State Estimation of Probabilistic Hybrid Systems with Particle Filters

by

Stanislav Funiak

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2004

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2004

Certified by .....
Brian C. Williams
Associate Professor
Thesis Supervisor

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

State Estimation of Probabilistic Hybrid Systems with Particle Filters

by

Stanislav Funiak

Submitted to the Department of Electrical Engineering and
Computer Science
on May 20, 2004, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Robotic and embedded systems have become increasingly pervasive in every-day applications, ranging from space probes and life support systems to autonomous rovers. In order to act robustly in the physical world, robotic systems must handle the uncertainty and partial observability inherent in most real-world situations. A convenient modeling tool for many applications, including fault diagnosis and visual tracking, are probabilistic hybrid models. In probabilistic hybrid models, the hidden state is represented with discrete and continuous state variables that evolve probabilistically. The hidden state is observed indirectly, through noisy observations. A challenge is that real-world systems are non-linear, consist of a large collection of concurrently operating components, and exhibit autonomous mode transitions, that is, discrete state transitions that depend on the continuous dynamics.

In this thesis, we introduce an efficient algorithm for hybrid state estimation that combines Rao-Blackwellised particle filtering with a Gaussian representation. Conceptually, our algorithm samples trajectories traced by the discrete variables over time and, for each trajectory, estimates the continuous state with a Kalman Filter. A key insight to handling the autonomous transitions is to reuse the continuous estimates in the importance sampling step. We extended the class of autonomous transitions that can be efficiently handled by Gaussian techniques and provide a detailed empirical evaluation of the algorithm on a dynamical system with four continuous state variables. Our results indicate that our algorithm is substantially more efficient than non-RaoBlackwellised approaches. Though not as good as a k-best filter in nominal scenarios, our algorithm outperforms a k-best filter when the correct diagnosis has too low a probability to be included in the leading set of trajectories. Through these accomplishments, the thesis lays ground work for a unifying stochastic search algorithm that shares the benefits of both methods.

Thesis Supervisor: Brian C. Williams
Title: Associate Professor

Acknowledgments

First and foremost, I would like to thank to my research advisor Brian C. Williams for guiding me in my research. He set a great example and provided hard-to-find insights that aided me in my academical career.

I would like to thank to my parents, my sister Monika Jacmenikova, and my friend Max Van Kleek for moral support and to my beloved, Iris F. Flannery, for her patience while I was writing this thesis.

Finally, I would like to thank to Martin Sachenbacher and Lars J. Blackmore for their valuable feedback on my thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Robotic and embedded systems have become increasingly pervasive in a variety of applications. Space missions, such as Mars Science Laboratory (MSL) [3] and the Jupiter Icy Moons Orbiter (JIMO) [1], have increasingly ambitious science goals, such as operating for longer periods of time and with increasing levels of onboard autonomy. Manned missions in space and in polar environments will rely on life support systems, such as the Advanced Life Support System developed at the NASA Johnson Space Center, to provide a renewable supply of oxygen, water, and food. Here on Earth, robotic assistants, such as CMU's Pearl and iRobot's Roomba, directly benefit people in ways ranging from providing health care to routine services and rescue operations.

In order to act robustly in the physical world, robotic systems must handle the uncertainty and partial observability inherent in most real-world situations. Robotic systems often face unpredictable, often harsh physical environments and must continue performing their tasks (perhaps at a reduced rate) even when some of their subsystems fail. For example, in land rover missions, such as MSL, the robot needs to detect when one or more of its wheel motors fail, which could jeopardize the safety of the mission. The rover can detect the failure from a drift in its trajectory and then compensate for the failure either by adjusting the torque to its other wheels or by replanning its path to the desired goal.

One of the major thrusts in reasoning under uncertainty is model-based probabilistic reasoning. Probabilistic model-based methods represent the uncertainty explicitly

by modeling the transition function, observation function, and relations among the variables as probability distributions. Employing these methods allows autonomous systems to reason explicitly about the uncertainty of their belief and to act robustly in their environment. As an example, consider the problem of tracking the position of a robotic arm. Typically, the position of an arm is observed only indirectly, through noisy sensors that measure arm angle and exerted force with a certain amount of error. Since the dynamic equations for the system and the amount of noise on the sensors is typically known, the problem of tracking the arm position can be framed as a *state estimation* problem, in which the given model is combined with the rover's perception in order to obtain an estimate of the arm position. This state estimation problem can then be solved using one of a wide range of methods offered by estimation and control theory. [7]

In this thesis, we investigate the problem of estimating the state of systems with probabilistic hybrid models. Probabilistic hybrid models represent the system with both discrete and continuous state variables that evolve probabilistically according to a known distribution. The discrete state variables typically represent a *behavioral mode* of the system, while the continuous variables represent its *continuous dynamics*. These representations often provide an appropriate level of modeling abstraction when purely discrete, qualitative models are too coarse, while purely continuous, quantitative models are too fine-grained. Probabilistic hybrid models are particularly useful for fault diagnosis, the problem of determining the health state of a system. With hybrid models, fault diagnosis can be framed as a state estimation problem, by representing the nominal and fault modes with discrete variables and the state of system dynamics with continuous variables. Probabilistic hybrid models can thus be viewed as a natural successor to discrete model-based diagnosis systems, such as Livingstone. [61].

State estimation techniques for probabilistic hybrid models has traditionally focused on a restricted subset of conditional linear Gaussian models, in which the discrete state d is a Markov chain with a known transition probability $p(d_t|d_{t-1})$, and the continuous state evolves linearly, with system and observation matrices de-

pendent on d_t. Under such conditions, the continuous estimate for each sequence or discrete state assignments can be computed with a Kalman Filter. The number of tracked estimates can be kept down at an acceptable level using one of a variety of methods, including the well-known interactive multiple model (IMM) algorithm [10], Rao-Blackwellised particle filtering [6, 21] and, more recently, efficient k-best filtering [44, 32]. Straightforward modifications, using variations of the Kalman Filter, such an Extended Kalman Filter [7] or an Unscented Kalman Filter [39], yield extensions to systems with non-linear dynamics, such as rover drive subsystems [36].

