture of Gaussians in [109]) that is recursively updated at every frame.

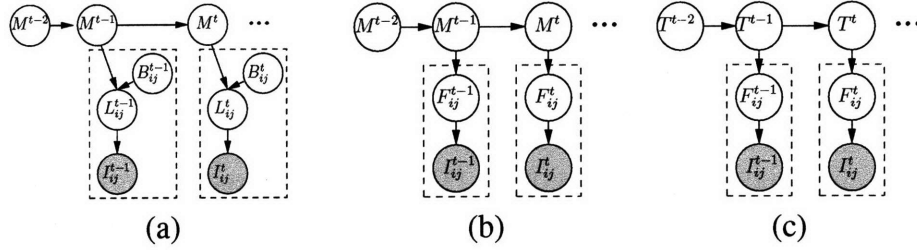


Figure 4-2: Combining background maintenance and object tracking models. (a) A generative model used for background maintenance. At time t , pixel j belongs to the background if $B_{ij}^t = 1$. In this case its latent value, L_{ij}^t , is generated according to $p(L_{ij}^t|M^t)$, where M_{ij}^t are the sufficient statistics of the scene background distribution. Otherwise the latent value is generated from a uniform distribution. L_{ij}^t contaminated by noise is observed as I_{ij}^t . Nodes enclosed in dashed rectangles are duplicated for every j . (b) The intermediate feature representation, $F_{ij}^t = (L_{ij}^t, B_{ij}^t)$, $p(F_{ij}^t|M^t) = p(L_{ij}^t|M^t, B_{ij}^t)p(B_{ij}^t)$. (c) Generative model used for object tracking. The state T^t contains both spatial and appearance information about moving objects. If a pixel belongs to an object, then $B_{ij}^t = 0$, and L_{ij}^t is set depending on the object appearance. Otherwise $B_{ij}^t = 1$ and L_{ij}^t is generated by a uniform distribution.

Local measurements, such as depth [42] and spatial and temporal gradients [84] have been used in addition to raw intensity values to improve segmentation.

While methods have been proposed for using high-level information to handle global changes (e.g., lights being switched on and off) [41], we are not aware of statistically consistent approaches to incorporating temporal information from object tracking into background modeling.

4.1.2 Redundant Model Formulation

For this application we have considered a background maintenance system, similar to that described in [109]. At every time step it performs two tasks: assigning each pixel in the image a probability of belonging to the background or foreground class, and modifying the internal representation of the scene based on the current input. Its operation may be described as inference (filtering) in the dynamic Bayesian network shown in Figure 4-2(a).

This network represents a *generative* model of image formation as follows: first the background model, M^t , is predicted based on the model at the previous time step, M^{t-1} , and transition probability $p(M^t|M^{t-1})$ that is usually based on a diffusion model. A binary background label, B_{ij}^t , is generated according to the prior probability, $P(B_{ij}^t)$, for every pixel (i, j) . The latent pixel value, L_{ij}^t , is generated according to the predicted model, M^t , if the pixel belongs to background ($B_{ij}^t = 1$) and by a uniform distribution otherwise. The value of L_{ij}^t contaminated by observation noise is then observed as I_{ij}^t . The posterior background label probability for every pixel and the updated model may be computed using standard inference techniques. It may be shown that the model update rules in such methods as [109] may be derived in this manner.

Note that while per-pixel models (M_{ij}^t) are usually used, this is not assumed. In the following discussion, we use the notation $I^t = \cup_{ij} I_{ij}^t$ to represent the complete observed

image, $B^t = \cup_{ij} B_{ij}^t$ for the background probability image, etc.

To simplify derivations, we use a slightly modified generative model shown in Figure 4-2(b), where $F_{ij}^t = (L_{ij}^t, B_{ij}^t)$ is the instantaneous pixel representation, $p(F_{ij}^t|M^t) = p(L_{ij}^t, B_{ij}^t|M^t) = p(L_{ij}^t|M^t, B_{ij}^t)p(B_{ij}^t)$, and $p(I_{ij}^t|F_{ij}^t) = p(I_{ij}^t|L_{ij}^t)$.

This generative model represents the evolution of the scene only approximately, since while it models background pixels, it makes an (incorrect) assumption that the foreground pixels are generated by a uniform distribution. Thus temporal dependency between foreground pixel locations and values is not modeled, and the independence assumptions made in DBN do not hold.

Incorporating the dependency between F^t and prior observations, which remains unrepresented by this generative model, allows for better estimation of both the background model and the background/foreground labels. When a model of foreground object motion is available (e.g., when the output of the background subtraction system is used for object tracking), we can use redundant-state formulation to incorporate it into our inference algorithm.

We use an independent object tracking model similar to that described in [51]. Each object state is a five-tuple $O_n^t = (x_n^t, y_n^t, u_n^t, v_n^t, G_n^t)$, with position (x_n^t, y_n^t) , velocity (u_n^t, v_n^t) , and shape G_n^t . The object pose evolves in a first-order linear fashion,

$$\begin{pmatrix} x_n^t \\ y_n^t \\ u_n^t \\ v_n^t \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_n^{t-1} \\ y_n^{t-1} \\ u_n^{t-1} \\ v_n^{t-1} \end{pmatrix} + \eta, \quad \eta \sim N(0, \Sigma_\eta), \quad (4.1)$$

The shape is encoded as a binary foreground/background mask. The shape evolution is modeled with random flipping of individual foreground/background labels

$$\begin{aligned} p(G_n^t(k, l) = b | G_n^{t-1}(k, l) = b) &= 1 - \alpha \\ p(G_n^t(k, l) = b | G_n^{t-1}(k, l) = 1 - b) &= \alpha \end{aligned} \quad (4.2)$$

where α is the evolution rate.

The state T^t is a collection of object states, $T^t = \{O_i^t\}$. Each object is modeled as evolving independently, and with certain probability objects can disappear and new ones appear anywhere in the scene. To simplify implementation we ignore the possibility of occlusion, and thus the background map B^t is generated deterministically via

$$B_{ij} = \max_n G_n(i - \lfloor x^t \rfloor, j - \lfloor y^t \rfloor) \quad (4.3)$$

Our object model does not include texture information, so the latent pixel values are assumed to have uniform probability at every time step.

It is worth noting that the probability of an object appearing in the scene can be made such that it makes $p(B_{ij}|T^{t-1}) \approx 0.5$, making the prediction at the locations not corresponding to any of the tracked objects uninformative. This has the effect of *not* modifying the behavior of the background maintenance module in the RSMCM in those locations. If the objects are modeled as being able to appear only in parts of the scene, then pixels

assigned low probability by the object model in the rest of the scene would be considered background, and the effect of the object-based prior would be to increase the adaptation rate at those pixels.

4.1.3 Implementation and Results

Since our objective was to compare the performance of the background maintenance system with and without tracking feedback, we chose to implement a very simple adaptive module, although a more advanced system can certainly be used. The background distribution was modeled with a single (per-pixel) Gaussian with fixed variance and variable mean. Model dynamics and observation noise were also represented with Gaussian distributions with fixed variances. We used an object (blob) tracker similar to the one described in [109].

The resulting RSMCM implementation is able to solve the “sleeping man” problem described in Section 2.3. Compare the segmentation results from a stand-alone system in Figure 2-6 and the redundant state system output in Figure 4-3.

We have evaluated the redundant state system and three stand-alone background subtraction models with different settings of $P(B)$ (0.5, 0.3, and 0.2) on datasets provided for the PETS 2001 workshop.¹ Algorithms were evaluated as follows: at every frame, we have computed a raw foreground map by thresholding (at 0.5) the background probability value at every pixel and extracted a set of connected components. Sample frames from the sequences with corresponding estimated background images and foreground components are shown in Figures 4-6 and 4-7.

We were interested in three common types of coarse errors: missing people, missing vehicles, and incorrectly detected “ghost” objects. We evaluated the following performance metrics: (1) less than 50% of a pedestrian covered by extracted components; (2) less than 50% of a vehicle covered by extracted components; and (3) a foreground component detected in a location where no moving objects were present. These error types are illustrated in Figure 4-4. Results of quantitative comparison between the RSMCM implementation and stand-alone modules are summarized in Figure 4-5. Raw results for the first and second sequences are presented in 4-5(a, c). Results for the first sequence corrected for the type 2 errors due to the car that remains stationary for the rest of the sequence are shown in 4-5(b). Changing $P(B)$ in a stand-alone module results in a trade-off between missing parts of foreground objects (types 1 and 2) and extra detections (type 3). In the redundant state system we are able to have high $P(B)$ (reducing the number of type 3 errors) and use high-level feedback to avoid fast adaptation (reducing the number of type 1 and 2 errors). As a result RSMCM system significantly outperforms stand-alone background subtraction modules.

Importantly, replacing the feed-forward tracking algorithm with a RSMCM framework did not result in a large performance penalty. In our experiments, the difference between running times of the RSMCM algorithm and the feed-forward system was less than 4%. Partially optimized code on a 2.8GHz workstation was able to achieve 9.6fps for sequential processing and 9.3fps for RSMCM processing on 768×576 images (this time included reading images from the hard drive).

¹Available from <ftp://pets.rdg.ac.uk/PETS2001/>

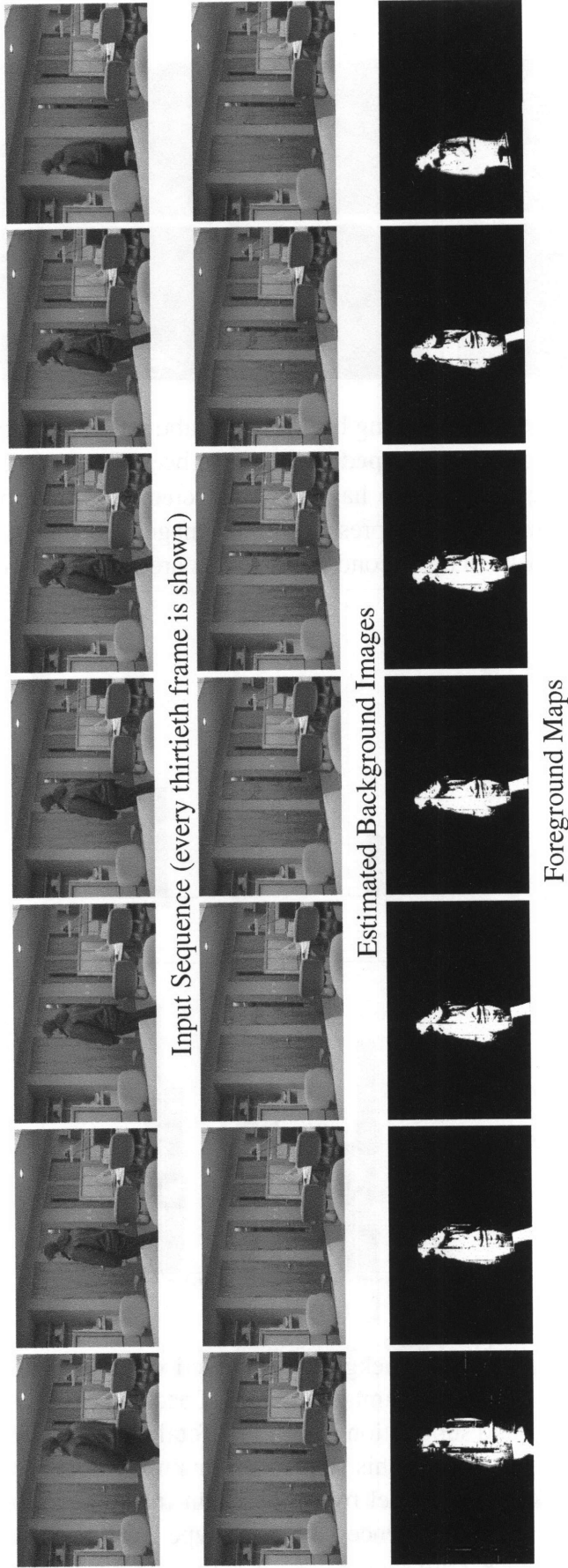


Figure 4-3: Fixing “sleeping man” problem. Performance of the dual-chain system on the sequence shown in Figure 2-6. Note that the correct background model and foreground maps are maintained while the person is stationary.

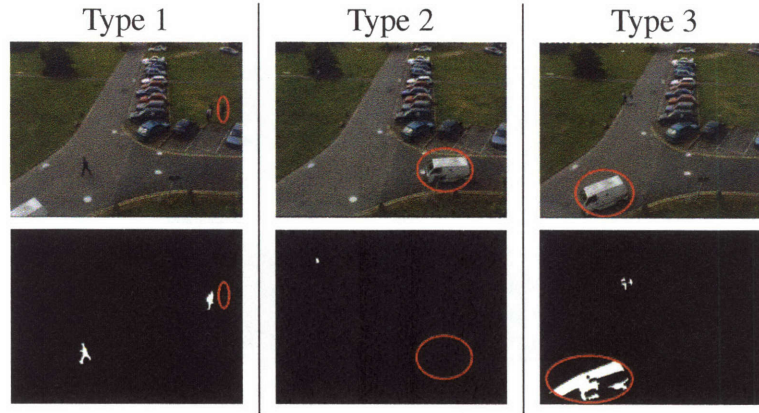


Figure 4-4: Error types used for evaluating background subtraction algorithms. 1: no foreground components corresponding to a **pedestrian** have been detected. 2: no foreground components corresponding to a **vehicle** have been detected. 3: foreground component detected when no foreground object is present. Input images are shown in the first, and erroneous segmentation masks in the second rows. Errors are circled in red.

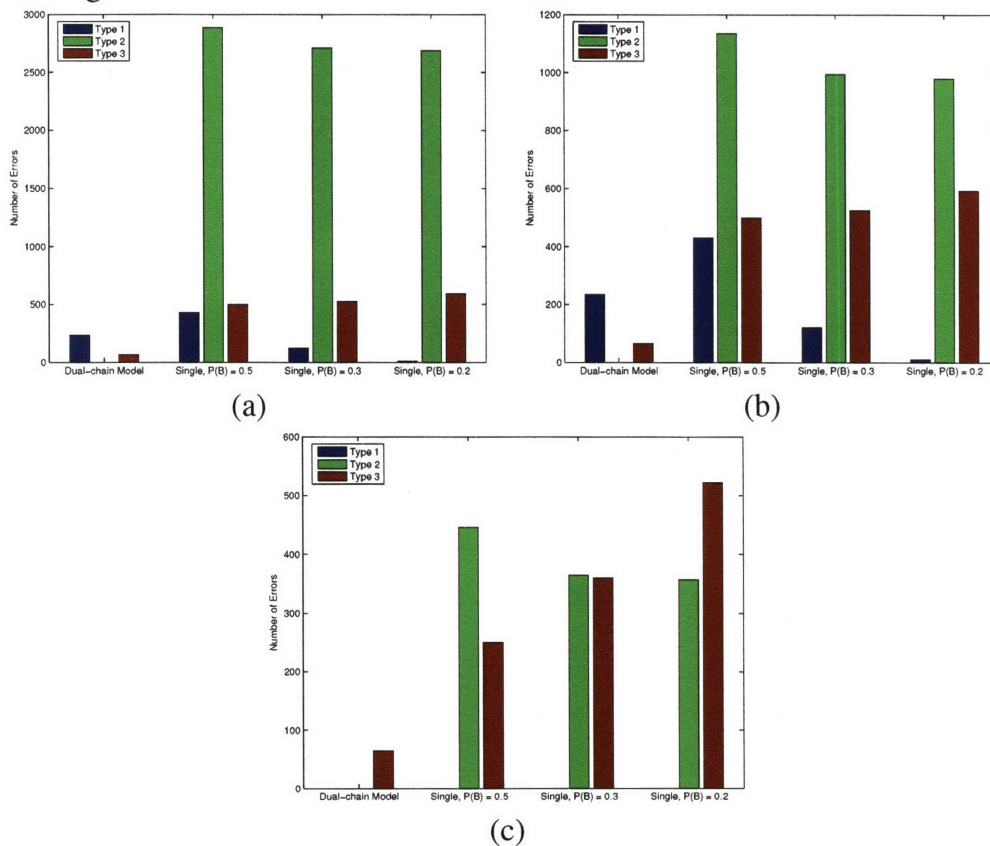


Figure 4-5: Quantitative evaluation of background subtraction performance on PETS 2001 image sequences. The charts show the number of errors of each type (described in Figure 4-4) produced by each background subtraction algorithm. Total number of errors in sequence 1 is presented in (a). Since one car in this sequence remains stationary after parking, its incorporation into the background model by single-chain trackers can be justified. The error chart in (b) shows results for sequence 1 ignoring type 2 errors corresponding to this car. Error statistics for sequence 2 are shown in (c). See the text for more details.

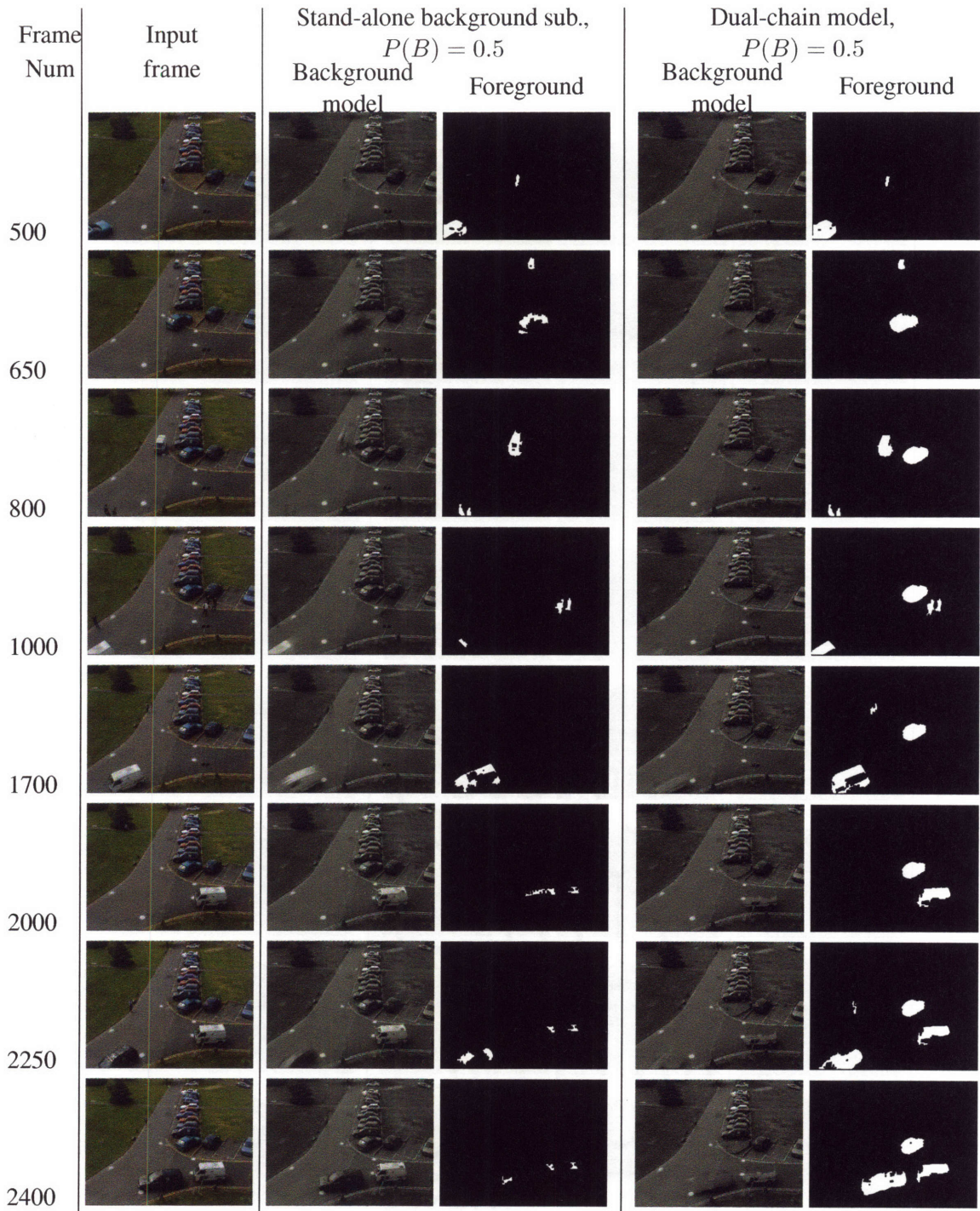


Figure 4-6: Qualitative comparison of background subtraction performance on one of PETS2001 image sequence. Second column holds input frames. Estimated background model and the computed foreground components are presented in the third and fourth columns for stand-alone background subtraction and in fifth and sixth columns for dual-chain model. Note that while input images are in color, all computations were performed in grayscale. See text for more details.

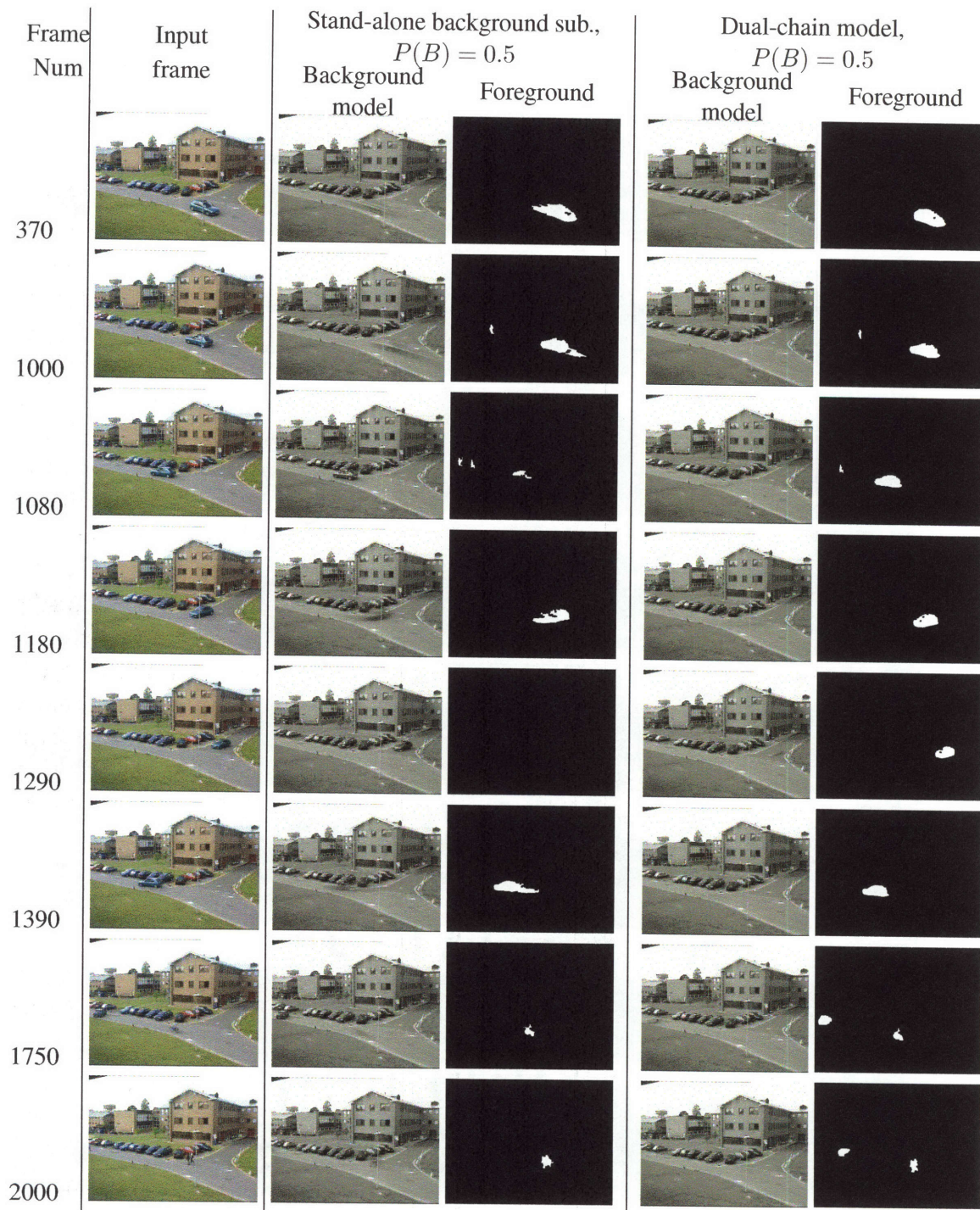


Figure 4-7: Qualitative comparison of background subtraction performance on the second PETS2001 image sequence. Second column holds input frames. Estimated background model and the computed foreground components are presented in the third and fourth columns for stand-alone background subtraction and in fifth and sixth columns for dual-chain model. Note that while input images are in color, all computations were performed in grayscale. See text for more details.

4.2 Structure from Motion Estimation

Estimating 3D structure of the world from one or a collection of 2D images is a task that is effortless for humans but is still extremely difficult for computers. Significant progress has been made over the last two decades of the task of Structure-from-Motion estimation (SFM). SFM is formulated as recovering 3D positions of the 2D feature points observed moving in the image plane. The original solutions assumed that the motion is caused either by the rigid motion of the observed object or, equivalently, by the motion of the camera observing the 3D world [70, 121]. These approaches have later been extended to multi-object and non-rigid motion cases [122, 17]. The key assumption of most SFM algorithms is that the 2D tracks of the features can be extracted from the image sequences.

Feature tracking is noisy and error-prone. It is often based on differential optical flow [121] or direct patch matching [70]. Each technique considers only a small number of pixels around a feature point, which may cause an individual track to drift or get completely lost due to such unmodeled behaviors as illumination changes. Feature tracking is done as a preprocessing step, and every feature point is considered independently; thus, the classically formulated SFM algorithms cannot use the global motion constraints to improve feature point tracking and the subsequent shape estimates.

Several ad-hoc methods that use the global motion to detect and terminate lost tracks have been proposed [52, 70]. In [70] the estimated feature positions were compared with ones predicted by the global model and ones that had significant differences were discarded. Local search was initialized from both global and local predictions of feature position in [52], and the best result was chosen. Neither of these methods was able to *recover* the track after occlusion, or had a consistent way of maintaining the uncertainty. In this section we describe how the RSMCM can be used to combine feature tracking and structure-from-motion recovery into a single, coherent probabilistic structure.