In many domains, however, such as rocket propulsion systems [42] or life-support systems [32], the simple Markovian transitions $p(d_t|d_{t-1})$ are not sufficiently expressive. In these domains, the transitions of the discrete variables often also need to depend on the continuous state. Such transitions are called *autonomous*[1], and are substantially more challenging to address. Recent work in k-best Gaussian filtering [32, 43] demonstrated efficient k-best filtering algorithms for hybrid models with autonomous transitions. Owing to their efficient representation and focused search, these methods have been succesfully applied to large systems with as many as 450,000 discrete states. The excessive focus may, however, come at a price if the correct diagnosis is not among the leading set of hypotheses.

In this thesis, we propose an efficient Gaussian particle filtering technique, which performs state estimation in hybrid models with autonomous transitions. In the spirit of prior approaches to k-best filtering and Rao-Blackwellised particle filtering, the algorithm samples mode sequences and, condition on each sequence, estimates the continuous state with a particle filter. Such solution is thus be substantially more efficient than traditional particle filters, yet offers the fair sampling benefits of particle filters.

Applying Rao-Blackwellisation schemes to models with autonomous transitions is difficult, since the discrete and continuous space in these models tends to be coupled. The key innovation in our algorithm is that it reuses the continuous state estimates

[1]In the terminology of hybrid Bayesian networks, these correspond to discrete nodes with continuous parents.

in the importance sampling step of the particle filter. In this manner, the algorithm does not need to maintain the full sample representation of the state space. In order to perform state estimation in Concurrent Probabilistic Hybrid Automata [32], a formalism for modeling large systems, we compute the optimal proposal distributions for single-component transitions and combine them as a proposal distribution for the overall model. We demonstrated the approach on a highly-nonlinear two link system and compared its performance to an efficient k-best filtering solution.

In the following section, we give an overview of probabilistic hybrid models, and clearly state the state estimation problem that we are addressing in this thesis. Then, we lay out the basic principles behind Gaussian filtering for hybrid models, and explain the key insights in our algorithm.

1.1 Probabilistic hybrid models

Continuous models have been a long-held standard in natural sciences, ranging from Newtonian mechanics to fluid dynamics. Their biggest advantage is their fidelity: since the models described detailed interactions in a physical system, accurate predictions and conclusions can be drawn based on these models.

Often, however, it is very challenging to obtain a faithful continuous model of a system or, given a complex continuous model, it may be difficult to reason about it. For example, in order to construct a continuous model of a fault in an actuator, one would have to model detailed interactions between currents and magnetic fields inside it. Constructing such a model may be overly difficult and costly for the given purpose. In these case, engineers typically divide the model into a finite set of steady-state behavioral modes, and model each behavioral mode with a separate set of equations. Such a model is called *hybrid*, because it contains both continuous and discrete variables. For example, by partitioning the behaviors of an actuator (motor) in a robotic arm into `nominal` (functional) and `loose-failed`, one can describe the torque in each mode separately. Such models are simpler and much easier to construct.

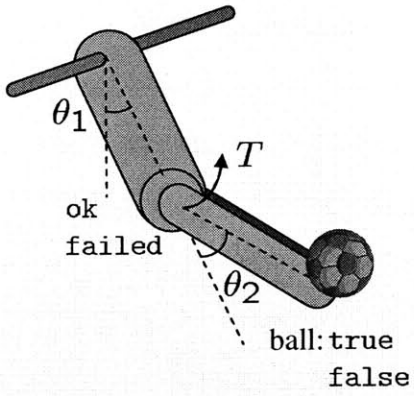

Figure 1-1: Model of an acrobatic robot (left), mimicing a human acrobat (right). *Check the copyright for the picture or replace it with one that's more appropriate.*

Probabilistic hybrid models can be thought of as extensions of discrete models, such as hidden Markov models [52] or dynamic Bayesian networks [14], to continuous dynamical models. As compared to more traditional hybrid systems, such as hybrid I/O automata [47], probabilistic hybrid models have properties crucial for reasoning under uncertainty, including probabilistic transitions between modes, stochastic continuous evolution, and noisy observations.

As an example, consider a simple two-link model of an acrobatic robot with two degrees of freedom, as shown in Figure 1-1. The robot is swinging on a high bar, controlled by an appropriate controller, which specifies the torque T to be exerted by the actuator at the center of the robot. If the actuator is functional (mode = `ok`), it will exert the specified torque. Otherwise, it will exert zero torque. Similarly, the additional weight at the end of the body could be modeled as a discrete mode (`true`, `false`), representing whether or not there is a ball of known weight at the end of the body.

1.2 Problem statement

In this thesis, we focus on estimating the hidden state of systems modeled with Concurrent Probabilistic Hybrid Automata (CPHA). [32] CPHA present several challenges for state estimation, compared to the linear switching models, including nonlinearities, autonomous mode transitions, and concurrency. Given a sequence of control

inputs and noisy observations, our goal is to estimate the discrete and continuous state of the hybrid model. This estimate can take the form of a Maximum a posterior estimate (MAP), or the Minimum Mean Square Error estimate (MMSE) of the discrete and continuous state.

The main application of this problem fault diagnosis. In fault diagnosis, the goal is to estimate the mode (discrete state) of the system from a sequence of noisy observations. Hybrid models are particularly well suited to fault diagnosis, since faults can be modeled as a discrete variable, while the system dynamics is modeled with continuous variables. The goal is then to filter out subtle symptoms from noisy observations.

1.3 Gaussian filtering in hybrid models

In the previous two sections, we have illustrated the class of hybrid models under our consideration and defined the state estimation problem, address in this thesis. In this section, we give an overview Gaussian filtering for switching linear models. In the next section, we describe the technical innovations in our algorithm.