4.2.1 Prior Approaches to Structure from Motion Recovery

Estimating the shape of the objects from their observed motion has been an active area of research for the last twenty years (for an overview see [32, 33]). One can view it as an extension of multi-baseline stereo [85] or optical flow analysis [45] using information from the whole image. However, most researchers have taken a different approach of basing the estimates only on the salient features (e.g., corners and lines) detected and/or tracked through the sequence.

The recursive formulations proposed in [4, 6, 70] are based on simultaneously updating the shape and motion estimates at every frame using Kalman filter variants. A powerful tool for using feature point tracks for recovering structure from motion is the *factorization* approach [121, 75]. It is based on the fact that the matrix consisting of all the coordinates of the tracked feature points in all frames is low-rank (modulo image noise), and can be *factored* into low-rank shape and motion matrices. This approach has been extended to the cases of multi-body [130] and non-rigid motion [122]. A recursive algorithm based on factorization concept has also been proposed [74].

4.2.2 Factorization Algorithm Incorporating Temporal Coherence

Until recently, no factorization method has taken advantage of the fact that not only feature tracks should be smooth, but so should be the motion estimates corresponding to the neighboring frames. The first factorization approach to use this constraint is was proposed in [39]. In this section we review this algorithm in detail, as we will use it as a part of our RSMCM shape-from-motion formulation.

This algorithm is based on the factor analysis [95] view of factorization. Denoting the 3D position of the i th point as (x_i, y_i, z_i) , its projection at time t as (u_i^t, v_i^t) , and the first two rows of the homogeneous projection matrix at time t as $m^t = (m_1^t, \dots, m_8^t)$, the noisy projection equations for P points in T frames are written by [39] as

$$\begin{pmatrix} u_1^1 & \dots & u_P^1 & v_1^1 & \dots & v_P^1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u_1^T & \dots & u_P^T & v_1^T & \dots & v_P^T \end{pmatrix} = \begin{pmatrix} m_1^1 & \dots & m_8^1 \\ \vdots & \ddots & \vdots \\ m_1^P & \dots & m_8^P \end{pmatrix} A + \omega_{[T \times 2P]}, \text{ where} \quad (4.4)$$

$$A = \begin{pmatrix} S & & \\ \mathbf{1}_{1 \times P} & & \\ & S & \\ & & \mathbf{1}_{1 \times P} \end{pmatrix}, S = \begin{pmatrix} x_1 & \dots & x_P \\ y_1 & \dots & y_P \\ z_1 & \dots & z_P \end{pmatrix}, \omega_{ij} \sim N(0, \sigma_{ij}^2).$$

This equation is then solved using the standard EM algorithm for factor analysis. The temporal coherence in pose estimates is enforced by adding second-order smoothness constraints over camera-motion parameters m^t :

$$\begin{aligned} m^t &= m^{t-1} + \dot{m}^{t-1} + \frac{1}{2}\ddot{m}^{t-1} + \epsilon_1, & \epsilon_1 &\sim N(0, \sigma_{\epsilon_1}^2) \\ \dot{m}^t &= \dot{m}^{t-1} + \ddot{m}^{t-1} + \epsilon_2, & \epsilon_2 &\sim N(0, \sigma_{\epsilon_2}^2) \\ \ddot{m}^t &= \ddot{m}^{t-1} + \epsilon_3, & \epsilon_3 &\sim N(0, \sigma_{\epsilon_3}^2). \end{aligned}$$

The factor-analysis algorithm may be converted to inference in the single-chain model in Figure 4-1(b) by using $S^t = (\mathfrak{S}^t, m^t, \dot{m}^t, \ddot{m}^t)$, where $\mathfrak{S}^t = (x_1, y_1, z_1, \dots, x_P, y_P, z_P)$ and $\Theta^t = (u_1^t, \dots, u_P^t, v_1^t, \dots, v_P^t)$. The model dynamics are then

$$p(S^t | S^{t-1}) = \delta(\mathfrak{S}^t - \mathfrak{S}^{t-1}) N\left(\begin{pmatrix} m^t \\ \dot{m}^t \\ \ddot{m}^t \end{pmatrix}; \begin{pmatrix} \mathbf{1} & \mathbf{1} & \frac{1}{2} \\ 0 & \mathbf{1} & \mathbf{1} \\ & & \mathbf{1} \end{pmatrix} \begin{pmatrix} m^{t-1} \\ \dot{m}^{t-1} \\ \ddot{m}^{t-1} \end{pmatrix}, \Sigma_\epsilon\right), \quad (4.5)$$

where the first factor ensures the constancy of shape estimates across time and the second term describes the pose evolution. The feature generation model is

$$p(\Theta^t | S^t) = N(F^t; m^t A, \Sigma_\eta), \quad (4.6)$$

with A defined in Equation 4.4.

4.2.3 Point Feature Tracking

We have implemented a Kalman-filter-based feature point tracker in a manner similar to that of [70]. Since point tracking is a part of a batch process, it is possible to further smooth point tracks using an RTS smoother [89]. The points are tracked independently, and each point-tracker state is a five-tuple $R_i^t = (x_i^t, y_i^t, u_i^t, v_i^t, o_i^t, G_i^t)$, where (x_i^t, y_i^t) is the point's position at time t , (u_i^t, v_i^t) is its velocity, and G_i^t is the appearance model (a template). The point is observed at time t if $o_i^t = 1$ and occluded if $o_i^t = 0$. Point's state evolves according to the first-order dynamics,

$$\begin{pmatrix} x_i^t \\ y_i^t \\ u_i^t \\ v_i^t \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i^{t-1} \\ y_i^{t-1} \\ u_i^{t-1} \\ v_i^{t-1} \end{pmatrix} + \eta, \quad \eta \sim N(0, \Sigma_\eta), \quad (4.7)$$

$$p(o_i^t = 1 | o_i^{t-1} = 1) = 1 - \alpha$$

$$p(o_i^t = 1 | o_i^{t-1} = 0) = \alpha,$$

$$G_i^t = G_i^{t-1} + \nu, \quad \nu \sim N(0, \Sigma_\nu)$$

The state R of the low-level tracking model (Figure 4-1(a)) is a collection of individual point-tracker states, $R^t = \{R_i^t\}$, and the feature set Θ consisted of the 2D positions of the individual feature points and their appearances $\Theta = \{(x_i, y_i, G_i)\}$. The appearance templates are combined based on the feature-point positions, and are observed and image I^t contaminated by sensor noise. Note that while the feature model of the point tracker includes the appearance information, the high-level model does not. This does not prevent us from combining them into RSMCM, since the form of the Figure 3-5(a) can be used.

4.2.4 Implementation

The feature tracking and structure estimation chains were combined as described in Algorithm 2. At every iteration of the coordinate-ascent algorithm, prediction available from the global model was incorporated into the feature tracking process by replacing the Kalman prediction in the individual feature's prior by the product of the and the global motion prediction available from Equation 4.5. Rigid object shape and motion were then reestimated using feature tracks produced by the point tracker. The effect of this combination was two-fold: it reduced the point drift and allowed for more robust handling of occlusions. If the feature point became occluded (i.e., the peak correlation value was below the threshold), the uncertainty in its position quickly became too large and it was dropped by the stand-alone tracker, and a new track was started when the point became visible again. In the RSMCM, the high-level prediction was, in effect, providing a virtual observation, which would preserve the track for longer periods of time.

We have experimented with RSMCM extensions of both the pure factor-analysis based algorithm and a variant that enforced pose coherence. In order to quantitatively compare the performance of these algorithms, we have created a synthetic dataset that emulated the behavior of common feature trackers on real data. Forty points randomly distributed on a unit cylinder were observed for sixty frames by a camera moving with a constant angular

velocity. To emulate occlusions and misdetections, every point changed state from visible to invisible in each frame with probability $P(\textit{loose})$. To emulate template drift, consistent bias was introduced into each visible point for five frames with probability $P(\textit{drift})$.

Shapes recovered for $P(\textit{loose}) = 0.1, P(\textit{drift}) = 0.3$ are shown in Figure 4-11. The shapes computed by the single-chain variants contain more points. This is due to the fact that each point on the cylinder had produced several partial tracks separated by occlusions. The inability of a feature tracker to recognize partial tracks as belonging to a single feature complicated shape recovery. Since RSMCM methods are able to use the global model for data association, their shape estimates are much more accurate. Figure 4-10 illustrates the interaction of feature tracking and shape recovery modules in the RSMCM. The true path of a particular feature point and the observations available to the system are shown in the first column. The second column shows the output of single-feature tracker, and the third – a re-projected path of the corresponding point in the 3D shape through several iterations of the coordinate ascent algorithm. Note that the global constraints available to the shape-from-motion module enabled the system to track the point through occlusions, which in turn improved shape estimates.

A quantitative evaluation of this experiment is shown in Figure 4-12. The errors in individual feature trackers and structure-based predictions have been empirically verified to have low correlation, so, as we would expect from the analysis in section 3.2, RSMCM estimates have significantly lower errors than those from a feed-forward system. Note that the number of occlusions (related to $P(\textit{loose})$) had the greatest impact on the shape estimation. Neither of the single-chain approaches was able to deal with the multiple partial tracks observed for one feature point. Both failed to correctly recover the shape (signified by large re-projection errors), even for small values of $P(\textit{loose})$.

We have also used the experimental setup described above to verify that the analysis of Section 3.2 is applicable to this problem. As can be expected, each individual model (feature tracker and rigid motion) is an unbiased predictor of the feature point position. The plots of errors in x and y predictions by each model, evaluated for different levels of observation noise are shown in Figure 4-8. We have computed Γ indicator matrices for each system – Γ_1 was computed for the rigid motion model and Γ_2 for feature tracking model – for the corresponding noise levels. Their minimum eigenvalues are plotted in Figure 4-9. Both minimum eigenvalues are small for very low noise levels indicating that the RSMCM system does not offer much improvement over single-chain estimates. Under these circumstances the likelihood is close to the delta-function, and completely dominates the posterior distribution. The minimum eigenvalue of Γ_2 is always positive and increasing (and thus Γ_2 is positive definite), indicating that the feature-tracking model estimates benefit from the RSMCM combination. For low noise levels the minimum eigen value of Γ_1 is also increasing, as the local smoothness constraints improve coarser global-motion-model estimates. When the observation noise level is increased even further, the minimum eigenvalue of Γ_1 becomes negative since the local motion model becomes too uncertain, and the RSMCM combination is detrimental to the quality of high-level estimates.

The results of applying factor analysis with temporal coherence and its RSMCM variant to a fifty-frame video sequence² of a rotating box are shown in Figure 4-13. The initial

²We used part of an original sequence from <http://www.cs.ucla.edu/~hljin/research/voi.html>

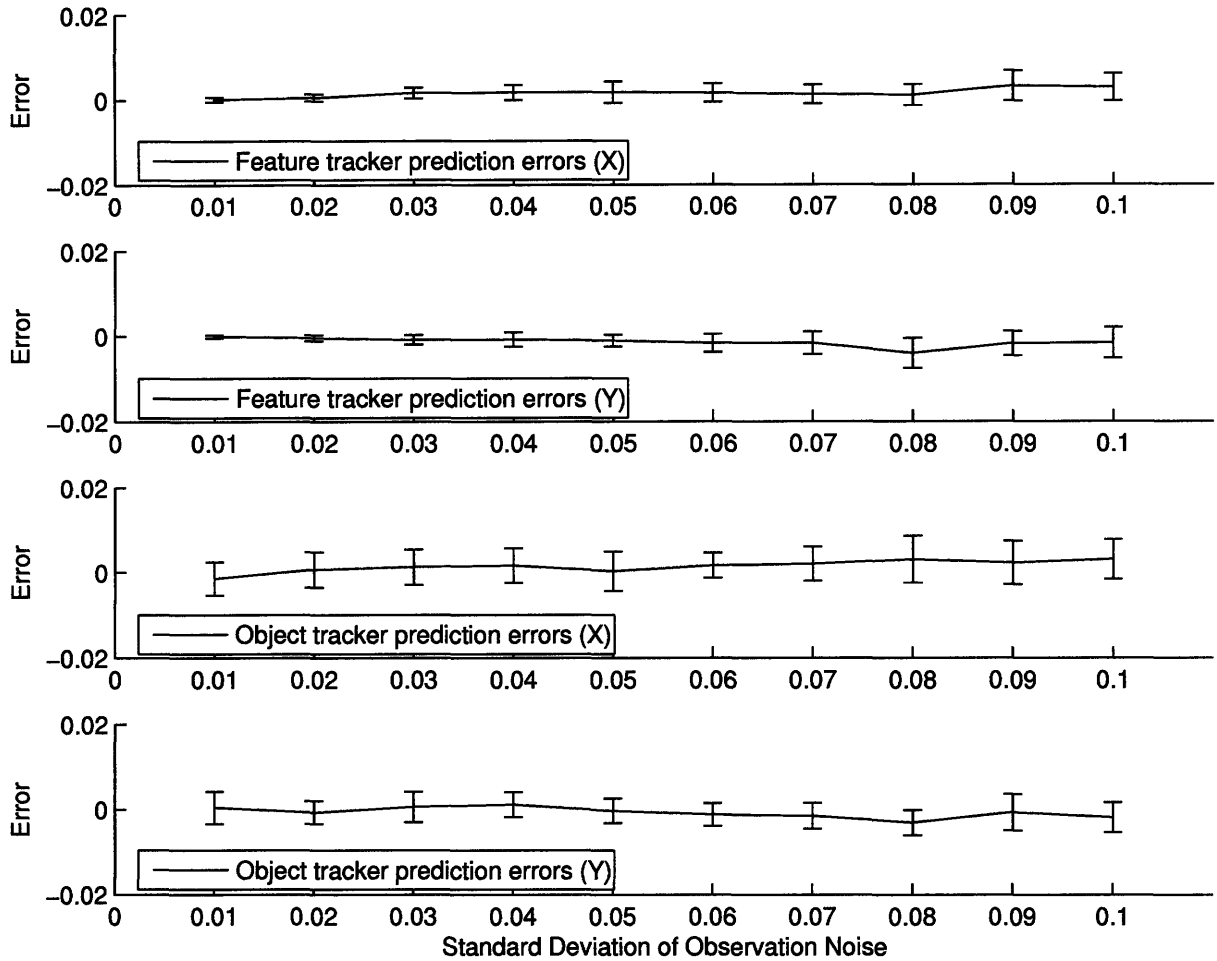


Figure 4-8: Prediction errors in structure-from-motion estimation. The prediction errors in x and y coordinates of local-feature tracking model (top plots) and rigid motion model (bottom plots) for different noise levels. The means and standard deviations were computed over 20 trials for each data point.

points of interest were located using Tomasi-Kanade feature point detector [120], and the 5×5 patches around the points were extracted. The points were then tracked using a first order Kalman filter, with the likelihood computed based on the normalized correlation scores around the location predicted by the filter. In our experiments we have not filtered out points with very short tracks before passing them to high-level inference algorithms – an approach frequently taken in feed-forward frameworks to reduce the effect of intermittent occlusions on the final shape estimate. As a result the shape recovered by stand-alone factor analysis contains many spurious points, exhibiting behavior we have observed in the synthetic experiments. The RSMCM framework, however, succeeded in estimating the correct shape.

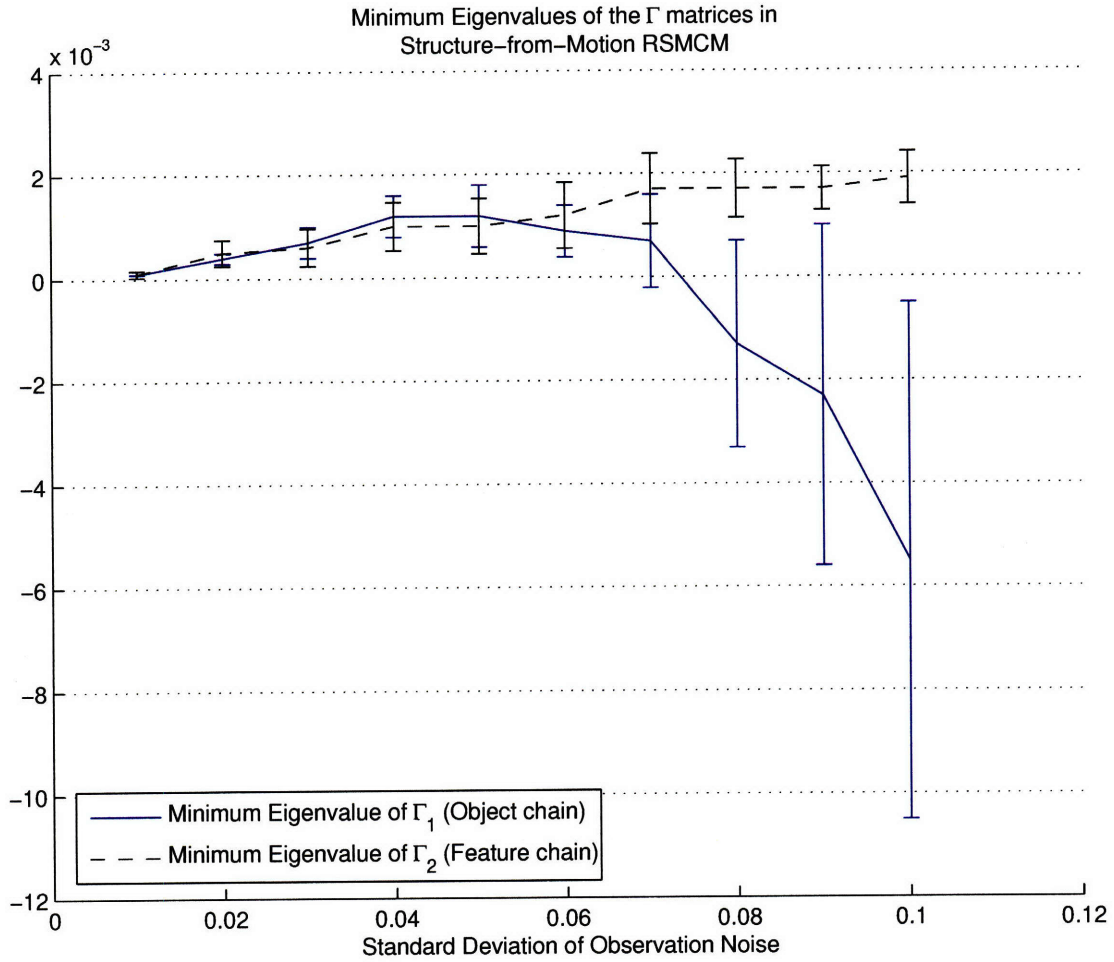


Figure 4-9: Minimum eigenvalues of the indicator matrices in RSMCM analysis computed for different noise levels. The minimum eigenvalue of Γ_2 corresponding to the local-feature tracking model is always positive, indicating that the local model always benefits from global constraints. The minimum eigenvalue of Γ_1 is negative for high noise levels, indicating that the over-emphasizing local smoothness can decrease quality of high-level estimates in these situations.

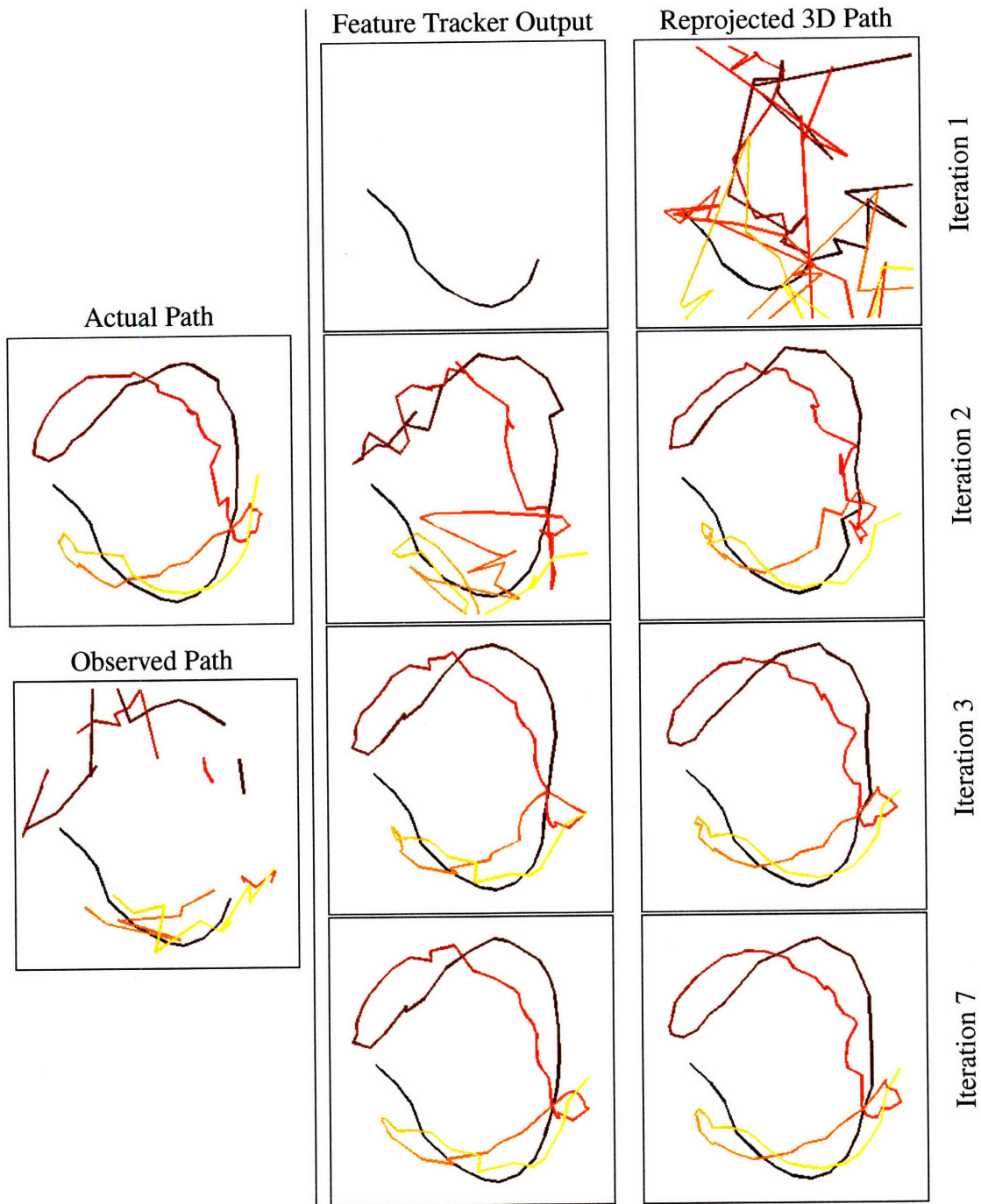


Figure 4-10: Improvement in feature tracking through iterations of coordinate ascent algorithm. First column: true path of a particular feature point, and its observations. Second and third columns: outputs of the feature tracking and structure from motion modules of the RSMCM described in Section 4.2. The lines are color-coded for convenience. Note that the feature tracking module was able to use output of SFM module to “stitch” the track broken by occlusion, which in turn enabled better global shape and motion estimates.

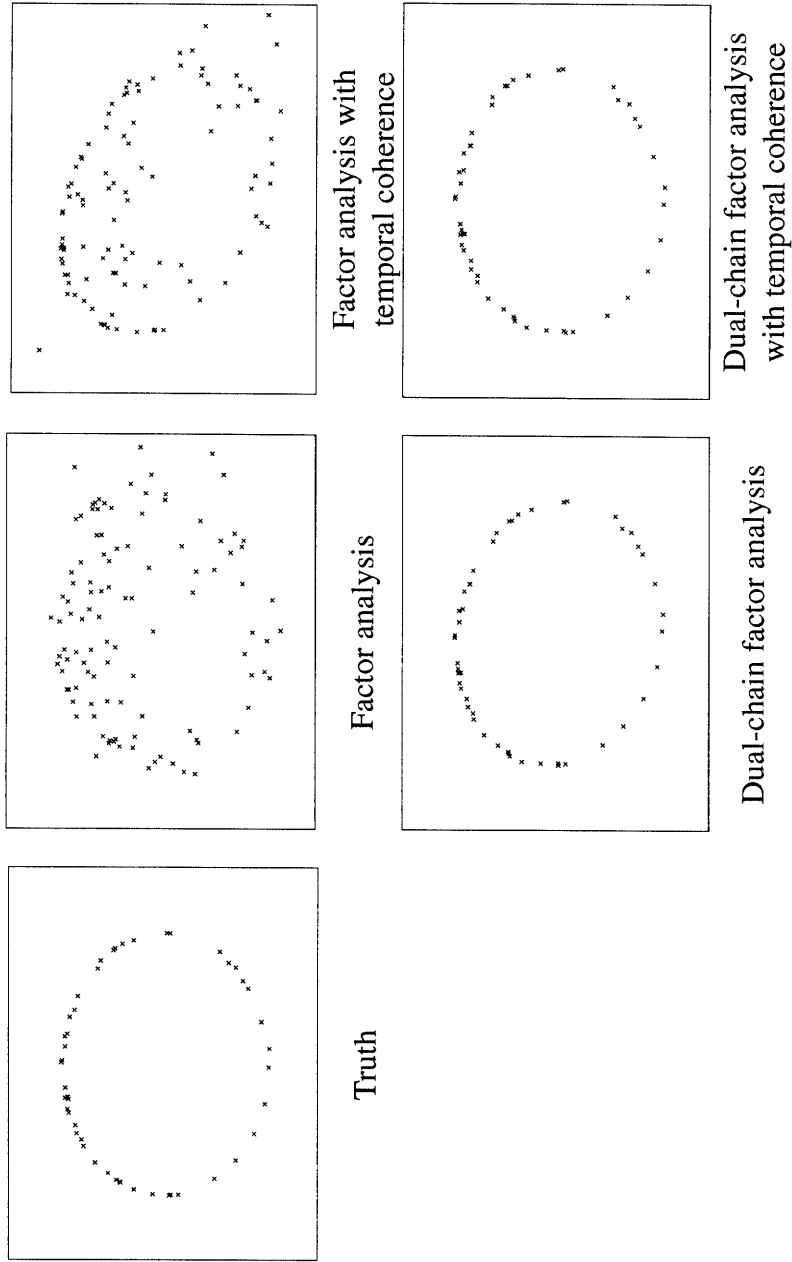


Figure 4-11: Comparison of typical performance of four structure-from-motion algorithms on a synthetic sequence ($P(loose) = 0.1$, $P(dirift) = 0.3$, see text for details). Single-chain methods produce much poorer results in the presence of occlusions due to their inability to establish correspondences between partial tracks of the same point.

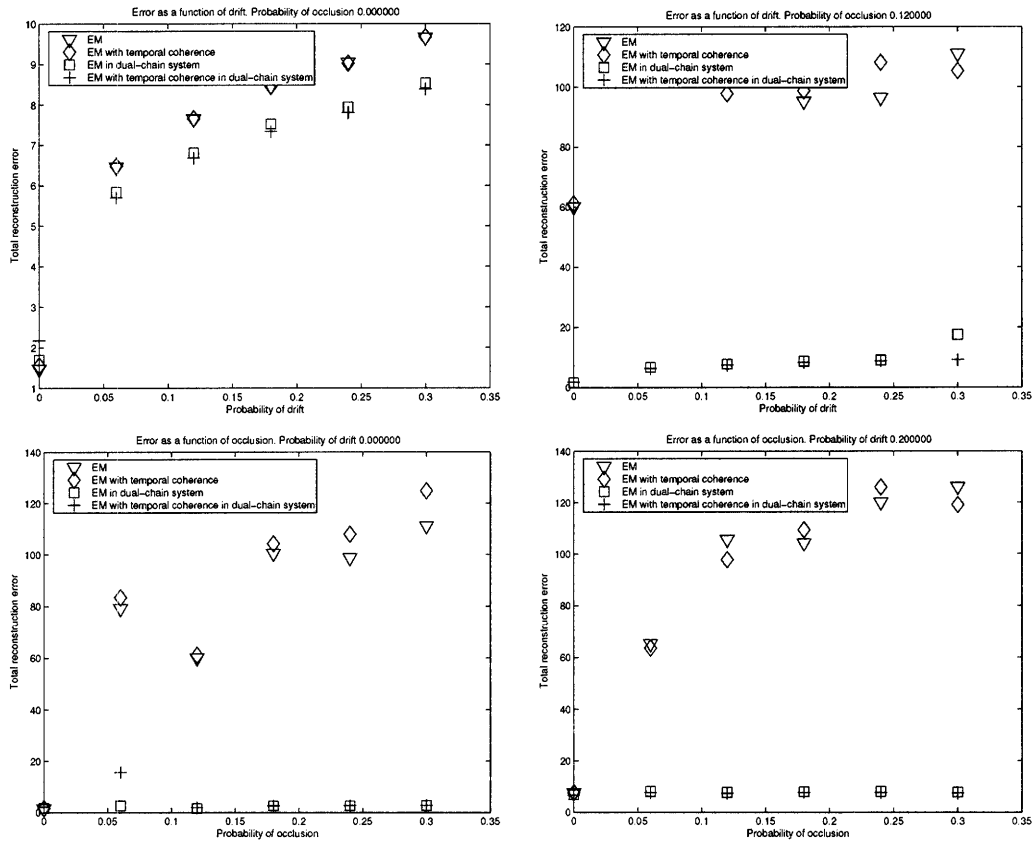


Figure 4-12: Quantitative comparison of structure-from-motion recovery algorithms on the synthetic sequence with varying amounts of drift and occlusion. Top row—total re-projection error as a function of drift with no occlusion, i.e., $P(\text{loose}) = 0$ (left) and with 12% chance of occlusion, i.e., $P(\text{loose}) = 0.12$ (right). Bottom row—total re-projection error as a function of occlusion for $P(\text{drift}) = 0$ (left) and $P(\text{drift}) = 0.2$ (right). Dual-chain algorithms were able to approximately reconstruct shape in all cases. Single-chain methods failed for even small values of $P(\text{loose})$.

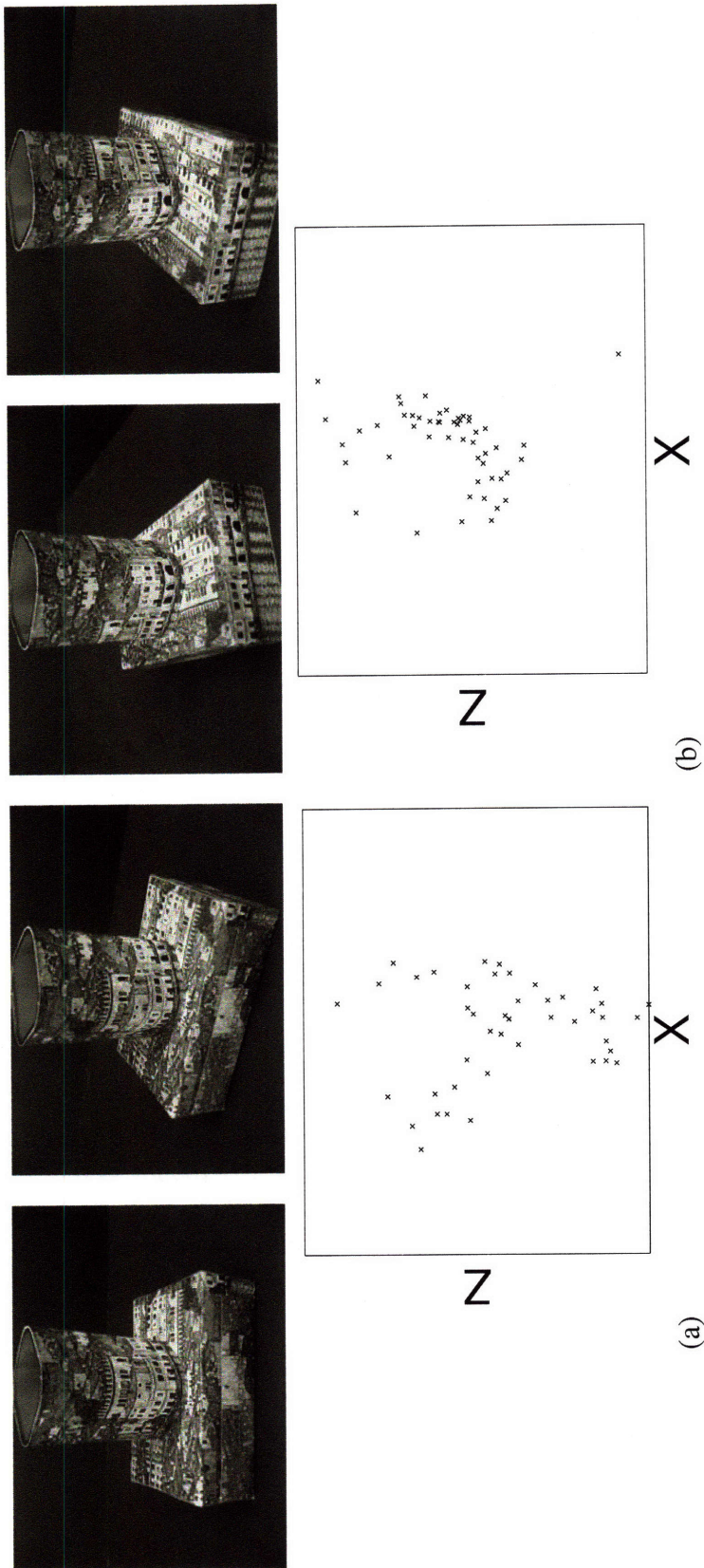


Figure 4-13: Comparing shape points computed by the stand-alone factor-analysis with temporal coherence and its dual-chain variant. Top row: four frames of the input video sequence. Bottom row: (a) view from above onto the top part of the shape produced by factor-analysis (b) view from above onto the top part of the shape produced by the dual-chain algorithm. Note that in the shape produced by factor analysis more than half of the points were spurious.

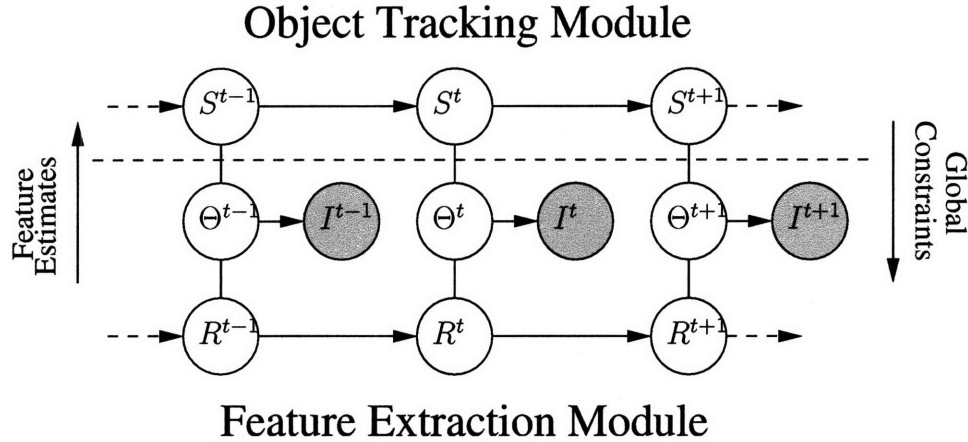


Figure 4-14: Module separation in RSMCM. The high-level (S) and low-level (R) inference algorithms (modules) can be designed and implemented separately. The only connection between modules are the (S, Θ) edges, along which the forward (feature estimate) messages $\mu_{\Theta^t \rightarrow S^t}$ are passed from feature extraction to object-level module, and feedback (global constraints) messages $\mu_{S^t \rightarrow \Theta^t}$ are passed another way.

4.3 Summary

In this chapter we have demonstrated using of the redundant state framework to combine probabilistic feature extraction and object tracking systems, resulting in significantly improved tracking results. We have shown that applying the approximate filtering algorithm (Section 3.3) to the model incorporating per-pixel adaptive background subtraction and object tracking systems results in improved segmentation results and thus better tracking performance. The shape estimates obtained from the RSMCM combination of high-level motion model and individual feature-point trackers was superior to the standard feed-forward system.

As shown in Figure 4-14, the RSMCM framework allows modular implementation of the hierarchical models. The only interaction between high- and low-level modules is through the (S, Θ) edges, along which forward and feedback messages flow.

Chapter 5

Likelihood Sampling and RSMCM for Articulated Body Tracking

In the preceding chapters we have presented approaches to coping with the deterioration of selectivity (i.e. ambiguity of posterior peaks) due to the use of uncertain dynamic model. In the applications we have considered, the state space was relatively low dimensional. This makes it possible to search the region covered by the relatively wide prior: the difficulty lies in deciding which likelihood peak corresponds to the true observation and which is due to unmodeled noise. In the applications such that require density estimation in high dimensional spaces, e.g. articulated body tracking, the search aspect becomes most important. Even if the likelihood function has only one peak, finding it in the large region with significant temporal prior probability is a computationally intensive task.

In this chapter we present a *likelihood sampling* approach to articulated pose tracking. Our method is based on using lower level features such as hand and face positions to explore the likelihood surface and using dynamics to provide temporal smoothing. The danger of locking onto the wrong peak in the likelihood and losing track is reduced since our tracker does not rely in the temporal prior to guide the search. Likelihood sampling assumes that the output of the low-level face and hand detectors or trackers is available, but the constituent trackers are often brittle. We improve the performance of the combined system by incorporating articulated-body- and low-level trackers into a redundant state model. In the following chapter we will present a body pose tracking method for the cases when low-level trackers are not available.

5.1 Introduction

Articulated body tracking has been traditionally approached from a differential tracking perspective [99, 10, 13, 16]. The trackers combine dynamics, prior pose information and the current frame data to estimate pose (or a pose distribution) for the current frame. This approach suffers from several common drawbacks, most critically error accumulation over time and the need for manual initialization. A complimentary approach is to track using the repeated application of a single image pose estimation technique at every frame [35, 93, 73, 94]. However, neither of these methods uses the pose information from previous

frames and only estimate a single “best” pose that corresponds to the current observed image. Sequences of such estimates do not always correspond to correct dynamics due to the ambiguities that arise from projecting 3D bodies onto 2D images.

Our probabilistic tracking framework incorporates features of both approaches. For a single frame, the distribution of articulated pose parameters is estimated from static observations; with multiple frames, pose posteriors are propagated through time using a Bayesian technique. Our framework uses information from the current observation early in the inference process which improves the tracking stability when a strong dynamics model is not available.

Since parametric modeling of pose distributions is not feasible, we represent them with weighted sample sets. In our system, single frame pose parameter distributions are estimated using an importance sampling technique [66]. We represent image likelihood functions using a generative model of body appearance (described in Section 5.2.1). Proposal distributions are automatically constructed from image measurements, kinematic constraints, and parameter priors (Section 5.2.2). Pose distribution samples for the current frame are evaluated with respect to a sampled representation of the prior pose distribution, producing a pose posterior conditioned on all observed data (Section 5.2.6). Since the pose is sampled at each frame independently, our system does not require initialization and is able to gracefully recover from tracking failures. Propagation over time ensures the temporal continuity of the pose estimate.

5.2 Tracking with Likelihood Sampling

5.2.1 Upper Body Model

We model the human upper body with the articulated model in Figure 5-1. The model configuration at time t is described by the parameter vector (ϕ, Θ^t) , where $\phi \in \mathcal{R}^7$ contains time independent metric parameters – neck and upper and lower arm lengths, body width, depth and length, and head size; $\Theta^t = (\theta_1^t, \dots, \theta_6^t) \in \mathcal{R}^{16}$ contains pose parameters (three rotational degrees of freedom at neck and each shoulder, one at each elbow, and five global position parameters)¹. Since we assume that the observed images are formed using orthographic projection, the global depth parameter is ignored.

The parameters of articulated joints $\{\theta_i^t\}$, metric parameters ϕ and segment appearances $\{A_i\}$ are modeled as independent. The pose of the i th segment, P_i , is deterministically computed from the pose of its parent in the articulated tree, denoted P_{p_i} and appropriate joint parameters θ_i . Segment appearance A_i and pose P_i are combined to produce a segment latent image L_i . Poses P_i are also used to compute the binary support maps M_i for each segment (note that if the segment is not occluded, the support map depends only on the corresponding pose).

Due to the deterministic nature of the above steps, the following conditional pdfs used in the graphical model become delta functions with the factorization of $q(\cdot)$ implied by the Bayesian Network in Figure 5-3.

¹A global scale parameter is subsumed into individual length parameters.

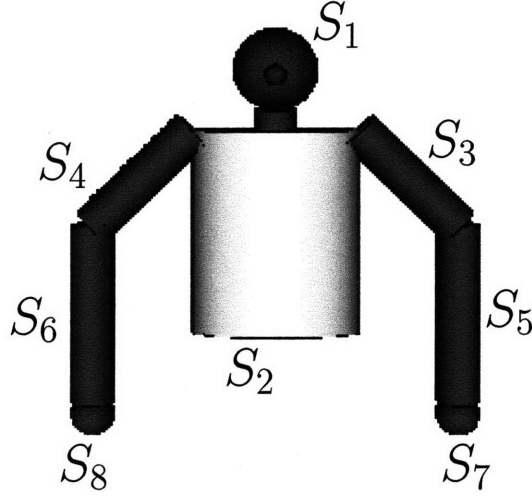


Figure 5-1: Articulated model of human upper body used in this work. The model consists of the head (S_1), torso (S_2), and two arms with upper and lower arm segments (S_3, S_5 and S_4, S_6 respectively) and hands (S_7 and S_8). The model configuration includes 7 metric parameters: head radius, neck length, body width (distance between shoulder joints) length and depth, and upper and lower arm lengths. The pose is specified by 16 parameters: 11 internal rotational parameters (3 degrees of freedom at the neck, 3 degrees of freedom at each shoulder, and 1 at each elbow) and 5 global degrees of freedom (2 translational and 3 rotational).

The combined latent images (masked by their regions of support) corrupted by uncorrelated Gaussian noise are then observed as \mathbf{I} ,

$$p(\mathbf{I}(x, y) | M_i, L_i) = N(\mathbf{I}(x, y); (\sum_i (M_i \cdot L_i)(x, y)), \sigma^2(x, y)) \quad (5.1)$$

The joint probability of the described model may then be factored as

$$p(\phi, \{\theta_i\}, \{P_i\}, \{A_i\}, \{L_i\}, \{M_i\}, \mathbf{I}) = p(\phi) \prod_i p(\theta_i) \prod_i p(A_i) \prod_i p(P_i | P_i, \theta_i, \phi) \quad (5.2)$$

$$\prod_i p(L_i | P_i, A_i) \prod_i p(M_i | \{P_i\}) p(\mathbf{I} | \{L_i\}, \{M_i\})$$

This generative model is described by a graphical network in Figure 5-2.

In our model, the i th segment is “responsible” for the region of the observed image \mathbf{I} that corresponds to its support map M_i . Let us define the i th observation region

$$\mathbf{I}_i = \mathbf{I} M_i = (\sum_i M_i L_i + \nu) M_i = L_i M_i + \nu M_i \quad (5.3)$$

Since support maps M_i are disjoint, the observation regions are independent, conditioned on $\{M_i\}$ and $\{L_i\}$. Furthermore, analyzing the conditional $p(\mathbf{I}_i | P_i, A_i, M_i)$, we find

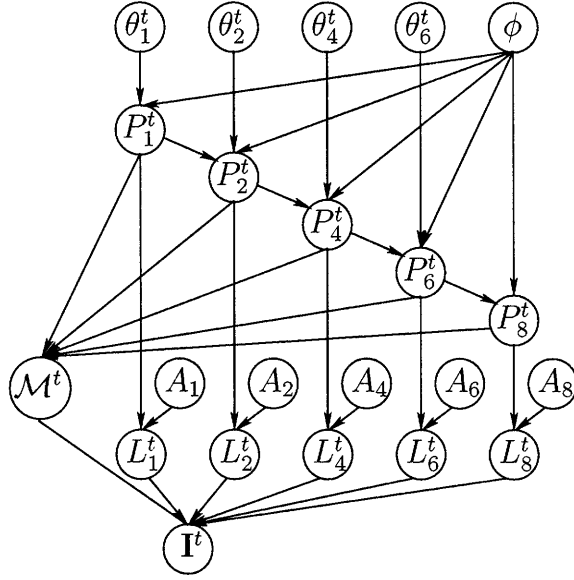


Figure 5-2: Generative model for an articulated body image (see Equation 5.2). The subscripts correspond to segment number as in Figure 5-1 (the nodes corresponding to S_3, S_5 and S_7 are symmetric to S_4, S_6 and S_8 and are not shown). The segment pose at time t , P_i^t depends on the pose of a parent segment $P_{p_i}^t$, body lengths ϕ , and corresponding joint parameters θ_i^t . θ_1 contains the global position parameters. The latent image of a segment, L_i^t is obtained by transforming appearance A_i according to the pose P_i^t . The observed image \mathbf{I} depends on the latent images masked by support maps $\mathcal{M}^t = (M_1^t, \dots, M_8^t)$, which are, in turn, determined from all segment poses.

that

$$\begin{aligned}
 p(\mathbf{I}_i | P_i, A_i, M_i) &= \int_{L_i} p(\mathbf{I}_i | L_i, M_i) p(L_i | P_i, A_i) \\
 &= \int_{L_i} p(L_i M_i + \nu M_i | L_i, M_i) \delta(L_i - f_i^l(P_i, A_i)) \\
 &= \prod_{x,y \in M_i} N(\mathbf{I}_i(x, y); (f_i^l(P_i, A_i))(x, y), \sigma^2(x, y))
 \end{aligned} \tag{5.4}$$

allowing us to use the simplified image generation model that induces the following joint pdf factorization

$$\begin{aligned}
 p(\phi, \{\theta_i\}, \{P_i\}, \{A_i\}, \{M_i\}, \{\mathbf{I}_i\}) &= p(\phi) \prod_i p(\theta_i) \prod_i p(A_i) \prod_i p(P_i | P_{p_i}, \theta_i, \phi) \\
 &\quad \prod_i p(M_i | \{P_i\}) \prod_i p(\mathbf{I}_i | \{P_i\}, \{A_i\}, \{M_i\})
 \end{aligned} \tag{5.5}$$

This equation can be used to evaluate image likelihoods of poses generated using our single frame pose sampling framework.

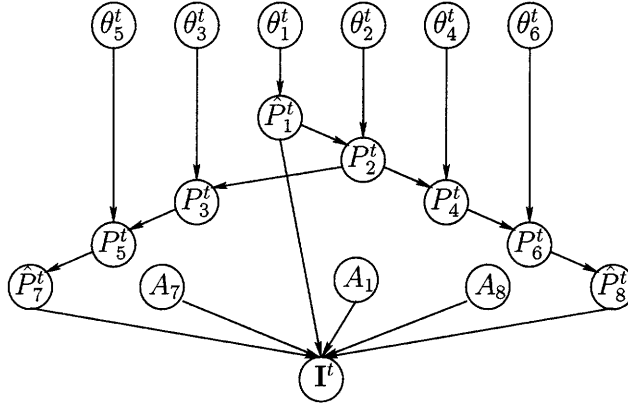


Figure 5-3: Generative appearance model used in defining a proposal distribution (see Equation. 5.7).

5.2.2 Likelihood Sampling

We wish to infer a distribution of articulated pose parameters from a single frame. We assume that segment appearances $\{A_i\}$ and prior distributions of parameters $p(\theta_i)$ are known, and wish to describe the posterior distribution $p(\Theta|\{A_i\}, \mathbf{I})$. In the following discussion we assume that the metric parameters of the model (ϕ_0) are also known and concentrate on sampling the posterior distribution of the pose parameters $p(\Theta^t|I^t)$. We address estimation of ϕ in Section 5.2.5. Using Bayes' rule, independence assumptions, and Equation 5.2, the pose posterior distribution may be expressed as

$$p(\Theta|\{A_i\}, \phi_0, \mathbf{I}) \sim p(\mathbf{I}|\Theta, \{A_i\}, \phi_0)p(\Theta) \quad (5.6)$$

The complexity of natural images makes it hard to specify this distribution analytically. While evaluating the posterior (up to a scaling factor) at any particular Θ_0 is relatively easy, sampling from it (necessary for tasks such as providing input to an articulated tracker) is hard. The alternative approach is to use Monte Carlo methods and represent $p(\Theta|\{A_i\}, \mathbf{I})$ as a set of samples with attached weights $\{\Theta_i, \pi_i\}$. One such method is importance sampling [66].

In the importance sampling framework, instead of sampling a target distribution $p(x)$, a *proposal* distribution $q(x)$ that approximates $p(x)$ is sampled, and then the weight of the sample x_k is set to $\pi_k = \frac{p(x_k)}{q(x_k)}$. A reasonable choice of a proposal distribution used in this technique should “concentrate” the samples in the areas of configuration space with high values of target distribution.

This approach is similar in spirit to [62] that used MCMC sampling incorporating observation-based proposals. The major difference is that while [62] uses single-body-part proposals, we sample from proposal distribution that combines multiple low-level body-part hypothesis that jointly satisfy inverse kinematics constraints.

5.2.3 Proposal Distribution

Our approach to constructing a proposal distribution is based on the assumption that partial pose information for certain segments in the model may be extracted directly from the image. That is, it is possible to efficiently sample from the conditional $p(\hat{P}_i|A_i, \mathbf{I})$, where \hat{P}_i contains partial information about P_i . In our system such segments are the head and hands (segments 1, 7, and 8), that are easy to detect in the image. The appropriate models are discussed in Section 5.2.5.

We define our proposal distribution as

$$\begin{aligned} q(\Theta|A_1, A_7, A_8, \phi_0, \mathbf{I}) &= q(\Theta|\hat{P}_1, \hat{P}_7, \hat{P}_8, \phi_0)q(\hat{P}_1|A_1, \mathbf{I})q(\hat{P}_7|A_7, \mathbf{I})q(\hat{P}_8|A_8, \mathbf{I}) \quad (5.7) \\ &= q_{\text{head}}(\theta_1|\hat{P}_1, \phi_0)q_{\text{neck}}(\theta_2|\theta_1, \hat{P}_7, \hat{P}_8, \phi_0)q_{\text{left arm}}(\theta_3, \theta_5|\theta_1, \theta_2, \hat{P}_7, \phi_0) \\ &\quad q_{\text{right arm}}(\theta_4, \theta_6|\theta_1, \theta_2, \hat{P}_8, \phi_0)p(\hat{P}_1|A_1, \mathbf{I})p(\hat{P}_7|A_7, \mathbf{I})p(\hat{P}_8|A_8, \mathbf{I}) \end{aligned}$$

We sample from q in five steps:

1. Obtain head and hands pose samples \hat{P}_1^s , \hat{P}_7^s , and \hat{P}_8^s from the appropriate distributions
2. Compute global parameters θ_1^s from \hat{P}_1^s
3. Obtain neck joint configuration θ_2^s from q_{neck}
4. Obtain left arm configuration (θ_3^s and θ_5^s) by sampling from $q_{\text{left arm}}$
5. Obtain right arm configuration (θ_4^s and θ_6^s) by sampling from $q_{\text{right arm}}$

5.2.4 Kinematic Constraints

We would like to specify $q_{\text{neck}}(\cdot)$, $q_{\text{left arm}}(\cdot)$, and $q_{\text{right arm}}(\cdot)$ based on image information, joint parameter priors and kinematic constraints.

The samples of the distributions conditioned on the image, $\hat{P}_1^s = (\Omega, x_{p_1}, y_{p_1})^T$, $\hat{P}_7^s = (x_{p_7}, y_{p_7})^T$, and $\hat{P}_8^s = (x_{p_8}, y_{p_8})^T$ are the orientation and image position of the head and hands (Section 5.2.5).

Without loss of generality we define the world coordinate system to have x and y axes parallel to image axes, and z axis passing through the origin of the image plane. Then the external parameters of the articulated model are simply $\theta_1 = (\Omega, x_{p_1}, y_{p_1})^T$ and $P_1^s = f_1^p(\theta_1, \phi_0)$.