In order to efficiently estimate the state of hybrid systems, Gaussian filtering approaches represent the posterior as a mixture of Gaussians. The key idea behind Gaussian filtering is to track sequences traced by mode variables and, for each sequence, maintain the sufficient statistics of the continuous state conditioned on that sequence. This process is illustrated in Figure 1-2. This figure shows a system, which is known to start in mode `Partially Open`. From mode `Partially Open`, the system can transition to two modes: `Partially Open` and `Fully Open`. Each of these modes can, in turn, transition to other modes, resulting in a total of five mode sequences up to time point 2. For each sequence, we compute the estimate of the continuous variables, given that the system's behavioral modes switched as determined by that sequence. Thus, as shown in Figure 1-2, we would compute an estimate of flow based on the assumption that the system was in mode `Partially Open` at time $t = 0$, in mode `Partially Open` at time $t = 1$, and in mode `Fully Open` at time $t = 2$.

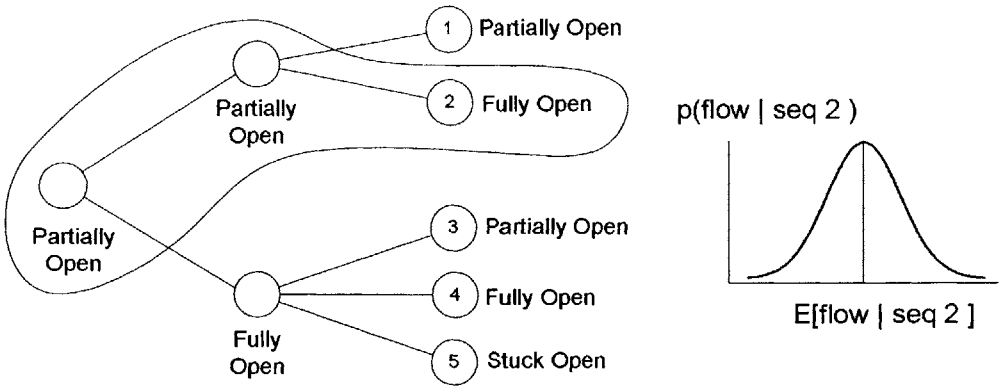

Figure 1-2: The hypothesis tree and the associated estimates.

Due to the nature of switching linear models, it is possible to efficiently compute the continuous estimate using a Kalman Filter [7], once the behavioral mode is fixed at each time step.

The compact representation of Gaussian methods provides an efficient solution to high-dimensional problems. In the case of particle filtering, it has been shown that this representation results in a smaller variance of the state estimate than if we sampled the complete state space. [18] Intuitively, to reach a given a precision, the Rao-Blackwellised estimate will require fewer samples than the non-RaoBlackwellised approach, since we sample from a lower-dimensional distribution. Nevertheless, these methods are generally restricted to models with Gaussian white noise, and may suffer from non-linearities in the model.

Naturally, tracking all possible mode sequences is infeasible: as time progresses, the number of such sequences increases exponentially. Two strategies are commonly employed in Gaussian filtering methods to address this issue: pruning and collapsing. These strategies are illustrated in Figure 1-3. Collapsing combines sequences with the same mode at their fringe to a single sequence. Typically, two or more sequences would be collapsed only if their continuous state estimates are close. Prunnig selects which sequences are less "relevant" given the evidence observed so far, and terminates those sequences. Which sequences are considered "relevant" varies among methods, as discussed below.

K-best filtering method [44, 32] (see Figure 1-4) focuses the state estimation on

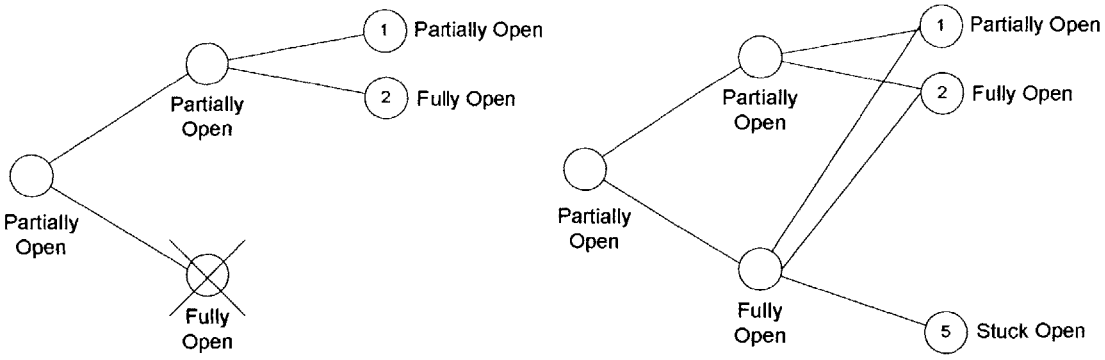

Figure 1-3: Pruning (left) and collapsing strategies (right).

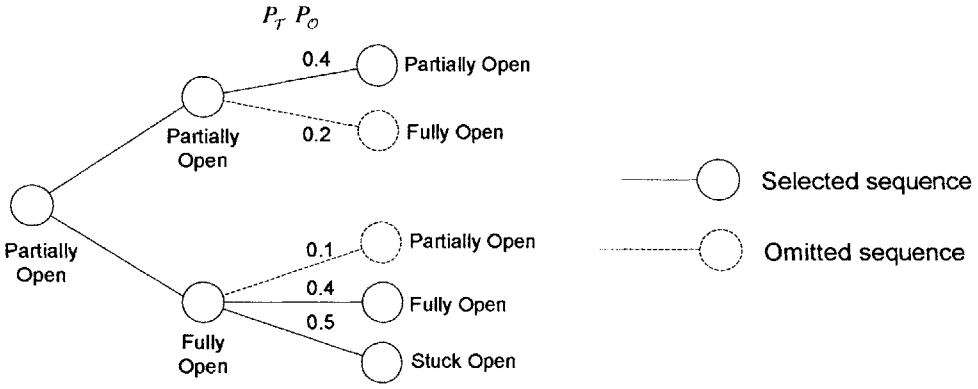

Figure 1-4: The k-best filtering method for pruning mode sequences. The first, fourth, and fifth sequences were selected, while the second and third were omitted.

sequences with high posterior probability. At each time step, the method starts with a set of mode sequences. Based on this set, it computes the probability P_T of transitioning to other modes in the system. The method then enumerates the mode sequences in the decreasing order of prior probability.