Let us define a *feasible* configuration of shoulder pose $P_2 = f_2^p(\theta_2, f_1^p(\theta_1))$ and image plane hand locations \hat{P}_7, \hat{P}_8 to be one in which it is possible to reach each hand from a corresponding shoulder, that is, the image plane distance from the shoulder joint to the hand is less than the arm-length. Then, if we disallow all infeasible configurations, we can define q_{neck} as

$$q_{\text{neck}}(\theta_2|\theta_1, \hat{P}_7 = \hat{P}_7^s, \hat{P}_8 = \hat{P}_8^s, \phi_0) \sim \begin{cases} p(\theta_2) & (f_2^p(\theta_2, f_1^p(\theta_1)), \hat{P}_7^s, \hat{P}_8^s) \text{ is feasible} \\ 0 & \text{otherwise} \end{cases} \quad (5.8)$$

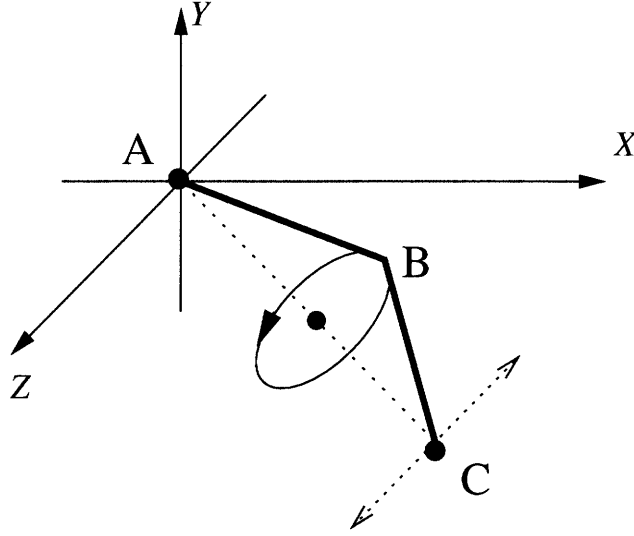


Figure 5-4: When the 3D position of the shoulder **A** and position of the hand **C** in the xy -plane are known, the arm has two degrees of freedom, depth of the hand and rotation of the elbow **B** about the shoulder-hand line.

Let us consider left arm as a two link assembly shown in Figure 5-4. The shoulder pose $P_2^s = f_2^p(P_1^s, \theta_2^s)$ uniquely determines the position of the left shoulder joint **A** = $(x_A, y_A, z_A)^T$. The position of the left hand, **C** is known up to the translation along z axis

$$\mathbf{C} = (x_{P_7^s}, y_{P_7^s}, z_C)^T \quad (5.9)$$

$$z_A - r \leq z_C \leq z_A + r \quad (5.10)$$

$$r = \sqrt{(l_{\text{upper arm}} + l_{\text{lower arm}})^2 - (x_A - x_{P_7^s})^2 - (y_A - y_{P_7^s})^2}$$

where limits in Equation 5.10 ensure that the distance between the shoulder joint and the hand is not greater than the total arm-length ($l_{\text{upper arm}} + l_{\text{lower arm}}$). The whole assembly may then be rotated about the line **AC** by $0 \leq \psi < 2\pi$. The configuration of the arm $\Theta_l = (\theta_3, \theta_5)^T$ is then uniquely determined by ψ and z_C (i.e. $\Theta_l = g(z_C, \psi, P_1, \theta_2, \hat{P}_7)$). We model $q_{\text{left arm}}$ as

$$q_{\text{left arm}}(\theta_3, \theta_5 | \theta_1, \theta_2, \hat{P}_7, \phi_0) = p(\theta_3, \theta_5 | z_C, \psi, \theta_2, \theta_2, \hat{P}_7, \phi_0) \quad (5.11)$$

$$p(z_C | P_1, \theta_2, \hat{P}_7, \phi_0) p(\psi | P_1, \theta_2, \hat{P}_7, \phi_0) p(\theta_3) p(\theta_5)$$

$$= \delta(\Theta_l - g(z_C, \psi, P_1, \theta_2, \hat{P}_7, \phi_0))$$

$$p(z_C | P_1, \theta_2, \hat{P}_7, \phi_0) p(\psi | P_1, \theta_2, \hat{P}_7, \phi_0) p(\theta_3) p(\theta_5)$$

where

$$p(z_C | P_1, \theta_2, \hat{P}_7, \phi_0) = u(z_C; z_A - r, z_A + r) \quad (5.12)$$

$$p(\psi | P_1, \theta_2, \hat{P}_7, \phi_0) = u(\psi; 0, 2\pi) \quad (5.13)$$

The corresponding proposal distribution for the right arm, $q_{\text{right arm}}$, is defined in the

same fashion.

Once the sample $\Theta = (\theta_1, \dots, \theta_6)$ is selected, we need to determine its weight $\pi = \frac{p(\Theta|\{A_i\}, \phi_0, \mathbf{I})}{q(\Theta)}$.
Note that

$$\begin{aligned} q_{\text{left arm}}(\Theta_3, \Theta_5 | \theta_1, \theta_2, \hat{P}_7, \phi_0) & \quad (5.14) \\ &= \int \delta(\Theta_l - g(z_C, \psi, P_1, \theta_2, \hat{P}_7, \phi_0)) p(z_C | P_1, \theta_2, \hat{P}_7, \phi_0) \times \\ & \quad \times p(\phi_0 | P_1, \theta_2, \hat{P}_7, \phi_0) p(\theta_3) p(\theta_5) dz_C d\psi \\ & \sim \begin{cases} p(\theta_3) p(\theta_5) & \text{if configuration is valid} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

And thus, if we have obtained a sample $\Theta = (\theta_1, \dots, \theta_6)$ from $q(\cdot)$, then

$$q(\theta_1, \dots, \theta_6 | A_1, A_7, A_8, \phi_0, \mathbf{I}) = p(\hat{P}_1 | A_1, \mathbf{I}) p(\hat{P}_7 | A_7, \mathbf{I}) p(\hat{P}_8 | A_8, \mathbf{I}) \prod_{i=2}^6 p(\theta_i) \quad (5.15)$$

and the weight is given by

$$\begin{aligned} \pi &= \frac{p(\Theta | \{A_i\}, \phi_0, \mathbf{I})}{q(\Theta | \{A_i\}, \phi_0, \mathbf{I})} & (5.16) \\ &= \frac{p(\theta_1) \prod_i p(\mathbf{I}_i | \{P_i\}, \{A_i\}, \{M_i\})}{p(\hat{P}_1 | A_1, \mathbf{I}) p(\hat{P}_7 | A_7, \mathbf{I}) p(\hat{P}_8 | A_8, \mathbf{I})} \end{aligned}$$

By processing a frame \mathbf{I}^t using the algorithm described in this section, we obtain a sample set $\{(\Theta_i^t, \pi_i^t)\}$ representation of $p(\Theta^t | \mathbf{I}^t)$ that may then be used for tracking or estimating the Maximum Likelihood pose at the current timestep.

5.2.5 Single Frame Pose Estimation Implementation and Results

The description of our algorithm is completed by specification of the parameter priors $p(\theta_i)$, appearance A_i , and image formation models. We also need to address recovering metric model parameters ϕ .

We have obtained the joint angle limits from [76], and have represented shoulder and elbow angle prior probabilities as uniform between those limits. The neck angle prior was specified as a broad Gaussian centered on the origin.

For our method to be practical, the image formation models $p(\mathbf{I}_i | P_i, A_i, M_i)$ have to be efficient to evaluate, and, in the case of the head and hands, lead to simple-to-sample-from posteriors $p(\hat{P}_i | \mathbf{I}, A_i)$. Many general techniques are possible. Here we use simple implementations flesh color and face pattern detection.

Our general framework requires the estimate of the head pose $p(\hat{P}_1 | \mathbf{I}, A_1)$. Ideally, we would use a face detector that is capable of detecting faces that have orientations other than frontal, while reporting size, location and orientation. For most of the experiments in this paper (other than Figure 5-6(b)) we assume that the person is in the upright position facing

the camera, so we estimate $p(\hat{P}_1|\mathbf{I}, A_1)$ based on the output of a 2D frontal face detector (we use the method described in [125]). The detector output is a set of image squares that are reasonably well centered on the faces, and the distribution is represented as a mixture of Gaussians,

$$p(\hat{P}_1|\mathbf{I}, A_1) = \sum_f N(\hat{P}_1; c_f, \begin{pmatrix} 0.05r_f^2 & 0 \\ 0 & 0.05r_f^2 \end{pmatrix}) \quad (5.17)$$

where c_f is a center of the detected square, and r_f is half of its width.

The face square size is also used to estimate the distribution of the metric parameter vector ϕ . We use anthropometric data from [76] combined with empirically estimated ratio of r_f to head radius to define the means and standard deviations of Gaussian distributions from which elements of ϕ is drawn. Once the tracker has settled in, we replace the original metric prior $p(\phi)$ by the new prior $p_e(\phi)$ that is computed from the body sizes estimated in the previous frames.

We model hands as flesh-colored blobs. The flesh color segmentation is performed on the input image using a detector initialized from the middle region of the face rectangle (Figure 5-5(c)). Connected components are then computed from the resulting binary image. All components that either overlap the face rectangle, are larger than it in one of the dimensions, or have an area smaller than 10% of the face rectangle area are filtered out. The hand pose posterior $p(\hat{P}_7|\mathbf{I}, A_7) = p(\hat{P}_8|\mathbf{I}, A_8)$ is then approximated as mixture-of-Gaussians where each constituent Gaussian distribution is initialized from one of the remaining components (Figure 5-5(d)).

We model elongated segments in the model (torso, lower and upper arms) as cylinders, and use the intensity gradient as an image measurement. Along the contour of the segment's image plane projection (cf. Figure 5-5(f, h)), we expect the gradient to be perpendicular to the edge, and to have high magnitude [90, 92]. Let G_i^α be the gradient direction image, E be the set of the points on the predicted edges under the support map, and α_0 be the predicted gradient direction. Then we define the image likelihood function as an average match along the predicted edges,

$$p(\mathbf{I}_i|P_i, A_i, M_i) \sim \frac{1}{|E|} \sum_{(x,y) \in E} e^{\cos 2(G_i^\alpha(x,y) - \alpha_0)} \quad (5.18)$$

Sampling from Gaussian and uniform distributions is implemented using direct methods [66]. The distributions defined in Equations 5.8 and 5.11 are sampled by discretizing the parameter space, assigning each discrete sample s_i weight w_i proportional to the value of the appropriate pdf ($q_{\text{neck}}(s_i \dots)$ or $q_{\text{arm}}(s_i)$ respectively), and then drawing a sample from the produced weighted sample set $\{(s_i, w_i)\}$.

We have applied our algorithm to a set of images of people in natural settings. Various stages of the algorithm are shown in Figure 5-5. The face rectangle detected in the input image (a) is shown in (b). Raw flesh color segmentation results and filtered image used to construct hand position distribution are (c) and (d). Panes (e) and (f), and (g) and (h) contain sample pose and corresponding edges overlaid on gradient magnitude image.

The results of applying our single frame pose detection algorithm to a set of four images is shown in Figure 5-6. For each of the examples, we present 20 random samples from the

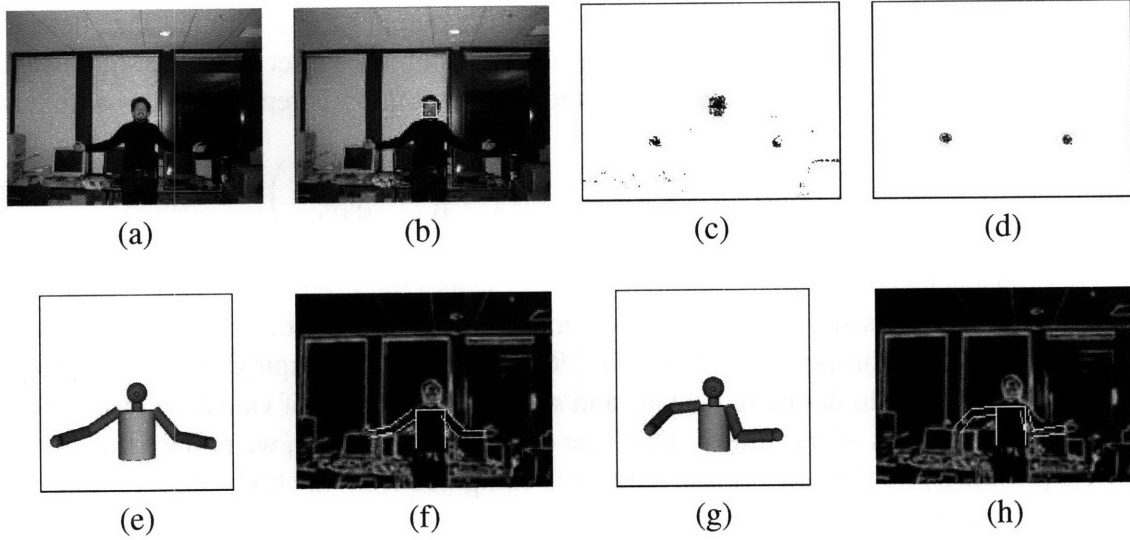


Figure 5-5: Stages of processing input image (a). The face rectangle (b) was located using a face detector [125], and the flesh color map (c) was computed by a detector initialized from the color distribution in the face rectangle. The result (d) of filtering the raw binary map (Section 5.2.5) was used to initialize the hand position distribution. Two sample poses with corresponding test edges overlaid on gradient image are shown in (e, f) and (g, h). The pose (e) was determined to be more likely than (g).

posterior pose distribution overlaid over the source image and the 3D reconstruction of the maximum likelihood particle. The head region and global transformation for the profile view (b) were manually initialized due to the lack of profile face detector. Despite gross estimation errors in some samples, reporting a distribution of poses as opposed to a single result allows a higher level process to use additional information (such as motion or context) to select the most appropriate one.

An example of the algorithm's failure is shown in Figure 5-7. The image likelihood computation is confused by the strong background gradients, which results in incorrect pose estimation.

5.2.6 Pose Propagation Over Time

When a motion sequence rather than a single frame are available, we would like to combine the pose estimate at the current frame with the previous observations, to produce a posterior distribution $p(\Theta^t | I^0 \dots I^t)$. We make a Markovian assumption that all information about observations $I^0 \dots I^t$ is preserved in distribution of Θ^{t-1} , that is $p(\Theta^t | \Theta^{t-1}, I^0 \dots I^{t-1}) = p(\Theta^t | \Theta^{t-1})$. We can then express the full posterior as

$$\begin{aligned}
 p(\Theta^t | I^0 \dots I^t) &\sim p(I^0 \dots I^{t-1} | \Theta^t) p(I^t | \Theta^t) p(\Theta^t) \\
 &\sim \frac{p(\Theta^t | I^t)}{p(\Theta^t)} \int p(\Theta^t | \Theta^{t-1}) p(\Theta^{t-1} | I^0 \dots I^{t-1}) d\Theta^{t-1}
 \end{aligned} \tag{5.19}$$

Algorithm 3 Sampling based articulated pose tracking

for all $t \geq 0$ **do**
 EXTRACT image features such as face position and flesh-colored blobs from the frame I^t .
 CONSTRUCT a proposal distribution $q^t(\Theta^t)$ from the extracted features and pose parameter priors.
 GENERATE N^t samples $\{(\Theta_i^t, \pi_i^t) | 1 \leq i \leq N^t\}$ from the proposal distribution with corresponding weight computed as $\pi_i^t = p(\Theta_i^t | I^t) / q(\Theta_i^t)$.
 if the prior $p(\Theta^{t-1} | I^0 \dots I^{t-1})$ is available **then**
 GENERATE samples $\{(\Theta_i^t, \lambda_i^t) | 1 \leq i \leq N^t\}$ from $p(\Theta^t | I^0 \dots I^t)$ by evaluating
 $\lambda_i^t = \pi_i^t \sum_{j=1}^{N^{t-1}} \lambda_j^{t-1} p(\Theta^t = \Theta_i^t | \Theta^{t-1} = \Theta_j^{t-1}) / p(\Theta_i^t)$
 UPDATE $p(\phi)$ from $\{(\Theta_i^t, \lambda_i^t) | 1 \leq i \leq N^t\}$
 else
 OUTPUT $\{(\Theta_i^t, \pi_i^t)\}$ as the estimate of $p(\Theta^t | I^0 \dots I^t)$
 end if
end for

As the shape of the posterior distribution is similar to the shape of the likelihood, we can use the pose samples $\{\Theta_i^t\}$ from $p(\Theta^t | I^t)$ to represent the full posterior. The extra information present in the temporal prior would be encoded in the new weights λ_i^t . If we assume that the prior $p(\Theta^{t-1} | I^0 \dots I^{t-1})$ is also represented with a set of weighted particles $\{(\Theta_j^{t-1}, \lambda_j^{t-1})\}$ obtained at the previous iteration of the algorithm, λ_i^t may be computed as

$$\lambda_i^t = \frac{\pi_i^t}{p(\Theta_i^t)} \sum_{j=1}^{N^{t-1}} \lambda_j^{t-1} p(\Theta^t = \Theta_i^t | \Theta^{t-1} = \Theta_j^{t-1}) \quad (5.20)$$

The complete tracking algorithm is presented in Algorithm 3.

5.3 Tracking in a Redundant State Framework

While a tracker based on likelihood sampling can successfully operate with a small number of samples and is self recovering, it is extremely sensitive to feature detector failures (such as flesh-color misdetections). In this work, we combine a likelihood-sampling tracker with low-level flesh-blob trackers and face trackers. The state of each part tracker contains positions and velocities of the corresponding part, The part trackers and the likelihood-sampling based tracker share latent features corresponding to 2D part positions (\hat{P}_1, \hat{P}_7 , and \hat{P}_7), and the observation models described in Section 5.2.5. They are combined into the RSMCM of the form shown in Figure 3-5(a), and the online inference algorithm is used.

We have applied our RSMCM tracker to three sample sequences, with results shown in Figures 5-8, 5-9, and 5-10. For each frame in the sequence, we have rendered forty randomly drawn samples from the posterior state distribution (the frontal view overlaid on top of the input image is shown in the middle row, and side view is shown in the bottom

row). In most frames, the tracker succeeded in estimating poses that contained significant out of plane components and self occlusions, and was able to recover from mistracks (e.g., around frame 61 in the third sequence).

In Figure 5-11, we compare the performance of the enhanced RSMCM tracker using 1,000 samples per frame (first column), likelihood-sampling tracker using 1,000 samples (second column), CONDENSATION tracker with 5,000 samples that runs as fast as the RSMCM tracker (third column), and finally CONDENSATION tracker with 15,000 samples (the smallest number of samples that enables CONDENSATION to perform with accuracy approaching RSMCM tracker performance). The results are presented using the same method as in Figure 5-8, the frontal view is shown overlaid on top of the input image, with the side view to the right of it.

The RSMCM tracker was able to successfully track the body through the entire sequence. The likelihood-sampling tracker was generally able to correctly estimate the pose distribution, but failed on frames where image features were not correctly extracted (cf. frames 20, 60, etc.). While CONDENSATION was using the same observation model as the RSMCM tracker (using both flesh color and edges), the CONDENSATION variant with 5,000 samples failed after 30 frames due to sample impoverishment (note that only a few distinct samples were drawn in frames 40 and later). Increasing the size of the sample set to 15,000 (with similar increase in running time) allowed CONDENSATION to successfully track through most of the sequence.

Our method improves upon likelihood-sampling, and compares favorably with the CONDENSATION algorithm in two ways. First, a monolithic approach using CONDENSATION requires a significantly greater number of samples in order to explore the configuration space sufficiently as compared to the RSMCM with likelihood sampling. Secondly, in the experiments presented the estimate of the posterior state distribution more accurately represents the uncertainty of the upper-body pose than the alternative methods.

We assumed that constituent low-level detectors or trackers were available – a significant limitation – the following chapter addresses this drawback.

5.4 Summary

We have presented a technique for sampling human upper body pose posterior distribution from single images, and its application to tracking. In our approach, the kinematic constraints and image information are incorporated at early stages of inference process, which allows us to reduce the number of samples needed to approximate the high-dimensional articulated body distributions.

We use importance sampling with proposal distribution is constructed from prior probabilities of joint angles obtained from anthropometric data, inverse kinematics constraints, and from image face and hand locations detected by well-known methods. The observation likelihoods are represented using a novel Bayesian network description of a generative appearance model that also explicitly incorporates kinematic constraints. The distribution is propagated in time using Bayesian methods and Monte Carlo integration.

The redundant state methodology have been empirically demonstrated to improve the behavior of the likelihood sampling tracker by incorporating information from the body

pose estimates into the face and hands trackers that are in turn used to guide pose estimation. The combined method compares favorably with the well-known CONDENSATION algorithm in two ways. First, a monolithic approach using CONDENSATION required a significantly greater number of samples in order to explore the configuration space sufficiently as compared to the multi-chain method. Secondly, and perhaps more importantly, in the experiments presented the estimate of the posterior state distribution more accurately represents the uncertainty of the upper-body pose than the alternative methods.

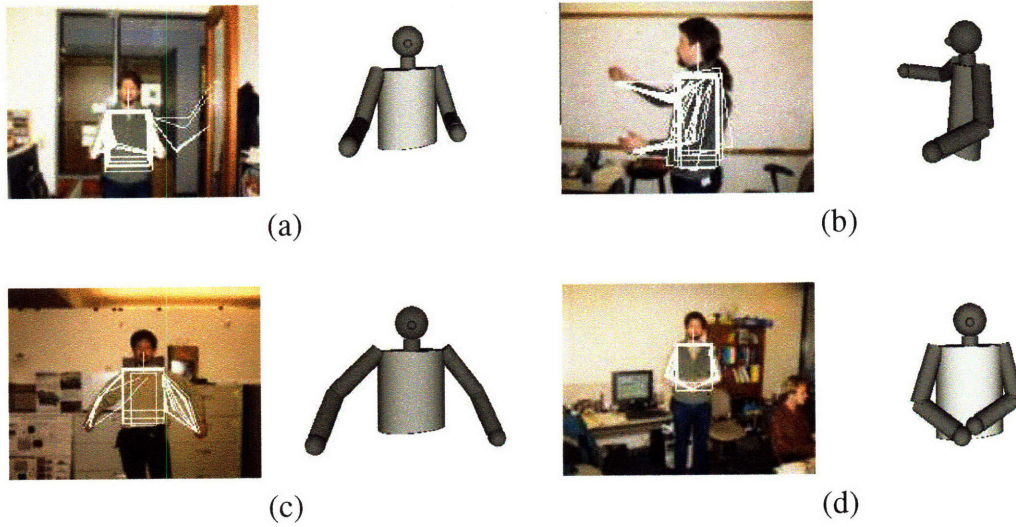


Figure 5-6: Sampling pose from a single image. For each example, twenty samples from the estimated posterior are shown overlaid on the image and the maximum likelihood pose is shown in 3D.

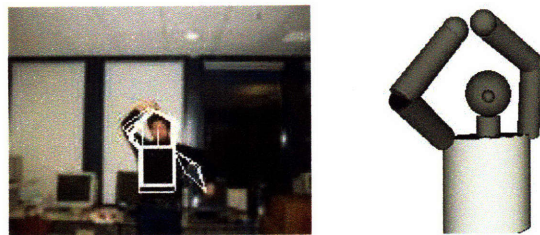


Figure 5-7: An example of the failure of pose estimation on a still frame. The algorithm was confused by the strong background image gradients, which resulted in assigning high probabilities to incorrect poses. Use of dynamic information in video mitigates such errors.

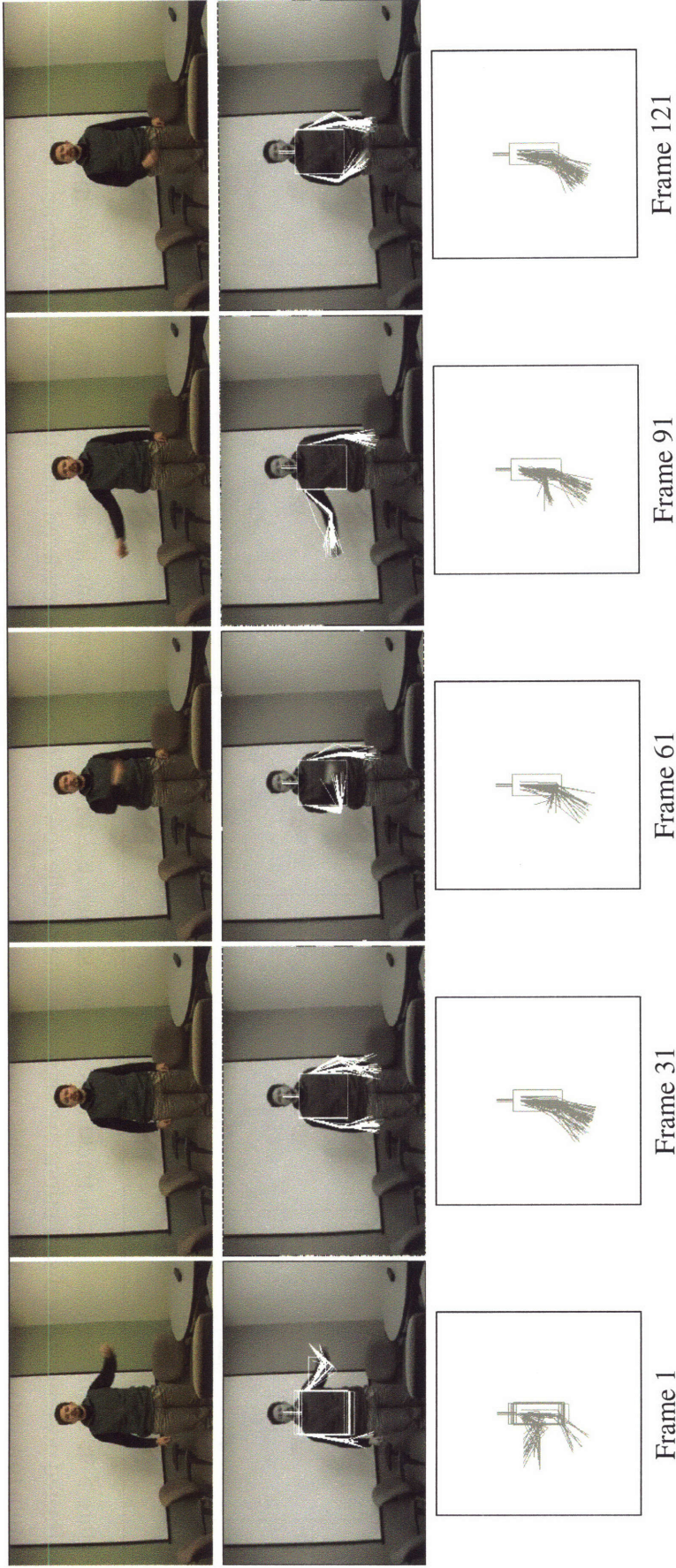


Figure 5-8: Applying dual-chain tracking to sample sequence 1. The top row contains input frames. Forty random particles from the estimated posterior pose distributions are shown: in the middle row, the particles are rendered onto the input image (frontal view); in the bottom row they are rendered in the side view.