Gaussian particle filtering, differs from the above method in that it samples each sequence fairly according to its posterior probability. This process is illustrated in Figure 1-5. At each time step, the algorithm starts with several mode sequences. Then, using the system model, the continuous estimates, and the latest observations, the algorithm computes the transition probability P_T and the observation likelihood P_O for each candidate sequence at the next time step. The transition probability P_T and the observation likelihoods P_O then determine the proposal distribution and weight of the sampled sequences.

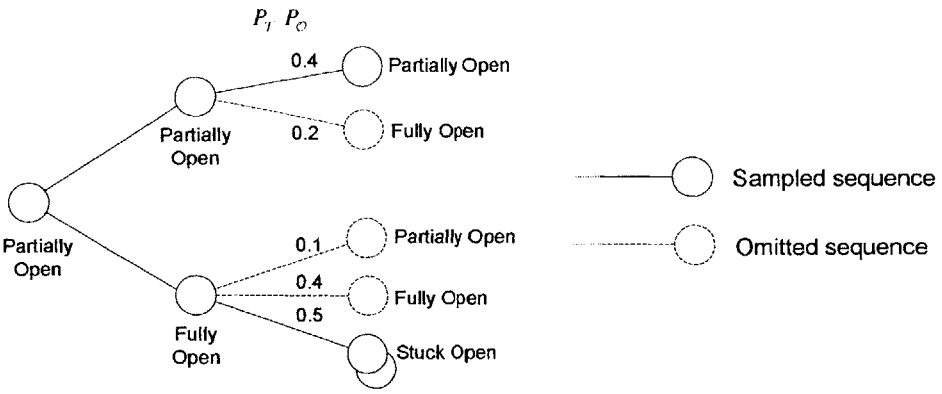

Figure 1-5: The Gaussian particle filtering method for pruning mode sequences. The first and fifth sequence were selected, and the fifth one was sampled twice.

1.4 Technical approach

Our algorithm extends upon the Gaussian filtering method described above. In the spirit of prior approaches to k-best filtering and Rao-Blackwellised particle filtering, our algorithm tracks the sequences of mode assignments and, for each sequence, estimates the state with an Extended Kalman Filter [7] or an Unscented Kalman Filter [39]. Our key insight to handling autonomous mode transitions is to reuse the continuous state estimates from the previous time step, by integrating the Gaussian over the set corresponding to each transition guard, as was done in [32]. This process is illustrated in Figure 1-6. In order to compute the transition probability for the sequence `Partially Open` $\rightarrow$ `Partially Open`, we compute an integral over the Gaussian estimate associated with this sequence. Compared to the prior publication in [32], we provide a rigorous derivation of this procedure, and extend the class of guard conditions that can be efficiently handled by both particle and k-best filtering methods to multivariate linear constraints.

The work presented in this thesis is based on our previously published work [23]. In order to handle multi-component systems, modeled as CPHA, we propose two efficient algorithms that sample the sequences on a component-by-component basis either according to their priors, or according to an approximate posterior, computed for individual component transitions. We demonstrate the algorithm on a 6-variable dynamical system and compare it to the corresponding efficient k-best filtering algo-

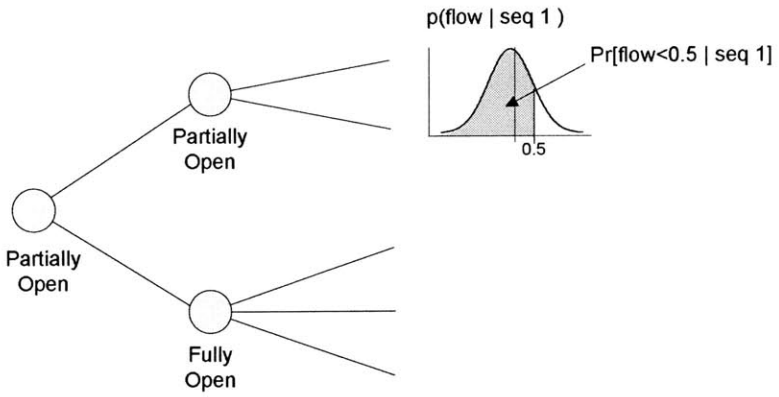

Figure 1-6: Computing the transition probabilities. *Question: how can I improve this figure? Or should I get rid of it?*

rithm [32]. We show that although our algorithm is not as good as a k-best filter when dealing with high-likelihood sequences, it outperforms the k-best filter when the correct diagnosis has too low a prior probability be included in the leading set of sequences. Our results thus lay ground work for a unifying approach, in which k-best filtering is interleaved with Gaussian particle filtering to improve upon the performance of both.

1.5 Thesis roadmap

The rest of this thesis is organized as follows. In Chapter 2, we give an overview of Concurrent Probabilistic Hybrid Automata (CPHA)In Chapter 3, we formally define the hybrid estimation problem addressed in this thesis and give a tutorial on particle filtering, concluding with an elementary Bootstrap particle filter for PHA. In Chapter 4, we describe our Gaussian particle filtering algorithm for PHA and relate it to prior work in particle filtering and hybrid model-based diagnosis. In Chapter 5, we generalize our algorithm to the setting of Concurrent Probabilistic Hybrid Automata. We evaluate this algorithm experimentally in Chapter 6 and compare its performance to the k-best filter [32]. We conclude the thesis with a summary and the discussion of future work in Chapter 7.

Chapter 2

Concurrent Probabilistic Hybrid Automata

Our estimation methods are based on Concurrent Probabilistic Hybrid Automata (CPHA), a formalism for modeling engineered systems with uncertain stochastic dynamics and switching behavioral modes [32, 35]. Examples of such systems include life support systems [32, 43], planetary rovers [15], and rocket propulsion [42]. A CPHA model consists of a network of concurrently operating Probabilistic Hybrid Automata (PHA), connected through shared continuous input/output variables. Each PHA represents one component in the system and has both discrete and continuous hidden state variables. For each assignment to the discrete (mode) variables, PHA specifies the continuous evolution of the component in terms of stochastic difference and algebraic equations. Based on these equations, a global model is constructed using an algebraic equation solver on a mode-by-mode basis, and then used in the inference process.