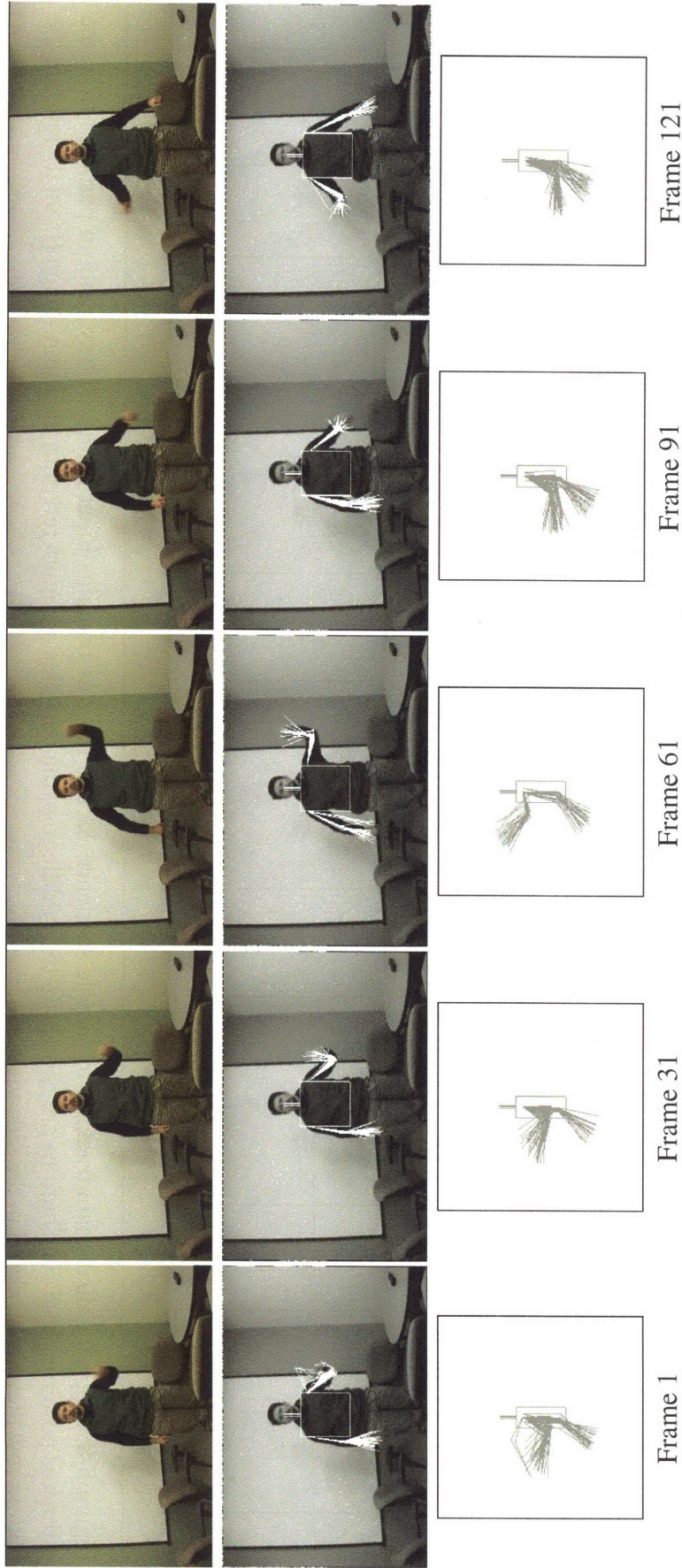


Figure 5-9: Applying dual-chain tracking to sample sequence 2. The top row contains input frames. Forty random particles from the estimated posterior pose distributions are shown: in the middle row, the particles are rendered onto the input image (frontal view); in the bottom row they are rendered in the side view.

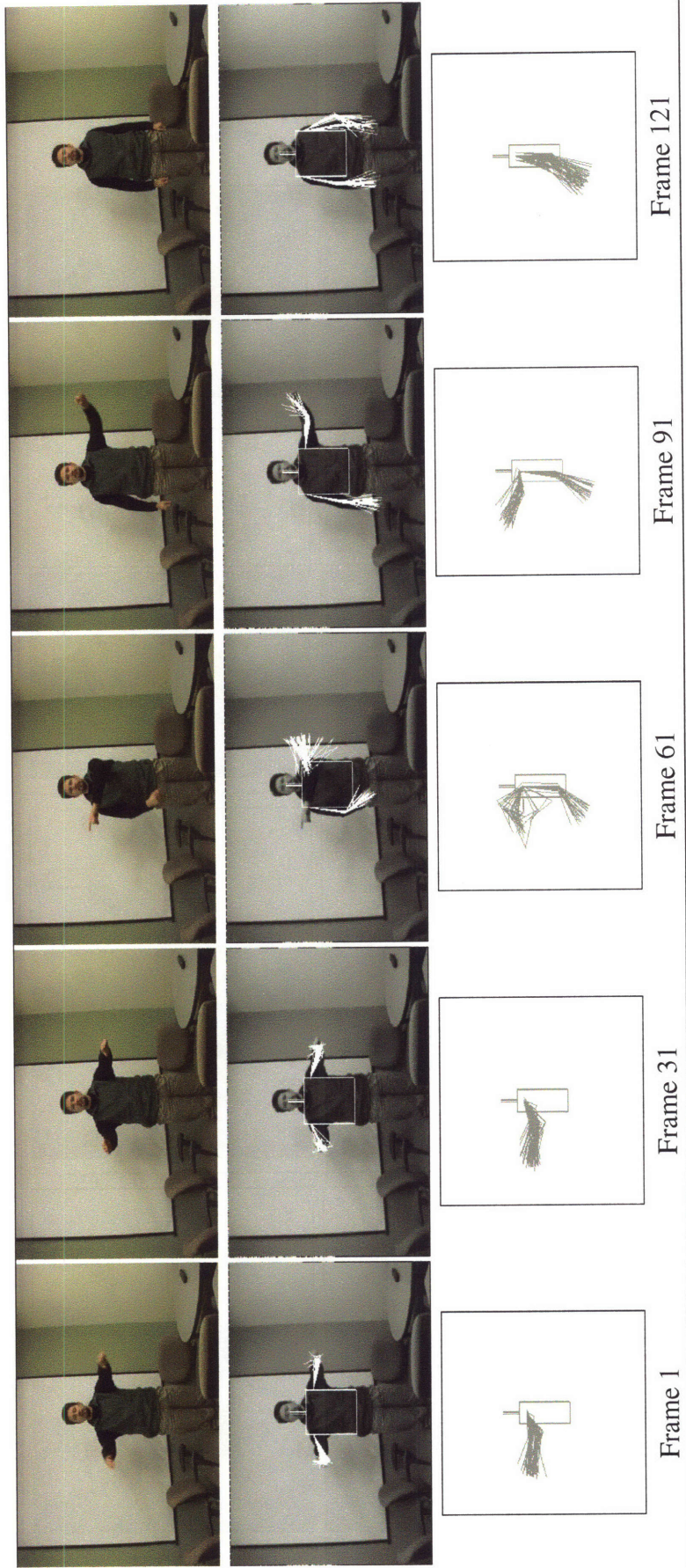


Figure 5-10: Applying dual-chain tracking to sample sequence 3. The top row contains input frames. Forty random particles from the estimated posterior pose distributions are shown: in the middle row, the particles are rendered onto the input image (frontal view); in the bottom row they are rendered in the side view. Note that while a mistrack has occurred on the third sequence near frame 61, the tracker was able to recover.

	Multi-chain Tracker with 1000 samples	Likelihood Sampling tracker with 1000 samples	CONDENSATION tracker with 5000 samples	CONDENSATION tracker with 15000 samples
1				
6				
11				
20				
24				

Continued on the next page

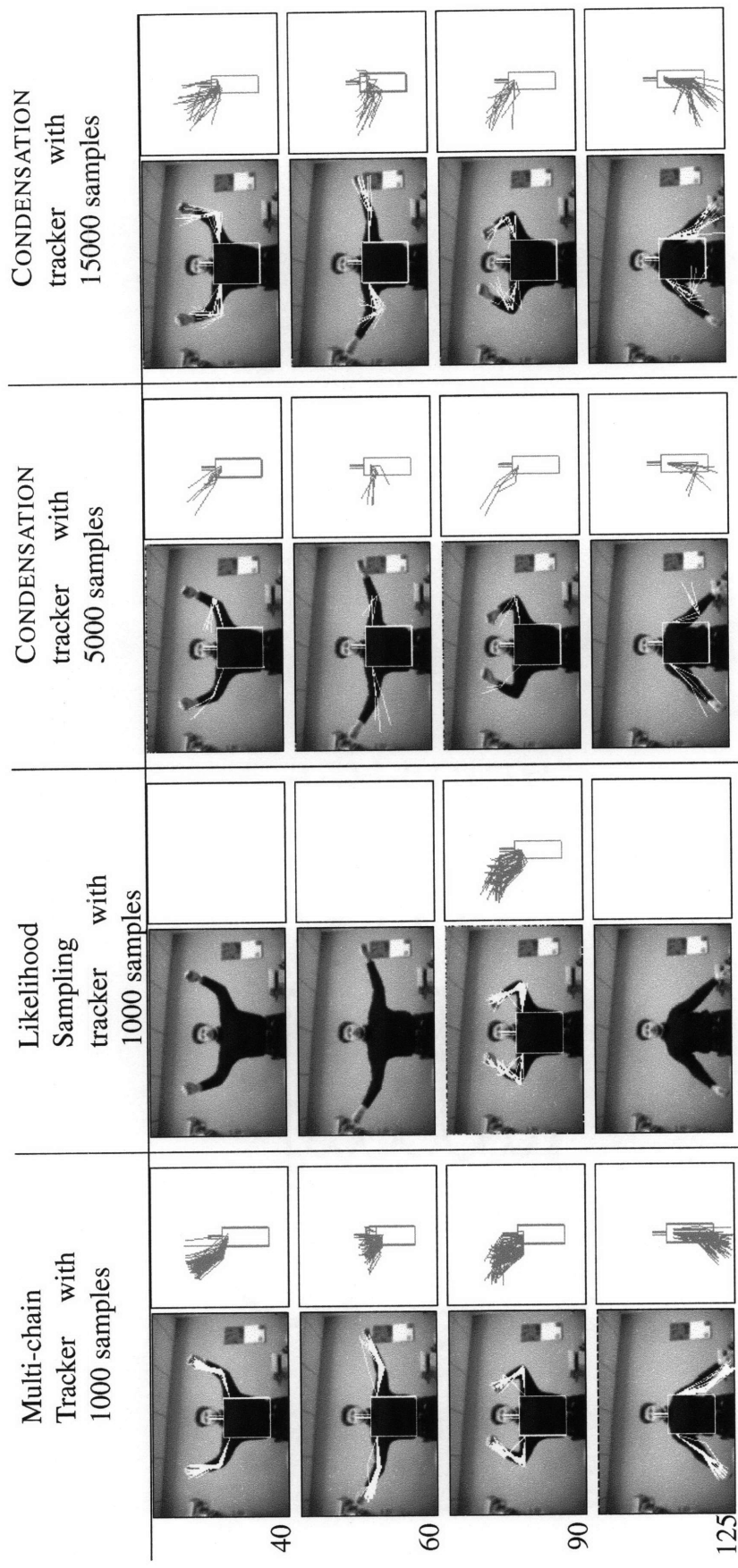


Figure 5-11: Applying four tracking algorithms to a sample sequence. For each frame a set of forty random pose samples were drawn from estimated posterior distribution and the corresponding skeletons were rendered (frontal view overlaid on the frame and side view below). Errors in feature detection caused likelihood-sampling tracker to fail on some of the frames (no samples were produced).

Chapter 6

Exploring Complex Distributions with Discrete Search

In the previous chapter we have described one method for coping with time complexity of searching for the peaks in the observation compatibility function – likelihood sampling. While it is appropriate in certain situations, it depended on a handcrafted proposal distribution that may not be available for a large set of tracking problems and is not sufficiently efficient for real-time tracking.

In this chapter we explore a different approach to the same problem – using search in sets of possible observations labeled with corresponding model states to quickly explore the pose-observation compatibility surface in large regions of the state space.

6.1 Introduction

As the amount of memory available to computers increases, it becomes viable to precompute and keep in memory large sets of images labeled with corresponding model states. We refer to such sets as “grids” in the state (pose) space without implying any regularity in the state values. Grids can be created, for example, by sampling the state space rendering the corresponding images. While this is a computationally intensive process, it can be performed offline and does not impact online performance.

We propose using grids to efficiently explore state-observation compatibility surfaces in large volumes of state space by supplementing (or completely replacing) optimization in continuous space by discrete search on a grid. The straightforward approach to using a grid is to linearly search for the pose that optimizes the objective function (matching the current observation and conforming to temporal constraints); this has two drawbacks. If the grid is large enough to cover the region of interest in pose space with sufficient density, linear search can still be slow. The search can also be imprecise if the distances between individual grid points are large so that no point lies sufficiently near the peak of the objective function.

We propose two complimentary approaches to overcoming these drawbacks. The first algorithm we propose does not assume strong temporal constraints and searches the whole grid at every time step by using an efficient *approximate* search algorithm [96]. The results

are refined by local search in the state space around grid points, and then applying temporal constraints to, in effect, select the correct mode of the objective function. In addition to rectifying possible errors in the approximate search, the refinement step is convenient for dealing with sparseness of the grid – it would locate the peak of the compatibility function if the initial grid point lies in its general vicinity. This algorithm applied to articulated body pose tracking has been described in [25] coauthored by David Demirdjian, Gregory Shakhnarovich, Kristen Grauman, and Trevor Darrell. Our contribution was posing the solution to pose estimation as a two-stage optimization problem and describing the late temporal integration technique.

The second approach is appropriate when the temporal constraints are relatively strong. In this case they can be used to restrict the number of grid points that need to be considered for linear search at every time-step. If the grid is sufficiently dense, then pose can be estimated directly from results of the search. Otherwise an additional local refinement step can be added. The pose tracking algorithm based on this approach has been presented in [118] and cowritten by Gregory Shakhnarovich, David Demirdjian, and Trevor Darrell. Our contribution was the probabilistic formulation and the transformation of the pose estimation problem to sparse HMM-like inference task.

The grid used for both algorithms described in this chapter consisted of 300,000 distinct image/pose pairs obtained using computer graphics package POSER [20].

The rest of this chapter is structured as follows: we describe global search algorithm *ELMO* and local search algorithm *CRP* in Sections 6.2 and 6.3. The details of the implementation and results are presented in Section 6.5.

6.2 Tracking by Exploring Likelihood Modes

When the temporal constraints are weak, and cannot be relied upon to effectively restrict the search region in the state space, an appropriate approach would be to search the full state space at every time frame. The exact search may be too complex to conduct in a reasonable amount of time but the approximate search techniques can be used. The results of approximate search need to be validated, and refined to obtain appropriate precision in the state estimates, especially when the grid is sparse. This is achieved by initializing a gradient-descent based optimization from every search result.

We embed global grid search into a probabilistic generative model framework in order to correctly account for uncertainties implicit in the tracking problem. An overview our Exploring Likelihood MOdes (ELMO) algorithm is shown in Figure 6-1. In contrast to the standard filtering techniques [3], but similar to the likelihood sampling approach proposed in Chapter 5, ELMO starts with the current observation and introduces temporal constraints late in the estimation. It represents the likelihood function as a mixture of a small number of Gaussians, each corresponding to one of the strong modes.

The likelihood approximation is carried out in two stages. First, the similarity sensitive representation of the observation image is computed using the embedding function defined in Section 2.8.1, and the approximate nearest neighbors are extracted using Locality Sensitive Hashing (LSH) [38] in a manner similar to [96]. The grid points located by the search algorithm are likely to be close to the peaks in the likelihood function, but this needs to

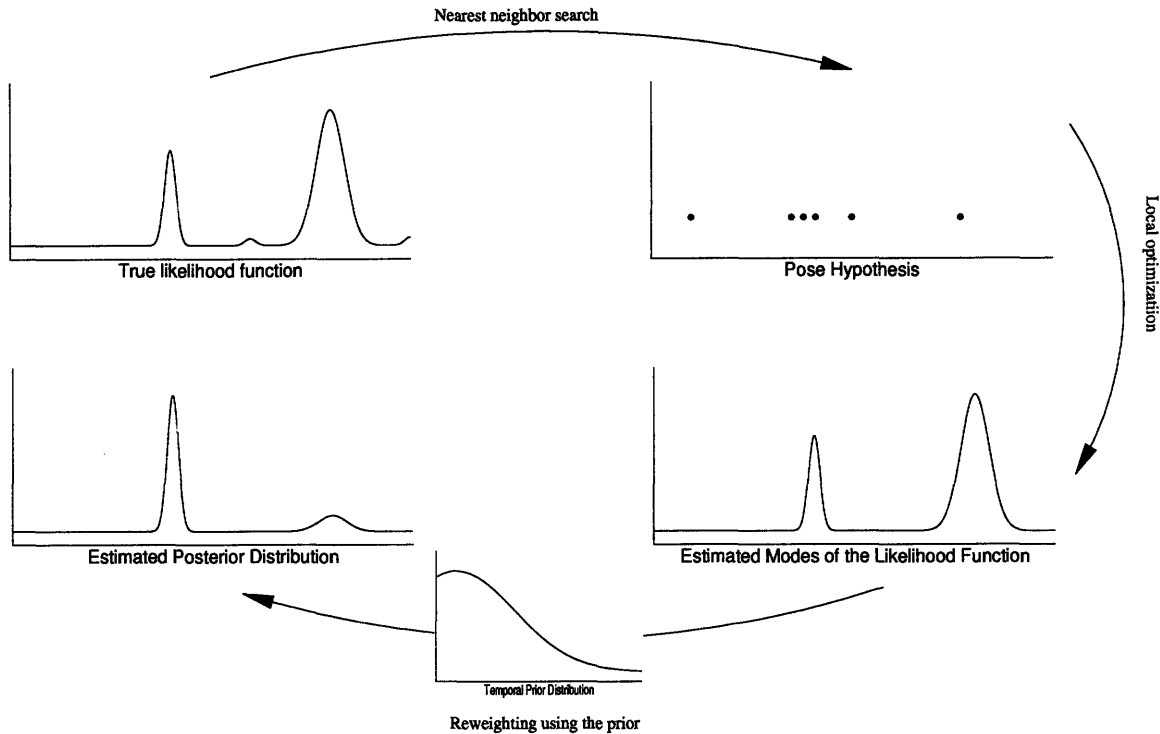


Figure 6-1: High-level overview of the ELMO algorithm. A set of pose hypotheses near the modes of the likelihood function are extracted using approximate nearest neighbor search. The modes are refined with a gradient ascent algorithm initialized at every hypothesis, and a weighted sum of Gaussians estimate is computed for the likelihood function. Note that the number of hypotheses corresponding to a mode does not impact its estimated value. The posterior is then estimated by reweighing members of the mixture according to the temporal prior. This figure originally appeared in [25].

be verified, and the location of the mode and its bandwidth need to be determined. This is achieved by initializing a deterministic local optimization algorithm, described in Section 6.2.1, at every discrete search result.

By reweighing estimated likelihood peaks according to the temporal prior we obtain an estimate of the full posterior model. In contrast to previous view-based tracking methods, our posterior accurately captures the multi-modality of the likelihood function when appropriate. In contrast to previous sample-based methods it is able to search more broadly through the state space, rather than only around the prior. The complete procedure is summarized in Algorithm 4.

There are significant methodological differences between ELMO and classic particle filtering approaches. At no time is a density represented as a (large) set of samples, and so the need for a large number of likelihood evaluations is avoided. Furthermore, repeated instances of the same hypothesis do not imply a greater probability of that hypothesis. We do assume that at least one pose hypothesis will be extracted for each significant peak in the likelihood function. Thus a mode with low likelihood will have low weight even if the gradient ascent algorithm converged to it from multiple starting hypotheses.

Algorithm 4 Tracking Articulated Body by Exploring Likelihood Modes

INITIALIZE the temporal prior to the uniform distribution

for all $t \geq 0$ **do**

 COMPUTE the similarity sensitive embedding of the input image $H_I = H(I^t)$

 COMPUTE the set $\{\theta_i\}$ of approximate neighbors of H_f in the embedding space.

 INITIALIZE local optimization from each θ_i to obtain likelihood mode estimates $\{(\mu_i, \Sigma_i, w_i)\}$

 CLUSTER localized modes

 COMPUTE the posterior estimate by reweighing the likelihood modes according to the temporal prior probabilities of μ_i

 PROPAGATE the posterior through the dynamics function to obtain a temporal prior estimate for the next frame

end for

6.2.1 Local Optimization

We define the likelihood function $p(I|\theta)$ that is optimized at the second step of the algorithm as the compatibility between the observed image I and the articulated body model with parameters θ . In this approach the compatibility depends on the distance between 3D points reconstructed from the observed stereo-pair and those computed from the volumetric model. In the case of monocular data, an adequate likelihood model could be defined (as in [99]) by the re-projection error of the 3D articulated model onto the images.

Formally, we define the set of 3D scene points as $\mathcal{M}(I)$ and body points as $\mathcal{B}(\theta)$. A likelihood model naturally follows as:

$$p(I|\theta) \propto \exp\{-\lambda d^2(\mathcal{M}(I), \mathcal{B}(\theta))\} \quad (6.1)$$

where λ a parameter depending on the uncertainty of the 3D reconstruction.

The local optimum μ_i can be found using standard optimization techniques such as gradient ascent or Levenberg-Marquardt. However, in the particular case of likelihood functions based on a 3D metric error such as $d^2(\mathcal{M}(y), \mathcal{B}(x))$, approximative techniques such as those based on the Iterative Closest Point (ICP) algorithm can be used in order to estimate the optimum μ_i and covariance C_i (see [21, 24]). Such algorithms are proven to converge (when initialized close to the solution) and are less computationally intensive than standard optimization techniques.

Local search is initialized from the set of initial hypothesis $\{\theta_i\}$ returned by the approximate search and from the locations of the modes of the temporal prior. In many cases, the local optima μ_i converge to the same peaks of the likelihood $p(I|\theta)$. Only the highest optima (μ_i^t, C_i^t) are kept to represent the full likelihood model $p(I|\theta)$. In practice, an average of 5 modes is usually kept.

6.2.2 Temporal Integration

As in many other visual tracking tasks, in articulated tracking the temporal prior often provides less information about the posterior distribution than the likelihood function. Given a

sum of Gaussians representation of the likelihood function, we show here how to efficiently integrate information over time and estimate an instantaneous posterior.

A key challenge when propagating mixture models is the combinatorial complexity cost. Indeed, if the posterior distribution at the previous time step (and thus the temporal prior, as we assume simple diffusion dynamics) is estimated as a mixture of K Gaussians, and the likelihood is a sum of L Gaussians, then it is reasonable to expect that the posterior estimate at the current time step will be a mixture of $L \times K$ Gaussians. When the temporal prior is wide (i.e. the noise covariance is much greater than the covariance of the likelihood modes), then the estimate of the posterior may be obtained simply by modifying the weights of the likelihood Gaussians according to the prior.

Let I^t be the observation at time t , and θ^t be the pose. Let the pose likelihood and temporal prior be

$$p(I^t|\theta^t) = \sum_{i=1}^L \hat{w}_i^t N(\theta^t; \mu_i^t, C_i^t) \quad (6.2)$$

$$p(\theta^t|I^0, I^1, \dots, I^{t-1}) = \sum_{j=1}^K w_j^{t-1} N(\theta^t; \mu_j^{t-1}, C_j^{t-1} + C_\eta) \quad (6.3)$$

$$\text{where } N(x; \mu, C) = \frac{1}{\sqrt{(2\pi)^D |C|}} e^{-(x-\mu)^T C^{-1}(x-\mu)}.$$

The i th mode in the likelihood has mean μ_i^t , covariance C_i^t and value $\frac{\hat{w}_i^t}{\sqrt{(2\pi)^D |C_i^t|}}$. Each component of the temporal prior has arisen from the posterior modes estimated at the previous time step (characterized by means μ_j^{t-1} , covariances C_j^{t-1} and weights w_j^{t-1}) after combination with Gaussian noise with covariance C_η .

In general the posterior distribution $p(\theta^t|I^0, I^1, \dots, I^t) \propto p(I^t|\theta^t) p(\theta^t|I^0, I^1, \dots, I^{t-1})$ would be a mixture of $L \times K$ terms of the form $N(\theta^t; \mu_i^t, C_i^t) N(\theta^t; \mu_j^{t-1}, C_j^{t-1} + C_\eta)$. Each such product can be expressed as:

$$\begin{aligned} N(\theta^t; \mu_i^t, C_i^t) N(\theta^t; \mu_j^{t-1}, C_j^{t-1} + C_\eta) &= k N(\theta^t; \hat{\mu}_i, \hat{C}_i), \text{ where} \\ k &= N(\mu_i^t; \mu_j^{t-1}, C_i^t + C_j^{t-1} + C_\eta) \\ \hat{C}_i &= \left((C_i^t)^{-1} + (C_j^{t-1} + C_\eta)^{-1} \right)^{-1} \\ \hat{\mu}_i &= \hat{C}_i \left((C_i^t)^{-1} \mu_i^t + (C_j^{t-1} + C_\eta)^{-1} \mu_j^{t-1} \right) \end{aligned}$$

Since we assume that the noise covariance is much greater than covariance of the likelihood modes, the following is true:

$$\begin{aligned} C_i^t + C_\eta &\approx C_\eta \\ (C_i^t)^{-1} + (C_\eta)^{-1} &\approx (C_i^t)^{-1} \end{aligned}$$

The product can be approximated as

$$N(\theta^t; \mu_i^t, C_i^t) N(\theta^t; \mu_j^{t-1}, C_j^{t-1} + C_\eta) \approx N(\mu_i^t; \mu_j^{t-1}, C_\eta) N(\theta^t; \mu_i^t, C_i^t) \quad (6.4)$$

and the posterior distribution is reduced to

$$p(\theta^t | I^0, I^1, \dots, I^t) \approx \frac{1}{\sum_i^L w_i^t} \sum_{i=1}^L w_i^t N(\theta^t; \mu_i^t, C_i^t), \quad (6.5)$$

$$w_i^t = \hat{w}_i^t \sum_{j=1}^K w_j^{t-1} N(\mu_i^t; \mu_j^{t-1}, C_\eta)$$

Intuitively, we can expect that the wide temporal prior does not vary much over the region of support of each Gaussian in the likelihood, and the posterior distribution is then the mixture of the same Gaussians but with their weights modified by the probabilities assigned to their means by the temporal prior.

6.3 Tracking with Strong Temporal Constraints and Local Search

In the previous section we have described a tracking algorithm based on the approximate global search through a labeled image database. This algorithm is appropriate when we believe that the temporal constraints are weak, and cannot be used to guide the search. This algorithm is effective in pose estimation, but it still involves using a significant number of model-based gradient descent steps which greatly increases the computation time.

An alternative explored in this section is to discard the generative model and to rely fully on the search in a discrete image/pose set (grid). When tracking unconstrained motion, the grid is large and searching the full grid at every time step is computationally expensive. If the temporal constraints are strong enough, then search becomes efficient since it has to be performed over a small region of the grid.

As in the previous section we would like to use search paradigm in a probabilistic framework to correctly propagate uncertainty about the pose estimates through time. To make search more efficient, we would like to use distances in the similarity sensitive embedding space described in Section 2.8.1. This distance is a compatibility function, but cannot be directly considered a likelihood, which precludes a generative model formalism. This leads us to describe pose estimation algorithm using local search as inference in an *undirected* Conditional Random Field model (CRF) [60].

Our algorithm, Conditional Random People (CRP), naturally operates on a discrete set of samples, and we will show how we can estimate the posterior probability in a *continuous* state space using grid filtering methods. The idea underlying these methods is that if the state-space can be partitioned into regions that are small then the posterior can be well approximated by a constant function within each region. We have argued that particle-filtering methods are computationally inefficient with inefficiency coming from the necessity to evaluate the model-based pose-observation compatibility for every sample generated from

the temporal prior. The proposed method, on the other hand, uses a compatibility function based on distance in similarity sensitive embedding space, and embeddings of database images can be precomputed, which is impossible for continuous-space Monte-Carlo methods as the possible pose values are not known in advance.

The direct application of grid filtering would result in the need to evaluate the potential function in each region in the partition, which may be impossible to do in real time even with fast implementation. Fortunately this is not necessary, since at a particular time-step, the *prior state probability* is negligible in the vast majority of the regions, allowing us to search only over regions in the database with significant priors.

Our algorithm operates in a standard predict-update framework: at every step we first estimate the temporal prior probability of the state being in each of the regions in the partition. We then evaluate the observation potential only for regions with a non-negligible prior. While one can view the resulting algorithm as a version of particle filtering where particles can assume only a fixed, finite set of values, this is not quite the case. When the cells are fixed, the transition probabilities between cells can be precomputed, and the temporal prior computation reduced to *single sparse matrix/vector multiplication*, in a manner similar to HMMs [86]. This computation avoids a sampling step altogether, producing better distribution estimates and significantly reducing computation time compared to Monte Carlo methods.

6.3.1 Probabilistic Model

As stated above, we would like to use similarity sensitive embedding as a vehicle of measuring similarity between poses and observed images with the compatibility function defined in Equation 2.5. This compatibility function *cannot* be converted to a conditional distribution $p(I|\theta)$ as commonly done in a generative model frameworks. For example, the compatibility function does not integrate to a finite number:

$$\int \phi(I, \theta) = \infty.$$

It can, however, be used as a *potential* function in an undirected model such as a Conditional Random Field described in Section 1.2.2.

Using standard inference methods in an undirected model with the continuous state can still require evaluating the observation potential $\phi(I, \theta)$ at arbitrary points in the pose space. This would be computationally expensive, since each such evaluation would involve evaluating the rendering function $\mathbf{I}(\theta)$. If we could limit the set of possible evaluation points to a finite (albeit large) discrete set $\{\theta_i\}$, the embeddings $H(\mathbf{I}(\theta_i))$ could be precomputed offline and stored in the memory, resulting in the extremely fast evaluation times.

6.3.2 Estimating Continuous Posterior Distribution with Grid Filtering

In the previous section we have proposed a CRF tracking framework where the observation potential is computed as the distance between embeddings of state and observation

described in the previous section. Computing this potential for an arbitrary pose and image is relatively slow since it would involve rendering an image of a person in this pose and then computing the embedding. This is part of the problem with generative-model-based tracking which we wanted to avoid.

Fortunately, if all of the poses where the observation potential is to be evaluated are known in advance, then we can precompute the appropriate embedding off-line, drastically reducing runtime evaluation cost. We would then compute a single embedding for the observed image, which would be amortized when the potential is evaluated at multiple poses.

While fixing the poses in advance seems too restrictive for continuous space inference, grid-based techniques pioneered by [12, 59] show that this can be a profitable approximation. The main idea underlying these methods is that many functions of interest can be approximated by piece-wise constant functions, if the region of support for each constant “piece” is small enough. As mentioned above, we follow the convention and denote such region of support as a “cell”.

In our case, the function we are interested in is the posterior probability of the pose conditioned on all previously seen observations (including the current one). The posterior is proportional to the product of the temporal prior (the pose probability based on the estimate at the previous time-step and the motion model) and the observation potential. We would like to define the cells such that both of the components are almost constant. The observation potential is often sharply peaked, so the cells should be small in the regions of pose space where we expect large appearance variations, but large in other regions. On the other hand the motion models are usually (and our work is no exception) very approximate and compensate for it by inflated dynamic noise. Thus the temporal prior is broad and should also be approximately constant on cells small enough for observation potential constancy. We derive the grid filter based on the assumption that the partition of the pose space into cells with the properties described above is available.

Let the space of all valid poses Θ be split into N disjoint (and not necessarily regular) cells \mathcal{C}_i , $\Theta = \cup_{i=1}^N \mathcal{C}_i$, $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, i \neq j$, such that both likelihood and prior can be approximated as constant within each cell. Furthermore, let us have a sample $\theta_i \in \mathcal{C}_i$ in every cell. The set of sample points $\{\theta_i\}_1^N$ is referred to as “grid” in the grid-filtering framework.

By virtue of our assumptions, the temporal prior can be expressed as

$$p(\theta^t \in \mathcal{C}_i | \theta^{t-1} \in \mathcal{C}_j) = \int_{\mathcal{C}_i} \int_{\mathcal{C}_j} \phi(\theta^t, \theta^{t-1}) d\theta^{t-1} d\theta^t \quad (6.6)$$

$$\approx \phi(\theta_i, \theta_j) |\mathcal{C}_i| |\mathcal{C}_j|,$$

where $|\mathcal{C}_i|$ is the volume of the i th cell, with the approximation valid when the noise covariance in the transition is much wider than the volume of the cell. The transition potentials $\phi(\theta_i, \theta_j)$ can be precomputed based on a parametric motion model or learned from the data. So the (time independent) transition probability from j th to i th cell is

$$T_{ij} = \frac{\phi(\theta_i, \theta_j) |\mathcal{C}_i|}{\sum_{k=1}^N \phi(\theta_k, \theta_j) |\mathcal{C}_k|}. \quad (6.7)$$

The compatibility between observation and the pose belonging to a particular cell can be written as

$$\phi_o^t(C_i) = \int_{C_i} \phi_o^t(\theta) d\theta \approx \phi_o^t(\theta_i) |C_i|. \quad (6.8)$$

Combining Equations 1.8, 6.6, and 6.8, the posterior probability of pose being in the i th cell is

$$\begin{aligned} p(\theta^t \in C_i | I^{1..t}) &\approx \frac{1}{Z} \phi_o^t(\theta_i) |C_i| \sum_{j=1}^N T_{ij} p(\theta^{t-1} \in C_j | I^{1..t-1}) \\ &= \frac{1}{Z} \phi_o^t(\theta_i) \sum_{j=1}^N S_{ij} p(\theta^{t-1} \in C_j | I^{1..t-1}), \end{aligned} \quad (6.9)$$

where $S_{ij} = |C_i|T_{ij}$ is time independent and can be computed offline. If we denote

$$\pi^t = \begin{pmatrix} p(\theta^t \in C_1 | I^{1..t}) \\ p(\theta^t \in C_2 | I^{1..t}) \\ \vdots \\ p(\theta^t \in C_N | I^{1..t}) \end{pmatrix} \quad \text{and} \quad l^t = \begin{pmatrix} \phi_o^t(\theta_1) \\ \phi_o^t(\theta_2) \\ \vdots \\ \phi_o^t(\theta_N) \end{pmatrix},$$

then the posterior can be written in vector form

$$\pi^t = \frac{1}{W} S \pi^{t-1} .* l^t, \quad (6.10)$$

where $.*$ is the element-wise product, and the scaling factor $W = \sum_{i=1}^N (S \pi^{t-1} .* l^t)_i$ is necessary for probabilities to sum to unity.

The final equation has striking resemblance to the standard HMM update equations. It defines our online CONDITIONAL RANDOM PERSON tracking algorithm (CRP). We can also use standard HMM inference methods [86] to define a batch version of CRP: CRP SMOOTHED (CRPS) uses a forward-backward algorithm to find the pose distribution at every time step conditioned on all observed images. In addition, the most likely pose sequence can be found by using Viterbi decoding and we call the resulting method CRP VITERBI (CRPV).

6.4 Implementation

We have implemented ELMO and CRP variants as described in the previous sections. We used a database of 300,000 pose exemplars generated from a large set of motion capture data in order to cover a range of valid poses. The images are synthetic, and were rendered, along with the foreground segmentations masks, in Poser [20] for a fixed viewpoint using motion-capture sequences are available from [31] and include large body rotations, complex motions, and self-occlusions. The transition matrix was computed by locating 1000 nearest neighbors in joint position space for each exemplar, and setting the probability of

transitioning to each neighbor to be Gaussian with $\sigma = 0.25$. The volume of each cell was approximated as that of a ball with radius equal to the median distance to 50 nearest neighbors. Storing the resulting sparse transition matrix required $500Mb$.

We used the multiscale edge direction histogram (EDH) [96] as the basic representation of images. The binary embedding H is obtained by thresholding individual bins in the EDH. It was learned using a training set of 200,000 image pairs with similar underlying poses (we followed an approach outlined in [133] for estimating false negative rate of a paired classifier without explicitly sampling dissimilar pairs). This resulted in 2,575 binary dimensions requiring $300000 \times 32bytes = 96.5Mb$ to store the descriptors of the whole database.

The tracking algorithms are initialized by searching for 50 exemplars in the database closest to the first frame in the sequence in the embedding space.

Due to the sizes of the database and the transition matrix, both algorithms require large amounts of memory, so we performed our tests on a computer with 3.4GHz Pentium 4 processor and 2GB of RAM.

6.5 Results

6.5.1 Experiments with synthetic data

We have quantitatively evaluated the performance of our tracking method on a set of motion sequences. These sequences were obtained in the similar way as the sequence used for training our algorithm but were not included in the training set.

We compared online algorithms, CRP and ELMO, and the batch version CRPS (CRP SMOOTHED), to three state-of-the-art pose estimation algorithms. The first baseline was a stateless k-Nearest Neighbors (kNN) algorithm that at every frame searches the whole database for 50 closest poses based on the embedding distance. The remaining baseline methods were incremental tracking algorithms: deterministic gradient descent method using the Iterative Closest Point (ICP) algorithm [24], and CONDENSATION [99]. The ICP algorithm directly maximizes the likelihood function at every frame, whereas CONDENSATION and ELMO evaluated the full posterior distribution. In our experiments, the posterior distribution in CONDENSATION was modeled using 2000 particles. The likelihood function defined in ICP, CONDENSATION and ELMO was based on the Euclidean distance between the articulated model and the 3D (reconstructed) points of the scene obtained from a real-time stereo system. In contrast, both CRP(S) and kNN algorithms require only single view intensity images and foreground segmentation.

We have chosen to use the mean distance between estimated and true joint positions as an error metric [5]. In Figure 6-2 we show the performance of 6 algorithms described above on four synthetic sequences. The distributions of pose estimation errors over all 20 test sequences are shown in Figure 6-3. As can be seen, both CRP and CRPS consistently outperform kNN, and CONDENSATION¹, and compare favorably to ICP. ELMO outperforms the rest of the algorithms, but it has to be noted that CRP(S) do not make use of

¹Increasing the number of particles used for CONDENSATION should improve performance, but the computational costs would become prohibitive.

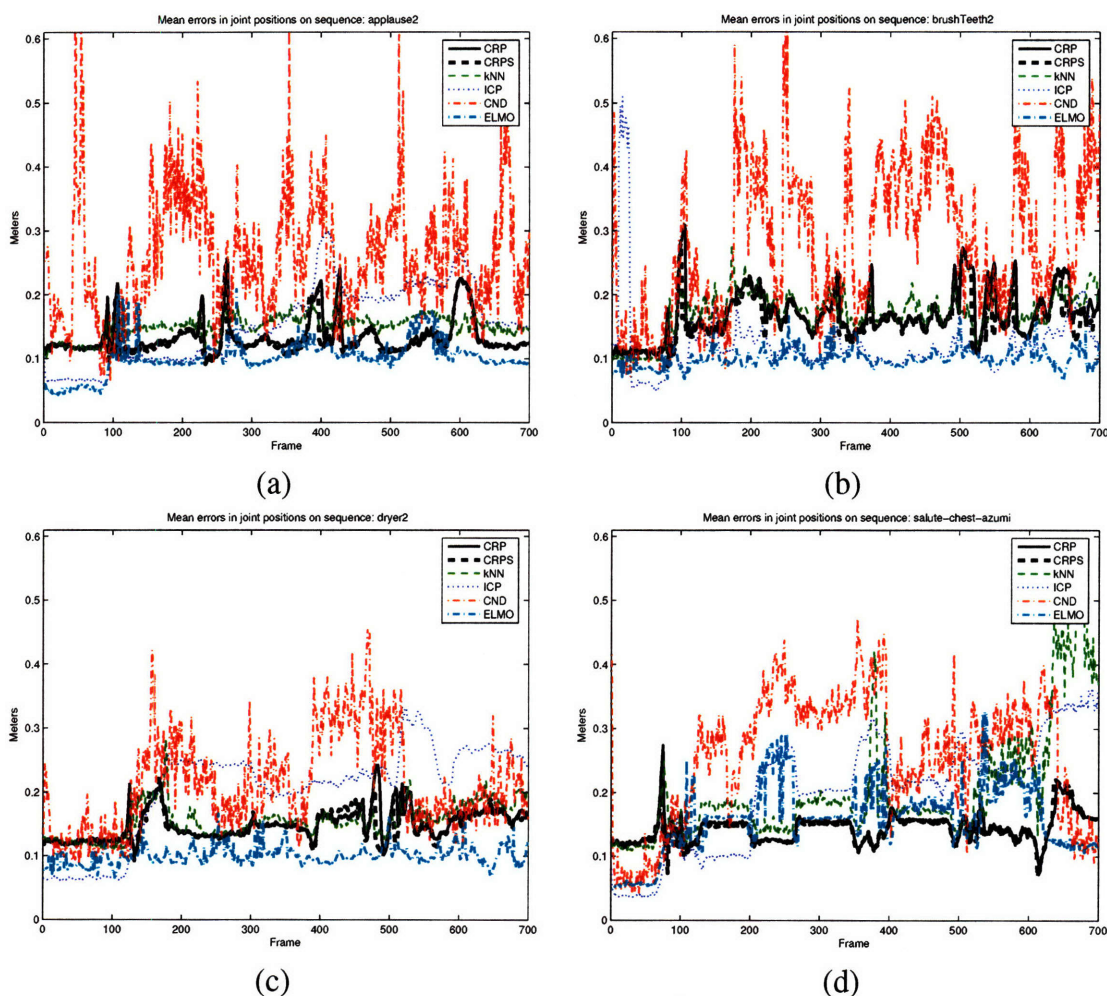


Figure 6-2: Comparing algorithm performance on four synthetic sequences: “applause” (a), “brush teeth”(b), “dryer” (c), and “salute” (d). The error is measured as an average distance between true and estimated joint positions. The graphs are best viewed in color.

stereo data and are orders of magnitude faster. The timing information for all compared algorithms is presented in Table 6.1.

The most likely poses estimated by kNN and CRP algorithms for every hundredth frame of synthetic *salute-chest-azimu* sequence are shown in Figure 6-5. It illustrates the reason for deteriorating performance on kNN on this sequence (Figure 6-2(d)) – there are two major modes of the observation potential corresponding to frontal and rear view of the person. Due to approximations inherent in using SSE to approximate the true observation likelihood, kNN tends to select the frontal view mode, while CRP uses temporal information to correctly select the rear view one.

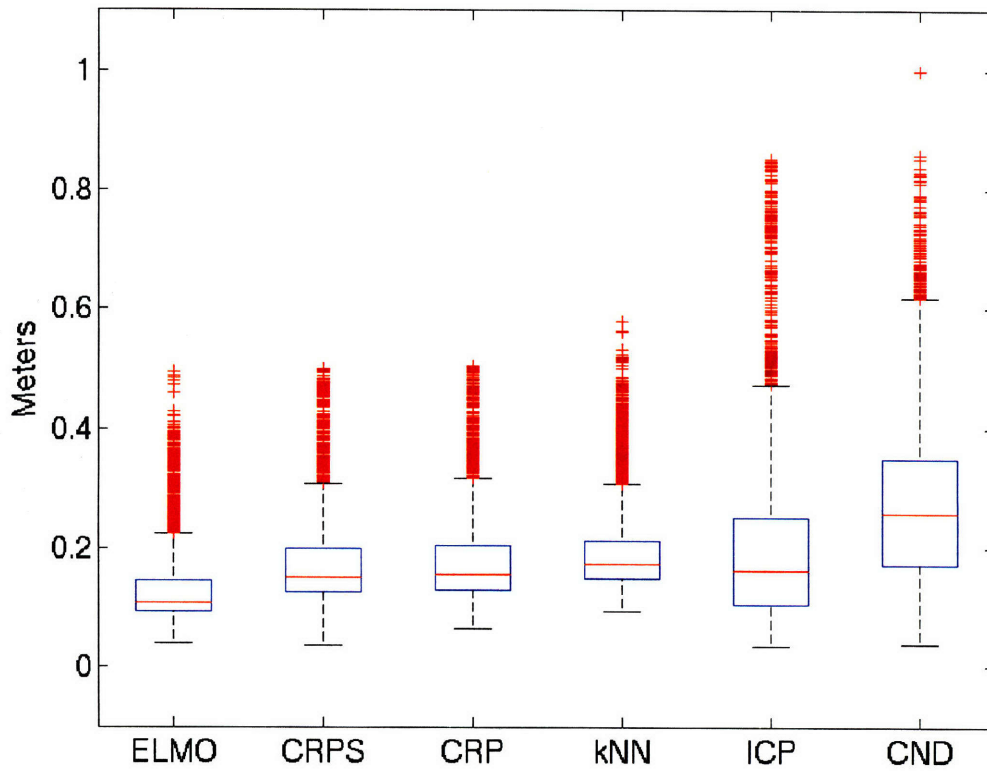


Figure 6-3: Error statistics for competing articulated tracking algorithms computed over 20 test sequences. The mean joints errors for each algorithms are: ELMO – 13cm, CRPS – 17cm, CRP – 18cm, kNN – 19cm, ICP – 20cm, CONDENSATION – 28cm.

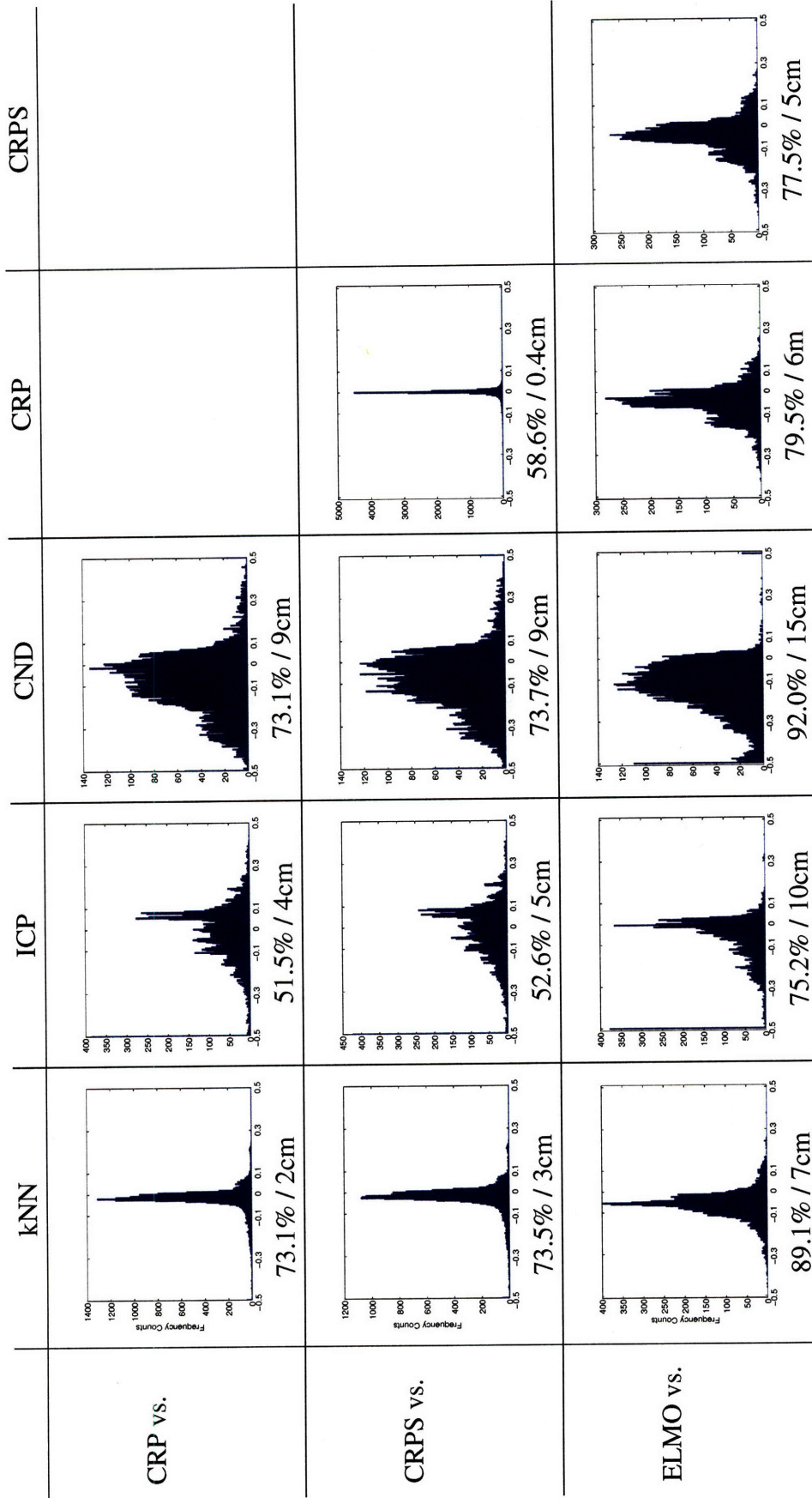


Figure 6-4: Distributions of improvements in joint position estimates of CRP (first row), CRPS (second row) and ELMO (third row) vs. kNN (first column), ICP (second), CONDENSATION (third), CRP (fourth) and CRPS (fifth). Negative values along the x-axis mean lower error for the proposed algorithm. Given for each comparison are the proportion of frames in which algorithm being evaluated was better than the alternative, and the average improvement in error. See text for results on statistical significance.

Algorithm	CRP	CRPS	kNN	ICP	CND	ELMO
Seconds	0.05	0.07	0.5	0.1	120	8

Table 6.1: Average times required for algorithms tested to process a single frame.

6.5.2 Statistical analysis of results ²

In order to evaluate the statistical significance of these results we used the following methodology. We use the mean joint position error as a measure of accuracy of pose prediction on a given frame. Suppose that algorithms A and B are both tested on the total of N frames, producing on the frame i errors e_i^A and e_i^B , respectively. The quantity of interest is the error difference $d_i^{A-B} = e_i^A - e_i^B$. Figure 6-4 shows the distribution of these differences between our algorithms (CRP, CPRS and ELMO) and competing algorithms computed over a large number of synthetic sequences. For example, the top right plot shows the distribution of $d^{CRP-ELMO}$. Negative value of d_i^{A-B} means that on frame i the algorithm A was better than the algorithm B . The lack of a parametric model for the distribution of d^{A-B} makes it difficult to apply thorough statistical testing to hypotheses involving the mean of that distribution. Therefore, the analysis below focuses on the median, which lends itself more easily to non-parametric tests.

One question we can ask is whether the results support the conclusion that A is expected to be better more than half the time. We answer this question using the binomial sign test [97]. Intuitively, it is equivalent to modeling the outcome of each comparison (on one frame) by a coin flip in which “tails” means that the sign of d^{A-B} is negative. The null hypothesis we wish to reject is that the coin is fair. We applied this test to the data histogrammed in Figure 6-4, using p -value³ of $p = 0.001$. At this significance level, CRP was better than k NN and CONDENSATION and worse than ELMO. CRPS was better than k NN, CONDENSATION and ICP and worse than ELMO; we could not establish significant differences in error of CRP vs. ICP.

A more refined statistical evaluation of the difference in performance between two estimation algorithms is based on establishing a confidence interval on the *median improvement*. Given the desired confidence value p we seek a value D such that the probability of the median difference of the errors being above D is less than p .

We apply the following procedure to perform this test. Suppose that D is the q -upper quantile of the observed distribution of d^{A-B} , i.e. qN values are above D . Under the assumption that the true median of the distribution lies below D , we have $P_D = Pr(d^{A-B} \geq D) < 1/2$. Now, we define a random variable Z_D that is the count of observed values of d^{A-B} that exceed D . Its distribution is binomial with parameters P_D and N . Consequently,

$$Pr(Z_D \geq qN) = \text{Bino}(Z_D; P_D, N) < \text{Bino}(Z_D; 1/2, N), \quad (6.11)$$

where $\text{Binomial}(x; p, n) = \binom{n}{x} p^x (1-p)^{n-x}$. Using De Moivre-Laplace approxima-

²The statistical analysis of the tracking results is due to Gregory Shakhnarovich.

³The p -value is the probability of obtaining the observed data under the null hypothesis; that hypothesis is rejected if the p -value falls below a specified threshold, which determines the significance of the test.