In this chapter, we first give an overview of Concurrent Probabilistic Hybrid Automata, following the discussion in [35]. We provide cleaner semantics of the discrete state transitions, by viewing the transition guards in CPHA as a set of constraints that partition the probability space.

2.1 Notation

In this chapter and the rest of the thesis, we use the following notation. We denote random variables with lower-case letters, such as x. To denote a vector of random variables, we use lower-case bold letters, such as $\mathbf{x}$. Where clear from the context, we use the same notation for the *set* of random variables; thus, $\mathbf{x}$ would represent both a vector and an equivalent set of random variables.

In order to distinguish between discrete (mode) variables and continuous variables, we use the lower-case subscript d or c, as in $\mathbf{x}_d$. We also use a subscript to refer to the value or instantiation of a variable at a particular time step. Thus, for example, $\mathbf{x}_{d,t}$ would refer to a vector of discrete variables at time step t.

Where it is clear from the context, we use the same notation to denote both the random variable (or a vector of random variables) and its instantiation. Thus, for example, $\mathbf{x}_{d,t}$ may refer to both the vector of discrete random variables at time step t and their instantiation at time step t. The only exceptions to this rule are individual mode sequences (hypotheses), which we refer to by upper index in parentheses, such as $^{(i)}$. Thus, for example, $\mathbf{x}_{d,t}^{(i)}$ would refer to the value of the (discrete) variable $\mathbf{x}_d$ at time step t, as specified by the sequence i.

2.2 Probabilistic hybrid automata

In Probabilistic Hybrid Automata (PHA), a system is modeled by a hybrid automaton that has both discrete and continuous state variables. This framework can be viewed as an extension of a hidden Markov model [52] that incorporates discrete and continuous inputs and stochastic continuous dynamics and autonomous mode transitions.

Figure 2-1 shows a PHA for a two-link acrobatic robot (see Figure 1-1). For this example, we focus our discussion on a model with one discrete (mode) variable, `has-ball`, which represents whether or not the robot carries a ball on its legs.[1] The continuous state of the robot is modeled with four variables, θ_1, θ_2, ω_1, and ω_2.

[1]This event is modeled by increasing the point mass m_2 by a known constant.

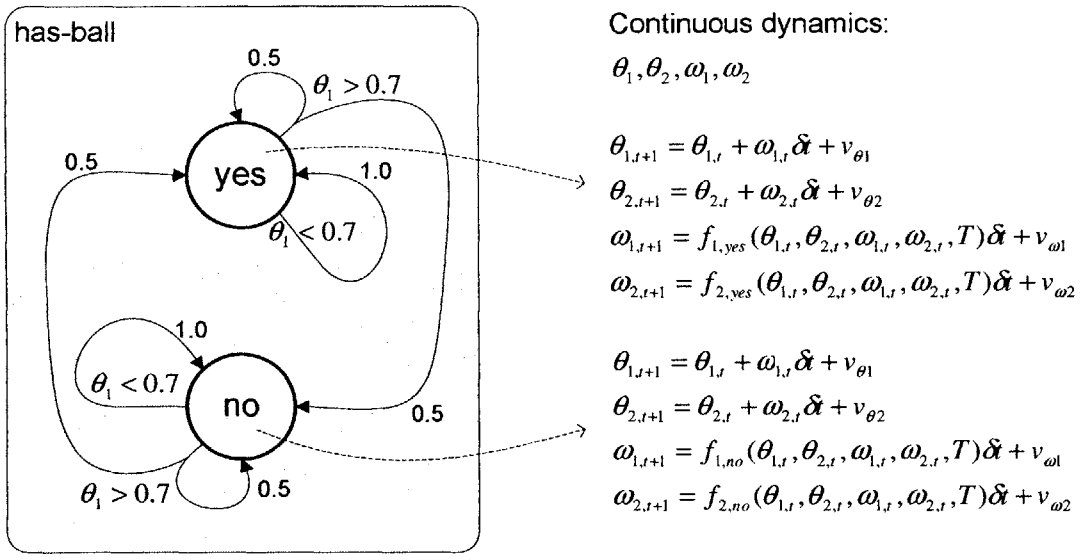

Figure 2-1: A probabilistic hybrid automaton for the acrobot example. Left: transition model for the discrete state of the system. Right: evolution of the automaton's continuous state, one set of equations for each mode.

As with the hidden Markov model, the transitions between modes **yes** and **no** are probabilistic: if, for example, the robot carries a ball at one instant (mode **yes**), the probability of it carrying a ball at the next instant, as specified by our model, is determined by the probability of transitions from that mode. Unlike with the hidden Markov model, however, the transitions can be conditioned on the continuous state of the system. Thus, if the robot carries a ball and $\theta_1 > 0.7$, the probability of transitioning to mode **no** and staying in mode **yes** are the same, whereas if $\theta_1 < 0.7$, the robot will keep the ball with probability 1. These probabilities reflect our modeling assumption that the robot is about as likely to lose the ball as it is to keep it when it is far to the right ($\theta_1 > 0.7$), but it will otherwise, keep the ball.[2]

Each mode is associated with a set of equations that describe the system's dynamics in that mode. For example, when the robot carries a ball, its dynamics is

[2]Certainly, one can imagine a higher-fidelity model that may be more appropriate in a real-world application. For example, rather than comparing θ_1 to one cut-off value 0.7, it may be desirable to consider several ranges of θ_1, with increasing likelihood of capturing a ball. Similarly, it may be desirable to consider not just the angle θ_1, but the placement of the robot's legs, which is a function of both θ_1 and θ_2. However, in order to simplify the explanation, we focus on this simple model.

described by the following differential equations:

$$\dot{\theta}_{1,t+1} = f_{1,yes}(\theta_{1,t}, \theta_{2,t}, \dot{\theta}_{1,t}, \dot{\theta}_{2,t}, T) \quad (2.1)$$

$$\dot{\theta}_{2,t+1} = f_{2,yes}(\theta_{1,t}, \theta_{2,t}, \dot{\theta}_{1,t}, \dot{\theta}_{2,t}, T) \quad (2.2)$$

In this equation, T represents the input torque, exerted by the robot at its center, and f_1, f_2 are nonlinear functions derived using Lagrangian mechanics, see [51]. The extra weight of the ball, the mass of which is assumed to be known, affects parameters in the functions $f_{1,yes}$ and $f_{2,yes}$.