Method	k NN	ICP	CND	CRP	CRPS
CRP vs.	1.5cm	-0.04cm	7.13cm		
CRPS vs.	1.75cm	0.28cm	7.47cm	0.11cm	
ELMO vs.	5.74cm	3.67cm	12.5cm	4.58cm	4.27cm

Table 6.2: Confidence intervals for median error reduction, with $p = 0.001$: with probability $1 - p$, the true median of d^{A-B} falls below the value for row A and column B . Positive values indicate cases where we are confident with respect to the improvement (error reduction) achieved by CRP(S) and ELMO over competing methods.

tion [79],

$$\text{Bino}(Z_D; 1/2, N) \approx N(Z_D, N/2, \sqrt{N}/2), \quad (6.12)$$

where $N(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-(x - \mu)^2/2\sigma^2)$. Combining (6.11) and (6.12), and solving for the desired significance p , we get

$$q = \mathbb{G}^{-1}(1 - p; N/2, \sqrt{N}/2)/N,$$

where \mathbb{G} is the inverse of the normal (gaussian) cumulative distribution function. In other words, if we choose the value of D corresponding to such q , the probability of the true median being *lower* than D is at least $1 - p$. Results of this test for $p = 0.001$ are given in Table 6.2: there is a robust advantage to both CRP methods over k NN and Condensation, but not over ELMO.

A relatively large difference between estimated mean (Figure 6-4) and median (Table 6.2) improvements of CRP variants over ICP can be explained by the fact that ICP is more likely to completely loose track (thus producing large errors) than CRP.

6.5.3 Experiments with real data

For the real data, segmentation masks were computed using color background subtraction. Sample frames from a complicated real-world motion sequence are shown in the Figures 6-6, 6-7 and 6-8. The pose estimates are visually similar to the observations, and are quite robust to the errors in the foreground segmentation (very apparent in Figure 6-7) that affect the embedding procedure. While the primary objective of our tracking algorithms is the provide input for higher-level gesture recognition or motion analysis algorithms, the results are of good enough quality to be used for motion reconstruction.

6.5.4 Discussion

In this section we have described the current implementation of CRP and ELMO concepts and demonstrated their advantage over a set of state-of-the-art algorithms. Both algorithms are able to successfully follow a wide range of motion without loosing track.

One limiting factor for use of the methods we have proposed is the way the grid is created. All images are rendered and the embedding is learned for a fixed camera position. In order to apply our algorithms using a differently positioned camera all images need to be

re-rendered and embedding relearned and recomputed – an extremely computationally intensive process. In the future we plan to investigate one method to cope with the viewpoint changes: creating grid and learning the embedding offline for a canonical camera position and using camera calibration information to warp the input images to the canonical view online.

6.6 Summary

In this chapter we have proposed two methods for coping with complexities arising in tracking algorithms in the presence of uncertain dynamics. Both approaches are based on the same paradigm – supplementing the search in the continuous domain by search in a fixed grid of points distributed through the continuous space. These methods have been enabled by availability of similarity sensitive embedding that allowed storing grid in reasonable amounts of memory and efficiently searching it. The algorithms are robust but for quite different reasons. ELMO is not susceptible to error accumulation common in standard trackers since it is being effectively reinitialized at every frame. While CRP operates in a more usual way, it is robust by the virtue of being able to explore very large volumes of the state space at every time step, and thus is not likely to miss the correct peak in the posterior.

The algorithms are complimentary not only in the approach to density estimation but also in the location on the speed/performance landscape. The main advantage of CRP is speed. By limiting the possible pose values to the points on the grid, it accepts the increase in estimation error that is due to discretization but gains the computational advantage. On the other hand, ELMO’s ability to “go off the grid” allows it to produce very good states estimates, but incurs the computational penalty.

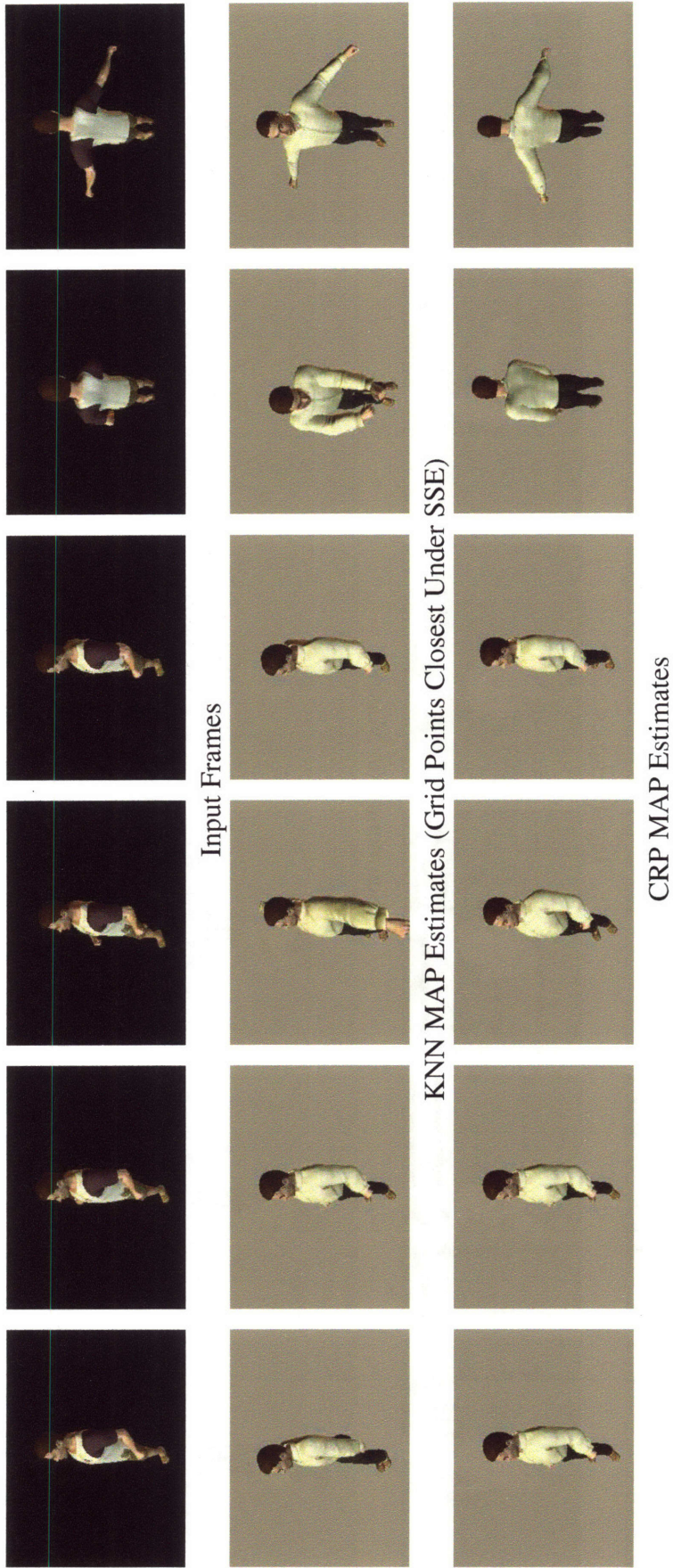


Figure 6-5: Qualitative performance evaluation on a salute-chest-azumi sequence (first row, every hundredth frame is shown), corresponding kNN MAP estimates – grid points closest in the embedding space (second row) and CRP MAP estimates (third row).

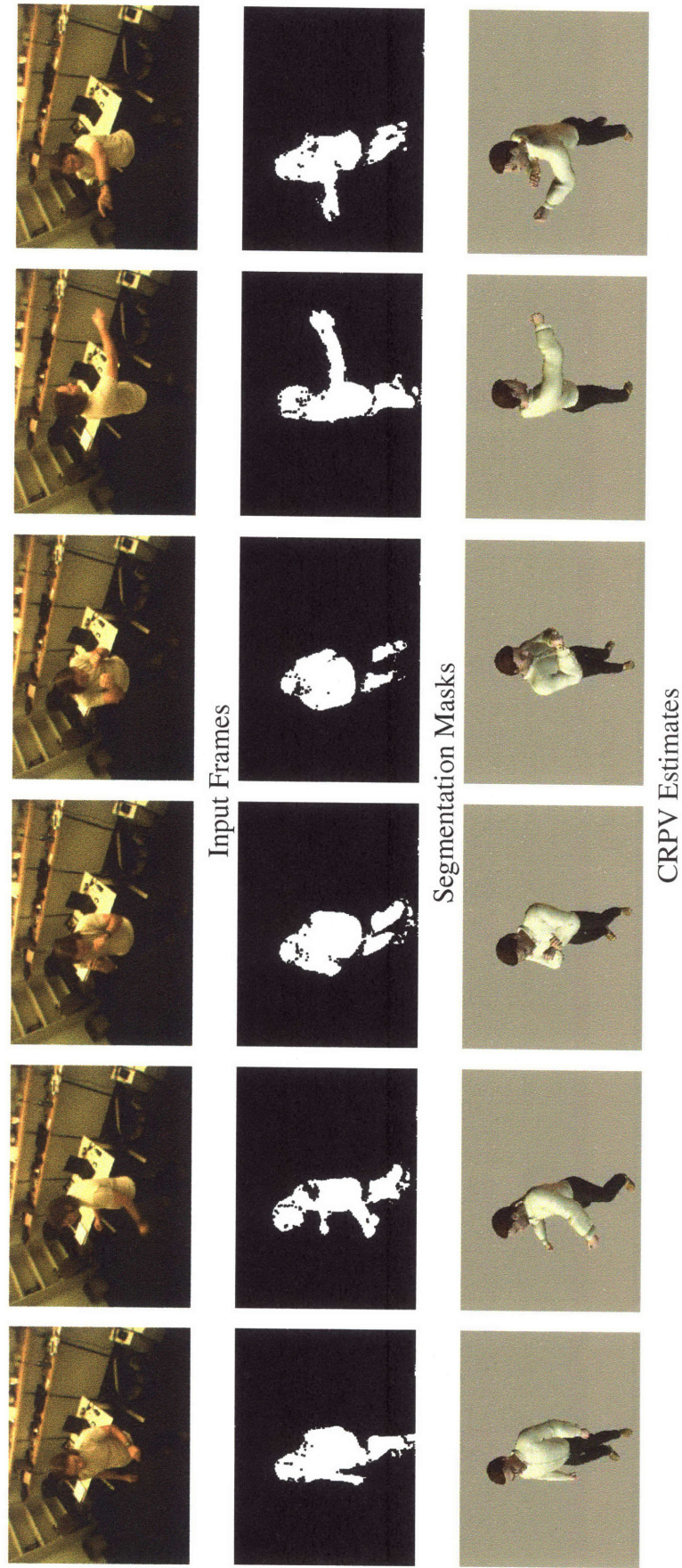


Figure 6-6: Qualitative performance evaluation on a gesture sequence 1. Sample frames from a gesture sequence (first row), segmentation masks (second row) and the corresponding frames from a most likely sequence computed by CRPV algorithm (third row).

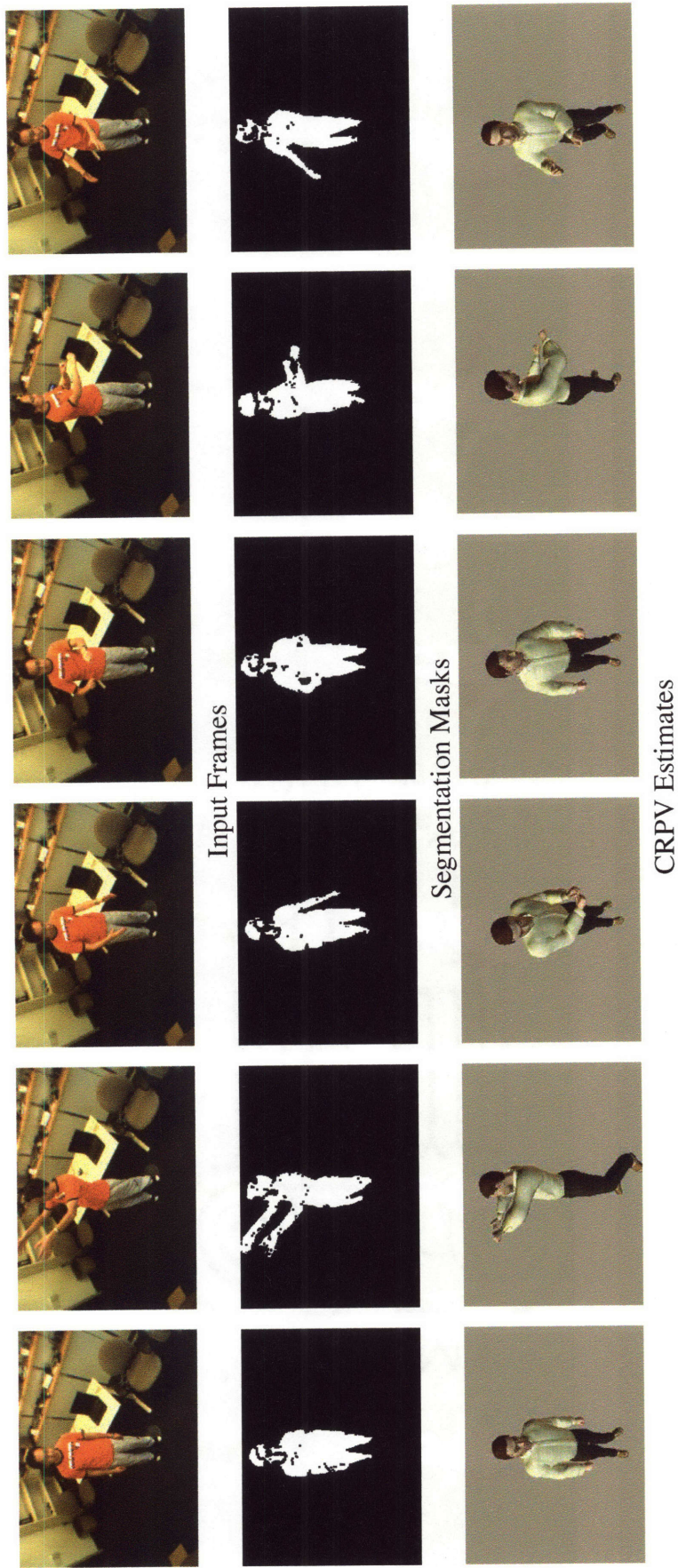


Figure 6-7: Sample frames from a gesture sequence (first row), segmentation masks (second row) and the corresponding frames from a most likely sequence computed by CRPV algorithm (third row)

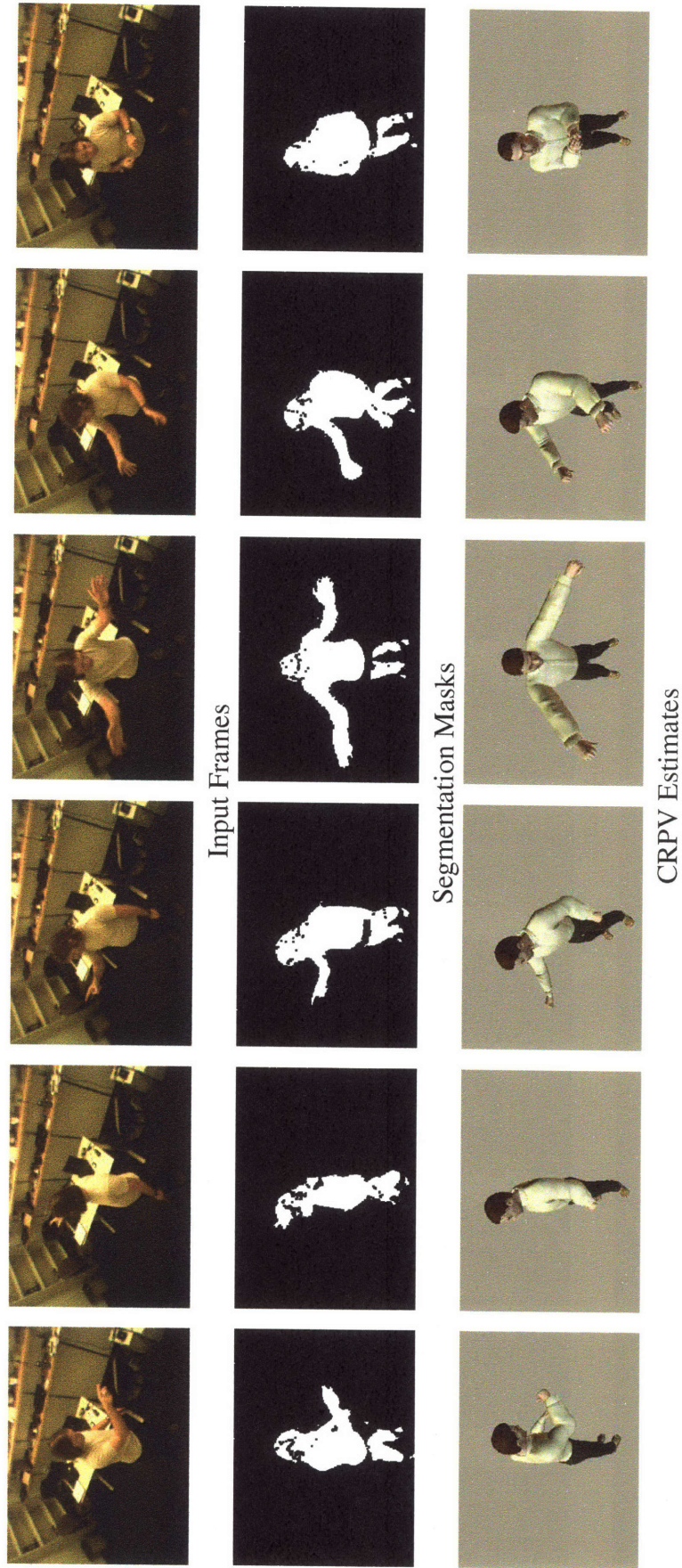


Figure 6-8: Qualitative performance evaluation on a gesture sequence 3. Sample frames from a gesture sequence (first row), segmentation masks (second row) and the corresponding frames from a most likely sequence computed by CRPV algorithm (third row).

Chapter 7

Conclusions

In this chapter we summarize the contributions of this dissertation and sketch out the possible directions of future work.

7.1 Contributions

In this thesis we have considered the problem of dealing with approximate dynamic models that often occur in visual tracking. The approximations involved in specifying the dynamics imply relaxing temporal constraints, which in turn causes two complications: diminishment of the ability to distinguish between true peaks (i.e. those corresponding to the objects of interest) from noise likelihood peaks, and the increase in volume of the state space regions that need to be considered at every time step.

The primary contribution of this thesis was the introduction of two complimentary techniques aimed at improving tracking performance when only in the approximate dynamic models are available: Redundant State Multi-Chain Models and Grid-Based Search.

Redundant State Multi-Chain Model A probabilistic framework that enables combining existing models of dynamics in a principled way to take advantage of differences in their characterization of the system behavior and of different constraints on the motion they represent. In particular the uncertainty in the state estimates is correctly propagated in all directions between states in individual systems in contrast to one-way propagation in common feed-forward frameworks. This approach used the fact that each model defines a temporal prior over the (possibly implicit) instantaneous state and pooling these temporal priors increases the selectivity of the full model. We have also proposed inference algorithms that fully incorporate existing algorithms designed to perform inference in constituent models simplifying system implementation.

The redundant state model allows, in particular, integrating multiple stages of the standard hierarchical vision systems into a single, coherent probabilistic framework. The main advantage of this framework is that implicitly introduces a principled, biologically plausible feedback connection from the high-level modules, aware of the global structure of the scene to the local, low level modules. The efficiency of local methods is thus augmented by the robustness of global methods to local disturbances.

A secondary contribution was developing a **likelihood sampling** technique for articulated body tracking that enables efficient sampling from the upper-body pose distribution based on the location of face and hands, when component detectors or trackers are available. Combining constituent trackers into RSMCM resulted in significant improvement in algorithm’s robustness.

Grid-Based Search An approach to supplementing continuous state space exploration with the fast search in a large discrete set of fixed points (grid) distributed through out the state space. This method allows locating sharp likelihood peaks in large volumes covered by the wide temporal prior. Two algorithms based on this approach were proposed: ELMO, combining approximate global grid search with local gradient-based optimization, and CRP that performs exhaustive search on the grid region constrained by the weak temporal prior.

The most important feature of the grid-based algorithms is their robustness. At every time step, they are capable of thoroughly searching a much larger region of the state-space than any previously proposed approaches and a highly unlikely to loose track. Even when the track is lost due, for example, to occlusion grid-based algorithms will be able to recover. An additional advantage of CRP algorithm is its speed – it has been demonstrated to have realtime performance.

7.2 Future Directions

Redundant State Multi-Chain Models The RSMCM framework, as presented in Chapter 3, assumes that the constituent models have been designed independently. and is primarily concerned with ways to combine them and inference procedures in the joint model. We did not address questions of

- Automatically learning new models from the observed motion and incorporating them into hierarchical RSMCMs. This process could be useful, for example, in detecting dependencies between motions of objects initially modeled as independent (e.g., in a manner described in [115]) and using these dependencies to improve individual object tracking
- Automatically detecting when one or more of constituent models becomes invalid and removing them from the RSMCM.

Grid-Based Methods We have demonstrated grid-based algorithms to be applicable to articulated-body tracking from a fixed viewpoint. We believe that this methodology is applicable to a wide range of tracking tasks but such applications would strongly depend on designing ways to create grids related to particular tracking problems. Introducing viewpoint independence can be achieved by normalizing input images prior to processing. Alternatively, the grids may be designed and similarity embedding learned based not on raw images but rather on higher-level features extracted from sequences. Such features are likely to be extracted using low-level tracking algorithms that could be made more robust by incorporating grid-based methods into RSMCM frameworks.

Appendix A

Proofs of Analysis Theorems

In order to prove Lemma 1 and Theorem 1 we first prove the following lemma

Lemma 2. *If*

$$\begin{aligned}p_y(y) &= \int p_{y|x}(y|x)p_x(x)dx, \\p_{x|y}(x|y) &= \frac{p_{y|x}(y|x)p_x(x)}{p_y(y)}, \\q_y(y) &= \int p_{y|x}(y|x)q_x(x)dx, \quad \text{and} \\q_{x|y}(x|y) &= \frac{p_{y|x}(y|x)q_x(x)}{q_y(y)}\end{aligned}$$

then

$$E_{y \sim p_y(y)} [D_{KL}(p_{x|y}(x|y) || q_{x|y}(x|y))] = D_{KL}(p_x(x) || q_x(x)) - D_{KL}(p_y(y) || q_y(y))$$

Proof. The lemma follows from [18](p 34), or, directly

$$\begin{aligned}E_{y \sim p_y(y)} [D_{KL}(p_{x|y}(x^t|y^t) || q_{x|y}(x^t|y^t))] &= \int p_y(y) \int p_x(x) \frac{p_{y|x}(y|x)}{p_y(y)} \log \left(\frac{p_x(x) \frac{p_{y|x}(y|x)}{p_y(y)}}{q_x(x) \frac{p_{y|x}(y|x)}{q_y(y)}} \right) dx dy \\&= \int_y p_y(y) \int_x p_x(x) \frac{p_{y|x}(y|x)}{p_y(y)} \log \left(\frac{p_x(x) \frac{1}{p_y(y)}}{q_x(x) \frac{1}{q_y(y)}} \right) dx dy \\&= \int_y p_y(y) \int_x p_x(x) \frac{p_{y|x}(y|x)}{p_y(y)} \log \frac{p_x(x)}{q_x(x)} dx dy \\&\quad - \int_y p_y(y) \int_x p_x(x) \frac{p_{y|x}(y|x)}{p_y(y)} \log \frac{p_y(y)}{q_y(y)} dx dy \\&= \int_x p_x(x) \log \frac{p_x(x)}{q_x(x)} - \int_y p(y) \log \frac{p_y(y)}{q_y(y)} \\&= D_{KL}(p_x(x) || q_x(x)) - D_{KL}(p_y(y) || q_y(y))\end{aligned}$$

□

Using Lemma 2, we can re-express $C(q)$ as

$$\begin{aligned}
C(q) &= E_{I^t, \Theta^{t-1}} [D_{KL} (p(\Theta^t | I^t, \Theta^{t-1}) || q(\Theta^t | I^t, \Theta^{t-1}))] \\
&= E_{\Theta^{t-1}} [E_{I^t} [D_{KL} (p(\Theta^t | I^t, \Theta^{t-1}) || q(\Theta^t | I^t, \Theta^{t-1}))]] \\
&= E_{\Theta^{t-1}} [D_{KL} (p(\Theta^t | \Theta^{t-1}) || q(\Theta^t | \Theta^{t-1}))] \\
&\quad - E_{\Theta^{t-1}} [D_{KL} (p(I^t | \Theta^{t-1}) || q(I^t | \Theta^{t-1}))]
\end{aligned} \tag{A.1}$$

Substituting expressions 3.18, 3.19 and 3.21 into equation A.1, using a closed-form expression for KL divergence¹

$$\begin{aligned}
&D_{KL} (N(x; m_1, S_1) || N(x; m_2, S_2)) \\
&= \frac{1}{2} \left(\log \frac{|S_2|}{|S_1|} + Tr (S_1 S_2^{-1} + S_2^{-1} (m_2 - m_1)(m_2 - m_1)^T) - d \right)
\end{aligned}$$

and denoting $\mu_1 = \hat{g}_1(\Theta^{t-1}) - g(\Theta^{t-1})$, we obtain an expression for $C(q_1)$,

$$\begin{aligned}
C(q_1) &= \frac{1}{2} E_{\Theta^{t-1}} \left[\log \frac{|\Sigma_1|}{|\Sigma_0|} + Tr (\Sigma_0 \Sigma_1^{-1} + \Sigma_1^{-1} \mu_1 \mu_1^T) \right] \\
&\quad - \frac{1}{2} E_{\Theta^{t-1}} \left[\log \frac{|\Sigma_1 + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \quad \left. + Tr ((\Sigma_0 + \Sigma_\nu)(\Sigma_1 + \Sigma_\nu)^{-1} + (\Sigma_1 + \Sigma_\nu)^{-1} \mu_1 \mu_1^T) \right] \\
&= \frac{1}{2} \left(\log \frac{|\Sigma_1|}{|\Sigma_0|} + Tr (\Sigma_0 \Sigma_1^{-1} + \Sigma_1^{-1} E_{\Theta^{t-1}} [\mu_1 \mu_1^T]) - \log \frac{|\Sigma_1 + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \left. - Tr ((\Sigma_0 + \Sigma_\nu)(\Sigma_1 + \Sigma_\nu)^{-1} + (\Sigma_1 + \Sigma_\nu)^{-1} E_{\Theta^{t-1}} [\mu_1 \mu_1^T]) \right) \\
&= \frac{1}{2} \left(\log \frac{|\Sigma_1|}{|\Sigma_0|} + Tr (\Sigma_1^{-1} (\Sigma_0 + P_1)) - \log \frac{|\Sigma_1 + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \left. + Tr ((\Sigma_1 + \Sigma_\nu)^{-1} (\Sigma_0 + \Sigma_\nu + P_1)) \right)
\end{aligned} \tag{A.2}$$

The expression for $C(q_2)$ can be obtained in the similar manner.