Using the Euler approximation, the system of differential equations 2.1,2.2 translates to the following set of discrete-time difference equations over the state variables θ_1, θ_2, ω_1, and ω_2:

$$\theta_{1,t+1} = \theta_{1,t} + \omega_{1,t}\delta t + v_{\theta_1} \quad (2.3)$$

$$\theta_{2,t+1} = \theta_{2,t} + \omega_{2,t}\delta t + v_{\theta_2} \quad (2.4)$$

$$\omega_{1,t+1} = \omega_{1,t} + f_{1,yes}(\theta_{1,t}, \theta_{2,t}, \omega_{1,t}, \omega_{2,t}, T) + v_{\omega_1} \quad (2.5)$$

$$\omega_{2,t+1} = \omega_{2,t} + f_{2,yes}(\theta_{1,t}, \theta_{2,t}, \omega_{1,t}, \omega_{2,t}, T) + v_{\omega_2}, \quad (2.6)$$

where $v_{\theta_1}, v_{\theta_2}, v_{\omega_1}, v_{\omega_2}$ are added white Gaussian noise variables that represent our uncertainty in the model.

The next two subsections define the Probabilistic Hybrid Automata and the semantics of their discrete transitions. Then we turn to the composition of PHA, which allows the modeler to describe complex systems component-wise as a Concurrent Probabilistic Hybrid Automaton (CPHA). We will conclude with a comparison of CPHA with hybrid dynamic Bayesian networks.

2.2.1 Definition

Formally, a Probabilistic Hybrid Automaton is defined as a tuple $\langle \mathbf{x}, \mathbf{w}, F, T, X_0, \mathcal{X}_d, \mathcal{U}_d \rangle$: [32, 35]

- $\mathbf{x}$ denotes the *hybrid state* of the automaton, which consists of discrete state variables $\mathbf{x}_d$ and continuous state variables $\mathbf{x}_c$.[3] The discrete variables $\mathbf{x}_d$ with finite domain $\mathcal{X}_d$ represent the *operational mode* of the system, while the continuous variables $\mathbf{x}_c$ with domain $\mathbb{R}^{n_x}$ capture its *continuous evolution.*

- $\mathbf{w}$ denotes the set of *input/output variables*, through which the automaton interacts with its environment. For example, a flow regulator interacts with its surrounding components through input flow, output flow, and pressure differences. $\mathbf{w}$ consists of *command variables* $\mathbf{u}_d$ and the set of continuous *input* variables $\mathbf{u}_c$, continuous *disturbances* $\mathbf{v}_c$, and continuous *output variables* $\mathbf{y}_c$, with domains $\mathcal{U}_d$, $\mathbb{R}^{n_u}$, $\mathbb{R}^{n_v}$, and $\mathbb{R}^{n_y}$, respectively.

- The set-valued function $F : \mathcal{X}_d \rightarrow 2^{\mathcal{F}_{DE}} \times 2^{\mathcal{F}_{AE}}$ specifies the *continuous evolution* of the automaton in terms of first-order discrete-time difference equations $F_{DE} \subset \mathcal{F}_{DE}$ and algebraic equations $F_{AE} \subset \mathcal{F}_{AE}$ over the continuous input/variables $\mathbf{w}_c$ and continuous state $\mathbf{x}_c$ for each mode. The discrete-time difference equations specify the continuous evolution of the continuous state between two time-steps, while the algebraic equations specify the relationship among variables in each time step.

- The set-valued function $T : \mathcal{X}_d \rightarrow 2^{\mathcal{T}}$ specifies the discrete transition distribution of the automaton in terms of a finite set of *transition tuples* $\tau_i := \langle p_{\tau i}, c_i \rangle \in \mathcal{T}$. Each transition tuple specifies a distribution $p_{\tau i}$ over the modes $\mathcal{X}_d$ in the automaton. The transition is guarded by a boolean expression over the continuous state and the input/output variables. The expression defines for which assignments to state and input/ouput variables the associated transition distributions hold.[4]

- X_0 specifies the distribution for the initial state of the automaton. X_0 is expressed as a probability mass function $p(\mathbf{x}_{d,0})$ over the modes of the automaton

[3]When clear from the context, we use lowercase bold symbols, such as $\mathbf{v}$, to denote a *set* of variables $\{v_1, \ldots, v_l\}$, as well as a *vector* $[v_1, \ldots, v_l]^T$ with components v_i.

[4]For simplicity, we omit the probabilistic *reset* of the continuous state introduced in [35]. For an elaboration, see [35].

and a normal distribution $\mathcal{N}(\mu_{\mathbf{d}}, \Lambda_{\mathbf{d}}) = p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0} = \mathbf{d})$ for each mode $\mathbf{d} \in \mathcal{X}_d$.

2.2.2 Semantics of the discrete state evolution

The transition tuples returned by the function T for some mode $\mathbf{d}$ specify the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1} = \mathbf{d}, \mathbf{x}_{c,t-1})$. Each tuple $\langle p_\tau, c \rangle \in T(\mathbf{d})$ defines the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1} = \mathbf{d}, \mathbf{x}_{c,t-1})$ to be p_τ in the regions satisfied by the guard c.