We can now prove Lemma 1.

Proof. Of Lemma 1.

¹d is the dimensionality of the space

The derivative of $C(q_1)$ with respect to Σ_1 is

$$\begin{aligned}
\frac{d}{d\Sigma_1}C(q_1) &= \frac{d}{d\Sigma_1} \frac{1}{2} \left(\log \frac{|\Sigma_1|}{|\Sigma_0|} + Tr(\Sigma_1^{-1}(\Sigma_0 + P_1)) - \log \frac{|\Sigma_1 + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \left. + Tr((\Sigma_1 + \Sigma_\nu)^{-1}(\Sigma_0 + \Sigma_\nu + P_1)) \right) \\
&= \frac{1}{2} \left(\frac{d}{d\Sigma_1} \log |\Sigma_1| + \frac{d}{d\Sigma_1} Tr(\Sigma_1^{-1}(\Sigma_0 + P_1)) - \frac{d}{d\Sigma_1} \log |\Sigma_1 + \Sigma_\nu| \right. \\
&\quad \left. - \frac{d}{d\Sigma_1} Tr((\Sigma_1 + \Sigma_\nu)^{-1}(\Sigma_0 + \Sigma_\nu + P_1)) \right) \\
&= \frac{1}{2} \left(\Sigma_1^{-1} - \Sigma_1^{-1}(\Sigma_0 + P_1)\Sigma_1^{-1} - (\Sigma_1 + \Sigma_\nu)^{-1} \right. \\
&\quad \left. + (\Sigma_1 + \Sigma_\nu)^{-1}(\Sigma_0 + \Sigma_\nu + P_1)(\Sigma_1 + \Sigma_\nu)^{-1} \right) \\
&= \frac{1}{2} \left(\Sigma_1^{-1}(\Sigma_1 - \Sigma_0 - P_1)\Sigma_1^{-1} \right. \\
&\quad \left. - (\Sigma_1 + \Sigma_\nu)^{-1}(\Sigma_1 - \Sigma_0 - P_1)(\Sigma_1 + \Sigma_\nu)^{-1} \right) \tag{A.3}
\end{aligned}$$

Setting the derivative to 0, we obtain the only minimum at $\Sigma_{1opt} = \Sigma_0 + P_1$. Applying the similar analysis to the second approximation, we obtain $\Sigma_{2opt} = \Sigma_0 + P_2$. \square

The costs of the optimal approximations can be obtained by plugging in the optimal values for dynamic noise covariance into cost expressions:

$$\begin{aligned}
C(q_{1opt}) &= \log \frac{|\Sigma_{1opt}|}{|\Sigma_0|} - \log \frac{|\Sigma_{1opt} + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \\
C(q_{2opt}) &= \log \frac{|\Sigma_{2opt}|}{|\Sigma_0|} - \log \frac{|\Sigma_{2opt} + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \tag{A.4}
\end{aligned}$$

The product of optimal individual priors (for a particular Θ^{t-1}) is a normal distributions with mean μ_* and covariance Σ_* , where

$$\begin{aligned}
\Sigma_* &= (\Sigma_{1opt}^{-1} + \Sigma_{2opt}^{-1})^{-1} \\
\mu_* &= \mu_*(\Theta^{t-1}) = \Sigma_* (\Sigma_{1opt}^{-1}(\hat{g}_1(\Theta^{-1}) - g(\Theta^{-1})) + \Sigma_{2opt}^{-1}(\hat{g}_2(\Theta^{-1}) - g(\Theta^{-1})))
\end{aligned}$$

The cost of the approximation is

$$\begin{aligned}
C(q_*) &= E_{\Theta^{t-1}, I^t} \left[D_{KL}(p(\Theta^t | \Theta^{t-1}) || q_*(\Theta^t | \Theta^{t-1})) - D_{KL}(p(I^t | \Theta^{t-1}) || q_*(I^t | \Theta^{t-1})) \right] \\
&= \frac{1}{2} E_{\Theta^{t-1}} \left[\log \frac{|\Sigma_*|}{|\Sigma_0|} - Tr(\Sigma_*^{-1} \Sigma_0 + \Sigma_*^{-1} \mu_* \mu_*^T) - \log \frac{|\Sigma_* + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \left. + Tr((\Sigma_* + \Sigma_\nu)^{-1} (\Sigma_0 + \Sigma_\nu) + (\Sigma_* + \Sigma_\nu)^{-1} \mu_* \mu_*^T) \right] \\
&= \frac{1}{2} E_{\Theta^{t-1}} \left[\log \frac{|\Sigma_*|}{|\Sigma_0|} - \log \frac{|\Sigma_* + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right. \\
&\quad \left. + Tr((\Sigma_* + \Sigma_\nu)^{-1} (\Sigma_0 + \Sigma_\nu) - \Sigma_*^{-1} \Sigma_0 + ((\Sigma_* + \Sigma_\nu)^{-1} - \Sigma_*^{-1}) \mu_* \mu_*^T) \right] \\
&= \frac{1}{2} \left(\log \frac{|\Sigma_*|}{|\Sigma_0|} - \log \frac{|\Sigma_* + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} + Tr((\Sigma_* + \Sigma_\nu)^{-1} (\Sigma_0 + \Sigma_\nu) - \Sigma_*^{-1} \Sigma_0) \right. \\
&\quad \left. + Tr(((\Sigma_* + \Sigma_\nu)^{-1} - \Sigma_*^{-1}) E_{\Theta^{t-1}}[\mu_* \mu_*^T]) \right) \quad (\text{A.5})
\end{aligned}$$

Expressing

$$E_{\Theta^{t-1}}[\mu_* \mu_*^T] = \Sigma_* (\Sigma_{1opt}^{-1} P_1 \Sigma_{1opt}^{-1} + \Sigma_{1opt}^{-1} P_{12} \Sigma_{2opt}^{-1} + \Sigma_{2opt}^{-1} P_{12}^T \Sigma_{1opt}^{-1} + \Sigma_{2opt}^{-1} P_2 \Sigma_{2opt}^{-1}) \Sigma_*$$

the cost may be rewritten as

$$\begin{aligned}
C(q_*) &= \frac{1}{2} \left(\log \frac{|\Sigma_{1opt}|}{|\Sigma_0|} - \log \frac{|\Sigma_{1opt} + \Sigma_\nu|}{|\Sigma_0 + \Sigma_\nu|} \right) \\
&\quad - \frac{1}{2} \left(\log \frac{|\Delta_1|}{\Sigma_{2opt}} + Tr(\Sigma_{2opt} \Delta_1^{-1} + \Delta_1^{-1} \Gamma_1) - d \right) \\
&= C(q_{1opt}) - \frac{1}{2} \left(\log \frac{|\Delta_1|}{\Sigma_{2opt}} + Tr(\Sigma_{2opt} (\Delta_1)^{-1} + \Delta_1^{-1} \Gamma_1) - d \right) \quad (\text{A.6})
\end{aligned}$$

with

$$\begin{aligned}
\Delta_1 &= (\Sigma_{1opt}^{-1} + \Sigma_\nu^{-1})^{-1} + \Sigma_{2opt} \\
\Gamma_1 &= (\Sigma_{1opt}^{-1} + \Sigma_\nu^{-1})^{-1} \Sigma_{1opt}^{-1} (P_1 - P_{12}) + (-\Sigma_0 - P_{12}^T) \Sigma_{1opt}^{-1} (\Sigma_{1opt}^{-1} + \Sigma_\nu^{-1})^{-1} \\
&= Q_{1\eta} P_1 - (Q_{1\eta} P_{12} + (Q_{1\eta} P_{12})^T + (Q_{1\eta} \Sigma_0)^T), \quad \text{where} \\
Q_{1\eta} &= (I + (\Sigma_0 + P_1) \Sigma_\nu^{-1})^{-1}
\end{aligned}$$

The proof of the Theorem 1 follows from the observation that $C(q_{1opt}) > C(q_*)$ iff $\frac{1}{2} \left(\log \frac{|\Delta_1|}{\Sigma_{2opt}} + Tr(\Sigma_{2opt} (\Delta_1)^{-1} + \Delta_1^{-1} \Gamma_1) - d \right) > 0$. If Γ_1 is positive semidefinite, then Γ_1 can be written as

$$\Gamma_1 = \sum_{i=1}^d (\sqrt{\lambda_i} e_i) (\sqrt{\lambda_i} e_i)^T$$

where (λ_i, e_i) are its eigenvalue/eigenvector pairs, and then

$$\begin{aligned}
C(q_{1opt}) - C(q_*) &= \frac{1}{2} \left(\log \frac{|\Delta_1|}{|\Sigma_{2opt}|} + Tr(\Sigma_{2opt}(\Delta_1)^{-1} + \Delta_1^{-1}\Gamma_1) - d \right) \\
&= \frac{1}{d} \sum_{i=1}^d \frac{1}{2} \left(\log \frac{|\Delta_1|}{|\Sigma_{2opt}|} + Tr(\Sigma_{2opt}(\Delta_1)^{-1} + \Delta_1^{-1})(\sqrt{d\lambda_i}e_i)(\sqrt{d\lambda_i}e_i)^T - d \right) \\
&= \frac{1}{d} \sum_{i=1}^d D_{KL} \left(N(x; 0, \Sigma_{2opt}) || N(x; \sqrt{d\lambda_i}e_i, \Delta_1) \right) > 0
\end{aligned}$$

□

Bibliography

- [1] A. Agarwal and B. Triggs. Learning to track 3d human motion from silhouettes. In *Proceedings of the 21 st International Conference on Machine Learning*, 2004.
- [2] I. MacDonald and W. Zucchini. *Hidden Markov and Other Models for Discrete-Valued Time Series*. Chapman & Hall/CRC, 1997.
- [3] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.
- [4] Ali Azarbayejani and Alex P. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):562–575, 1995.
- [5] A. O. Balan, L. Sigal, and M. J. Black. A quantitative evaluation of video-based 3d person tracking. In *PETS 2005*, 2005.
- [6] S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. In *Intl. Conf. on Pattern Recognition (ICPR '96)*, Vienna, Austria, 1996.
- [7] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding. In *Proc. ICC*, pages 1064–1070, May 1993.
- [8] Xavier Boyen and Daphne Koller. Tractable inference for complex stochastic processes. In *Proc. Uncertainty in Artificial Intelligence*, pages 33–42, 1998.
- [9] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proceedings of IEEE CVPR'97*, 1997.
- [10] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. of CVPR*, 1998.
- [11] A. Brown and G. E. Hinton. Products of hidden markov models. In *Proceedings of Artificial Intelligence and Statistics*, pages 3–11, 2001.
- [12] R. S. Bucy and K. D. Senne. Digital synthesis of nonlinear filters. *Automatica*, pages 287–298, 1971.
- [13] T.-J. Cham and J. M. Rehg. A multiple hypothesis approach to figure tracking. Technical report, Compaq Cambridge Research Laboratory, 1998.

- [14] T.-J. Cham and J. M. Rehg. A multiple hypothesis approach to figure tracking. In *CVPR*, pages 239–245, 1999.
- [15] S.-C. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. In *Video Communications and Image Processing, SPIE Electronic Imaging*, San Jose, Jan 2004.
- [16] K. Choo and D. J. Fleet. People tracking using hybrid monte carlo filtering. In *Proc. ICCV*, 2001.
- [17] J. P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.
- [18] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. J. Wiley & Sons, Inc., New York, 1991.
- [19] I. J. Cox and S. L. Hingorani. An efficient implementation of reid’s multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *PAMI*, 18(2):138–150, Feb 1996.
- [20] Curious Labs, Inc., Santa Cruz, CA. *Poser 5 - Reference Manual*, 2002.
- [21] Q. Delamarre and O. D. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In *Proceedings of International Conference on Computer Vision (ICCV’99)*, pages 716–721, 1999.
- [22] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Em, mcmc, and chain flipping for structure from motion with unknown correspondence. *Machine Learning*, 2003.
- [23] D. Demirdjian. Enforcing constraints for human body tracking. In *Proceedings of Workshop on Multi-Object Tracking, Madison, Wisconsin, USA*, 2003.
- [24] D. Demirdjian, T. Ko, and T. Darrell. Constraining human body tracking. In *IEEE International Conference on Computer Vision*, pages 1071–1078, 2003.
- [25] D. Demirdjian, L. Taycher, G. Shakhnarovich, K. Grauman, and T. Darrell. Avoiding the streetlight effect: Tracking by exploring likelihood modes. In *ICCV*, volume I, pages 357–364, 2005.
- [26] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2000.
- [27] J. Deutscher, A. Davison, and I. Reid. Automatic partitioning of high dimensional search spaces associated with articulated body motion capture. In *Computer Vision and Pattern Recognition*, Dec 2001.
- [28] D. E. DiFranco, T.-J. Cham, and J. M. Rehg. Recovery of 3d articulated motion from 2d correspondences. Technical report, Compaq Cambridge Research Laboratory, 1999.

- [29] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proc. Uncertainty in Artificial Intelligence*, 2000.
- [30] Alexei A. Efros, Alexander C. Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *IEEE International Conference on Computer Vision*, pages 726–733, Nice, France, 2003.
- [31] {Eyes, JAPAN}. Motion capture sequences database. www.mocapdata.com, 2005.
- [32] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.
- [33] O. Faugeras and Q.-T. Luong. *The Geometry of Multiple Images*. MIT Press, 2001.
- [34] H. Fei and I. Reid. Joint bayes filter: A hybrid tracker for non-rigid hand motion recognition. In *ECCV (3)*, 2004.
- [35] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 66–73, 2000.
- [36] T. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journ. Oceanic Engineering*, 8(3):173–183, Jul 1983.
- [37] Z. Ghahramani and M. Jordan. Factorial hidden markov models. *Machine Learning*, 29:245–273, Nov/Dec 1997.
- [38] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *The VLDB Journal*, pages 518–529, 1999.
- [39] A. Gruber and Y. Weiss. Factorization with uncertainty and missing data: exploiting temporal coherence. In *NIPS*, 2003.
- [40] B. Han, D. Comaniciu, and L. Davis. Sequential kernel density approximation through mode propagation: Applications to background modeling. In *ACCV*, 2004.
- [41] M. Harville. A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models. In *Proc ECCV*, 2002.
- [42] M. Harville, G. G. Gordon, and J. Woodfill. Foreground segmentation using adaptive mixture models in color and depth. In *IEEE Workshop on Detection and Recognition of Events in Video*, pages 3–11, 2001.
- [43] G. E. Hinton. Products of Experts. In *Proc. of the Ninth International Conference on Artificial Neural Networks*, pages 1 – 6, 1999.
- [44] David C. Hogg. Model-based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

- [45] B.K.P Horn. *Robot Vision*. The MIT Press, Cambridge, 12th edition, 1998.
- [46] M. Irani and P. Anandan. Factorization with uncertainty. In *ECCV (1)*, pages 539–553, 2000.
- [47] M. Isard. Pampas: real-valued graphical models for computer vision. In *Proc. Computer Vision and Pattern Recognition 2003*, 2003.
- [48] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.
- [49] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *ECCV (1)*, pages 893–908, 1998.
- [50] M. Isard and A. Blake. A smoothing filter for CONDENSATION. *Lecture Notes in Computer Science*, 1406, 1998.
- [51] M. Isard and J.P. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *ICCV01*, pages II: 34–41, 2001.
- [52] T. Jebara and A. Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. Technical report, MIT Media Lab, 1997.
- [53] N. Jovic, M. Turk, and T. S. Huang. Tracking self-occluding articulated objects in dense disparity maps. In *Proc of International Conference on Computer Vision*, 1999.
- [54] M. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.
- [55] S. Julier and J. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Orlando, FL*, 1997.
- [56] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME—Journal of Basic Engineering*, 82(Series D):35–45, 1960.
- [57] Z. Khan, T. Balch, and F. Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *ECCV04*, pages Vol IV: 279–290, 2004.
- [58] O. King and D. A. Forsyth. How does CONDENSATION behave with a finite number of samples? In *ECCV (1)*, pages 695–709, 2000.
- [59] S. C. Kramer and H. W. Sorenson. Recursive bayesian estimation using piece-wise constant approximations. *Automatica*, 24(6):789–801, 1988.
- [60] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

- [61] Mun Wai Lee and Isaac Cohen. Human body tracking with auxiliary measurements. In *AMFG '03: Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, page 112, Washington, DC, USA, 2003. IEEE Computer Society.
- [62] Mun Wai Lee and Isaac Cohen. Proposal maps driven mcmc for estimating human body pose in static images. In *cvpr*, volume 02, pages 334–341, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
- [63] T. S. Lee and D. Mumford. Hierarchical bayesian inference in the visual cortex. *J. Opt. Soc. Am. A*, 20(7), June 2003.
- [64] Ido Leichter, Michael Lindenbaum, and Ehud Rivlin. A general framework for combining visual trackers
u2014the 'black boxes' approach. *Int. J. Comput. Vision*, 67(3):343–363, 2006.
- [65] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In *ECCV (2)*, pages 3–19, 2000.
- [66] D.J.C MacKay. Introduction to monte carlo methods. In Micael I. Jordan, editor, *Learning in Graphical Models*, Adaptive Computation and Machine Learning, pages 175–204. MIT Press, 1998.
- [67] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, May 1999.
- [68] D. Marr. *Vision*. W.H Freeman and Company, New York, 1982.
- [69] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of pearl's "belief propagation" algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, 1998.
- [70] P. F. McLauchlan, I. D. Reid, and D. W. Murray. Recursive affine structure and motion from image sequences. In *ECCV (1)*, pages 217–224, 1994.
- [71] I. Mikic, M. Triverdi, E. Hunter, and P. Cosman. Articulated body posture estimation from multi-camera voxel data. In *Computer Vision and Pattern Recognition*, 2001.
- [72] P. Morasso. Spatial control of arm movements. *Experimental Brain Research*, 42:223–227, 1981.
- [73] G. Mori and J. Malik. Estimating human body configuration using shape context matching. In *European Conference on Computer Vision*, 2002.
- [74] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. *IEEE PAMI*, 19(8):858–867, 1997.
- [75] D. D. Morris and T. Kanade. A unified factorization algorithm for points, line segments and planes with uncertainty models. In *ICCV*, pages 696–702, 1998.

- [76] NASA. *Man-Systems Integration Standards Handbook*, 1995.
- [77] N. M. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.
- [78] Dirk Ormoneit, Christiane Lemieux, and David Fleet. Lattice particle filters. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 395–40, San Francisco, CA, 2001. Morgan Kaufmann.
- [79] A. Papoulis. *Probability, random variables, and stochastic processes*. McGraw Hill, New York, 3rd edition, 1991.
- [80] V. Pavlović, J. M. Rehg, T.-J. Cham, and K. P. Murphy. A dynamic bayesian network approach to figure tracking using learned dynamic models. In *Proc. ICCV*, 1999.
- [81] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1997.
- [82] Vasanth Philomin, Ramani Duraiswami, and Larry S. Davis. Quasi-random sampling for condensation. In *ECCV (2)*, pages 134–149, 2000.
- [83] R. Plänkers and P. Fua. Articulated soft objects for video-based body modeling. In *Proc. of International Conference on Computer Vision*, 2001.
- [84] R. Pless, J. Larson, S. Siebers, and B. Westover. Evaluation of local models of dynamic background. In *Proc CVPR*, 2003.
- [85] Philip Pritchett and Andrew Zisserman. Wide baseline stereo matching. In *ICCV*, pages 754–760, 1998.
- [86] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in speech recognition*, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.
- [87] Rajesh P. N. Rao. Hierarchical bayesian inference in networks of spiking neurons. In *Advances in NIPS 17*, 2005.
- [88] Rajesh P. N. Rao. Neural models of bayesian belief propagation. *Bayesian Brain (to appear)*, 2006.
- [89] H. Rauch, F. Tung, and C. Striebel. Maximum likelihood estimates of linear dynamic systems. *American Institute of Aeronautics and Astronautics Journal*, 3(8):1445–1450, 1965.
- [90] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc. Fifth International Conference on Computer Vision*, pages 612–617, 1995.

- [91] I. Reid and D. Murray. Active tracking of foveated feature clusters using affine structure. *International Journal of Computer Vision*, 18(1):41–60, 1996.
- [92] K. Rohr. Towards models-based recognition of human movements in image sequences. *CVGIP*, 59(1):94–115, Jan 1994.
- [93] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *European Conference on Computer Vision*, Jun 2002. Copenhagen.
- [94] R. Rosales and S. Sclaroff. Inferring body pose without tracking body parts. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2000.
- [95] Sam Roweis and Zoubin Ghahramani. A unifying review of linear Gaussian models. Technical report, University of Toronto, 6 King’s College Road, Toronto M5S 3H5, Canada, 1997.
- [96] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proc. 9th Intl. Conf. on Computer Vision*, 2003.
- [97] D. J. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman & Hall/CRC, 3rd edition edition, 2004.
- [98] H. Sidenbladh. *Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences*. PhD thesis, Royal Institute of Technology, Stockholm, 2001.
- [99] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *Proc. European Conference on Computer Vision*, 2000.
- [100] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *Advances in Neural Information Processing Systems*, 2003.
- [101] L. Sigal, Bhatia S., Roth S., Black M. J., and Isard M. Tracking loose-limbed people. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2004.
- [102] C. Sminchiesescu and B. Triggs. Covariance scaled sampling for monocular 3d human tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.
- [103] C. Sminchiesescu and B. Triggs. Estimating articulated human motion with covariance scaled sampling. *Int. J. Robotics Research*, 22:371–391, June 2003.
- [104] C. Sminchiesescu and B. Triggs. Kinematic jump processes for monocular 3d human tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2003.
- [105] C. Sminchiesescu, A. Kanaujia, Z. Li, and D. Metaxas. Conditional visual tracking in kernel space. In *NIPS*, 2005.

- [106] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Discriminative density propagation for 3d human motion estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2005.
- [107] C. Sminchisescu, M. Welling, and G. Hinton. A mode-hopping MCMC sampler. Technical Report CSRG-478, University of Toronto, Sep 2003.
- [108] M. W. Spratling and M. H. Johnson. A feedback model of perceptual learning and categorisation. *Visual Cognition*, 13(1):129–165, 2006.
- [109] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of CVPR'99*, 1999.
- [110] B. Stenger, P. R. S. Mendonca, and R. Cipolla. Model-based hand tracking using an unscented kalman filter. *Proc. British Machine Vision Conference*, 2001.
- [111] E. Sudderth, A. Ihler, W. Freeman, and A. Willsky. Nonparametric belief propagation. In *CVPR*, Dec 2003.
- [112] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *NIPS*, Dec 2004.
- [113] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Object localization by bayesian correlation. In *ICCV (2)*, pages 1068–1075, 1999.
- [114] L. Taycher and T. Darrell. Bayesian articulated tracking using single frame pose sampling. In *SCTV*, Oct 2003.
- [115] L. Taycher, J. W. III Fisher, and T. Darrell. Recovering articulated model topology from observed motion. *Advances in Neural Information Processing Systems*, Dec 2002.
- [116] L. Taycher, J. W. Fisher III, and T. Darrell. Combining object and feature dynamics in probabilistic tracking. In *CVPR*, volume II, pages 106–113, June 2005.
- [117] L. Taycher, J. W. Fisher III, and T. Darrell. Incorporating object tracking feedback into background maintenance framework. In *Motion*, January 2005.
- [118] L. Taycher, G. Shakhnarovich, D. Demirdjian, and T. Darrell. Conditional random people: Tracking humans with CRFs and grid filters. In *CVPR*, 2006.
- [119] P. Tisainayagam and D. Suter. Visual tracking and motion determination using the imm algorithm. In *ICPR '98: Proceedings of the 14th International Conference on Pattern Recognition-Volume 1*, page 289, Washington, DC, USA, 1998. IEEE Computer Society.
- [120] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991.

- [121] C. Tomasi and T. Kanade. Shape and motion from image streams – a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.
- [122] L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.
- [123] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV (1)*, pages 255–261, 1999.
- [124] Kentaro Toyama and Andrew Blake. Probabilistic tracking in a metric space. In *CVPR*, pages 50–59, 2001.
- [125] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision - to appear*, 2002.
- [126] A.W. Wilson and A.F. Bobick. Learning visual behavior for gesture analysis. In *SCV95*, 1995.
- [127] C. R. Wren. *Understanding Expressive Action*. PhD thesis, Massachusetts Institute of Technology, 2000.
- [128] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.
- [129] Y. Wu, J. Y. Lin, and T. S. Huang. Capturing natural hand articulation. In *Proc. International Conference on Computer Vision*, 2001.
- [130] Y. Wu, Z. Zhang, T. S. Huang, and J. Y. Lin. Multibody grouping via orthogonal subspace decomposition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.
- [131] Ying Wu and Thomas Huang. A co-inference approach to robust visual tracking. In *ICCV*, pages 26–33, 2001.
- [132] Ying Wu and Thomas S. Huang. Robust visual tracking by integrating multiple cues based on co-inference learning. *Int. J. Comput. Vision*, 58(1):55–71, 2004.
- [133] Y. Ke, D. Hoiem, and R. Sukthankar. Computer Vision for Music Identification. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, San Diego, CA, June 2005.
- [134] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *NIPS*, pages 689–695, 2000.
- [135] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free energy approximations and generalized belief propagation algorithms. Technical report, MERL, 2004.

- [136] A. Yuille and N. Grzywacz and. A theoretical framework for visual motion. In T. Watanabe, editor, *High-level Motion Processing*. MIT Press, 1996.
- [137] Q. Zhou and J. Aggarwal. Tracking and classifying moving objects from videos. In *Proc. IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, 2001.