For example, consider the acrobot model in Figure 2-1. When $\mathbf{x}_{d,t-1} = \texttt{no}$, the transition distribution is specified by the tuples $\langle [0.5\ 0.5], \theta_1 > 0.7 \rangle$ and $\langle [0\ 1.0], \theta_1 < 0.7 \rangle$. When $\theta_1 > 0.7$, the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}, \mathbf{x}_{c,t-1})$ is uniform, otherwise it is distributed as [0 1.0].

For the purpose of this thesis, we restrict our attention to the guards of the form $c_d(\mathbf{u}_d) \wedge c_c(\mathbf{x}_c, \mathbf{w}_c)$, where c_d is a constraint over the domain of the discrete input variables, $\mathcal{U}_d$, and u_c is a constraint over the space $\mathbb{R}^{n_x} \times \mathbb{R}^{n_w}$ of the continuous state variables and continuous input/output variables. This form is sufficiently expressive to represent both commanded and autonomous transitions.[5] Depending on its form, the constraint c_c can be handled more or less efficiently (see Section 4.3.3).

In order for the PHA to be well-defined, we need to impose certain restrictions on the guards in each mode. Let $A_i \subseteq \mathcal{U}_d \times \mathbb{R}^{n_x}$ denote the set of values for the guard c_i is satisfied. Then, provided that the sets A_i partition the space $\mathcal{U}_d \times \mathbb{R}^{n_x}$, the transition probability p_τ is uniquely defined for all possible values of the continuous state and inputs.

2.3 Concurrent Probabilistic Hybrid Automata

Composition provides a method for defining a new automaton as a combination of existing automata. This allows the modeler to model the individual components of the system separately and then to define a model for the overall system by combining

[5] In fact, the hybrid model of the BIO-Plex plant growth chamber in [34] only contained individual commanded and autonomous transitions.

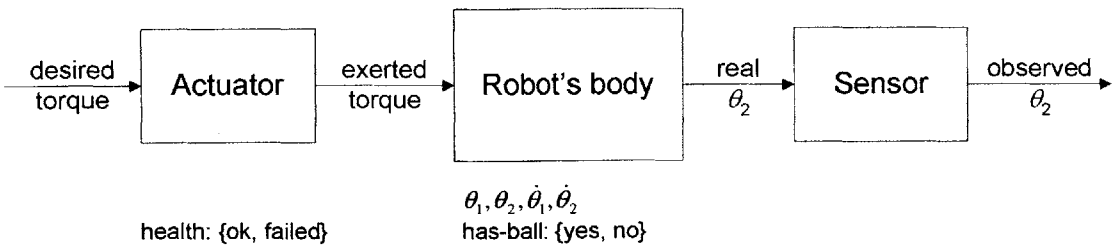

Figure 2-2: Concurrent Probabilistic Hybrid Automata for the acrobatic robot. The component automata are shown in rectangles, with their state variables shown beneath.

models of the system's components.

An example of composition is shown in Figure 2-2. This figure shows a model of the acrobot with three components: one for the actuator at the middle joint, one for the robot's body, and one for a noisy sensor that measures the angle at the center joint, θ_2 (see Figure 1-1). Each component is modeled by a PHA with its own discrete and continuous state variables (if any).

Composed automata are connected through shared continuous input/output variables. In physical systems, this notion corresponds to physically connecting the system's components through natural phenomena, such as force, fluid pressure and flow, electrical potential, and electromagnetic radiation. For example, the actuator and robot's body components of the system interact through the force that is exerted on the robot's body by the actuator. At an abstract level, the sensor and the robot body interact through the true value of θ_2.

Formally, a CPHA $\mathcal{CA}$ is defined as a tuple $\langle \mathcal{A}, \mathbf{u}, \mathbf{y}_c, \mathbf{v}_c, N \rangle$: [32, 35]

- $\mathcal{A} = \langle \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_l \rangle$ denotes the finite set of PHAs that represent the components $\mathcal{A}_i$ of the CPHA. We denote the components of a PHA $\mathcal{A}_k$ by $\mathbf{x}_{dk}$, $\mathbf{x}_{ck}$, $\mathbf{u}_{dk}$, $\mathbf{u}_{ck}$, $\mathbf{y}_{ck}$, F_k, and T_k

- The set of *input variables* $\mathbf{u} = \mathbf{u}_d \cup \mathbf{u}_c$ consists of the sets of discrete input (command) variables $\mathbf{u}_d = \mathbf{u}_{d1} \cup \ldots \cup \mathbf{u}_{dl}$ and continuous input variables $\mathbf{u}_c \subseteq \mathbf{w}_c$.

- The set of *output variables* $\mathbf{y}_c \subseteq \mathbf{y}_{c1} \cup \ldots \cup \mathbf{y}_{cl}$ specifies the subset of observed

continuous I/O variables of $\mathcal{A}$ that are visible to the outside world.

- The set of *noise variables* $\mathbf{v}_c$ specifies the subset of continuous I/O variables that model the disturbances that act upon the system. The disturbances are distributed according to the function $N : \mathcal{X}_d \rightarrow pdf$, which specifies a multivariate p.d.f. for the noise variables $\mathbf{v}_c$.

The discrete transitions for CPHA are defined independently for each component, conditioned on the continuous state. For example, the transition probability

$$p(\texttt{actuator}_t = \texttt{failed}, \texttt{ball}_t = \texttt{no} | \texttt{actuator}_{t-1} = \texttt{ok}, \texttt{ball}_{t-1} = \texttt{no}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) \tag{2.7}$$

is defined as a product of independent transitions

$$\begin{aligned} p(\texttt{actuator}_t = \texttt{failed}, | \texttt{actuator}_{t-1} = \texttt{ok}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) \\ p(\texttt{ball}_t = \texttt{no} | \texttt{ball}_{t-1} = \texttt{no}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) \end{aligned} \tag{2.8}$$

The overall continuous evolution of the CPHA varies in each mode, and is determined by the equations is determined by taking the algebraic and difference equations for each component PHA. If the k-the component is in the mode $\mathbf{d}_k$, the overall model is determined by the union $\cup_k F_k(\mathbf{d}_k)$, where $F_k(\mathbf{d}_k)$ are the algebraic and difference equations for the k-the component. These equations are then solved using xxx and xxx into the standard form

$$\begin{aligned} \mathbf{x}_{c,t} &= \mathbf{f}(\mathbf{x}_{c,t-1}, \mathbf{u}_{c,t-1}, \mathbf{v}_{c,t-1}; \mathbf{x}_{d,t}) \\ \mathbf{y}_{c,t} &= \mathbf{g}(\mathbf{y}_{c,t-1}, \mathbf{u}_{c,t-1}, \mathbf{v}_{c,t-1}; \mathbf{x}_{d,t}). \end{aligned} \tag{2.9}$$

Typically, we further assume that the noise is additive, white Gaussian, that is,

$$\begin{aligned} \mathbf{x}_{c,t} &= \mathbf{f}(\mathbf{x}_{c,t-1}, \mathbf{u}_{c,t-1}, \mathbf{v}_{x,t-1}; \mathbf{x}_{d,t}) \\ \mathbf{y}_{c,t} &= \mathbf{g}(\mathbf{y}_{c,t-1}, \mathbf{u}_{c,t-1}, \mathbf{v}_{y,t-1}; \mathbf{x}_{d,t}). \end{aligned} \tag{2.10}$$

However, using the Unscented Kalman Filter [59], it would be possible to use the more general setting of Equation 2.9.

Chapter 3

Particle Filtering

Given a hybrid model of the system, our goal is to estimate its state from a sequence of control inputs and observations. This estimate can then be used for a number of tasks, ranging from diagnostics to autonomous control. In this chapter, we first discuss and formally define the hybrid estimation problem. Then we give an overview of particle filtering concepts and algorithms, which will be relevant in the next chapter. Finally, we show how a simple particle filtering algorithm, the *Bootstrap filter* [26, 40], specializes to the class of systems modeled as Probabilistic Hybrid Automata. This algorithm is not as efficient as the prior approaches in particle filtering for hybrid models [58, 22, 55, 20], but it demonstrates the key principles of particle filtering in PHA.

3.1 Hybrid estimation problem

Given a CPHA description of a system, our goal is to estimate the state of the system from a sequence of noisy observations and control inputs known to drive the system. Depending on the application, we might be interested in different aspects of this problem:

1. **Mode estimation** typically refers to the task of computing the most likely mode (MAP mode estimate) of the system or the distribution over the set of

possible modes. This task is most applicable in areas, such as fault diagnosis, which concern themselves primarily with detecting the nominal and off-nominal modes of the system and less with its continuous state.

2. **Continuous state estimation** refers to the task of computing a MMSE (minimum mean square error) estimate of the continuous state. This can be used in applications, such as target tracking and improved odometry calculation for land rovers, in which the continuous state is of primary concern.

3. **Hybrid state estimation** refers to the task of computing the *joint* estimate over both the discrete and the continuous state. This task is useful in the area of model-based programming, which allows the programmer to write control programs directly in terms of the hidden state of the system, and needs to compute queries over the joint probability space. In addition, this task is useful for state tracking under failure.

In this thesis, we address the last of these tasks, hybrid state estimation. For brevity, we also use the term *hybrid estimation.* More precisely, we wish to compute the probability distribution over the discrete and continuous state variables at time t, given the control inputs and outputs. Formally:

Definition 1 Hybrid Estimation: *Given a CPHA model of the system $\mathcal{CA}$ and the sequence of control inputs $\mathbf{u}_{0:t}$ and observed outputs $\mathbf{y}_{0:t}$, estimate the hybrid state $\mathbf{x}_{d,t}, \mathbf{x}_{c,t}$ at time t.*

The motivation for this definition is twofold: First, the discrete and continuous variables in PHA are highly intertwined. Not only does the continuous evolution of the state and the observations depend on the discrete state of the system, but also the discrete transitions depend on the continuous state, as we have seen in Section 2.2. This makes separate mode estimation and continuous state estimation infeasible, and the two problems must be addressed jointly. The second motivation for this definition is that one can extract a mode estimate from a hybrid state estimate as a marginal, by ignoring the continuous portion of the estimate.

Since we are dealing with highly nonlinear systems that do not generally have a closed-form solution, we do not require the estimator to be unbiased or to have minimum variance. Chapter 6 provides an empirical evaluation of various estimators.

3.2 Particle filtering

Given the problem of estimating state in a hybrid model, a natural question is, what techniques can be used to address this problem. Particle filters offer an appealing alternative, because they make, unlike linear continuous solutions, such as a Kalman Filter, very weak assumptions on the form of the models. This property enables their immediate use in hybrid models, which have both discrete and continuous state variables and may have non-linear dynamics.

In this section, we give an overview of the particle filtering method. Conceptually, particle filters reason in terms of the discrete-time evolution taken by the state variables $\mathbf{x}$ from the initial time step 0 to the present time step t. We are interested in the joint posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ over the set of possible state evolutions $\mathbf{x}_{0:t} \triangleq \{\mathbf{x}_0, \ldots, \mathbf{x}_t\}$, given the observations $\mathbf{y}_{1:t} \triangleq \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$ and control inputs $\mathbf{u}_{0:t} \triangleq \{\mathbf{u}_0, \ldots, \mathbf{u}_t\}$. For example, for a system with one state variable θ and $t = 1$, this distribution describes the joint probability of the state at time 0, θ_0, and the state at time 1, θ_1, as shown in Figure 3-1.

Given the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, it is possible to express the characteristics of interest, such as the marginal distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ and the minimum mean square error (MMSE) estimate $\mathbb{E}[\mathbf{x}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}]$, by taking appropriate integrals, for example,

$$\mathbb{E}[\mathbf{x}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}] = \int_{\mathbf{x}_{0:t}} \mathbf{x}_t p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) d\mathbf{x}_{0:t}. \tag{3.1}$$

Unfortunately, these integrals, as well as the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, are rarely tractable, except in the case of simple models, such as linear Gaussian models. In particle filters, this difficulty is addressed by approximating the posterior

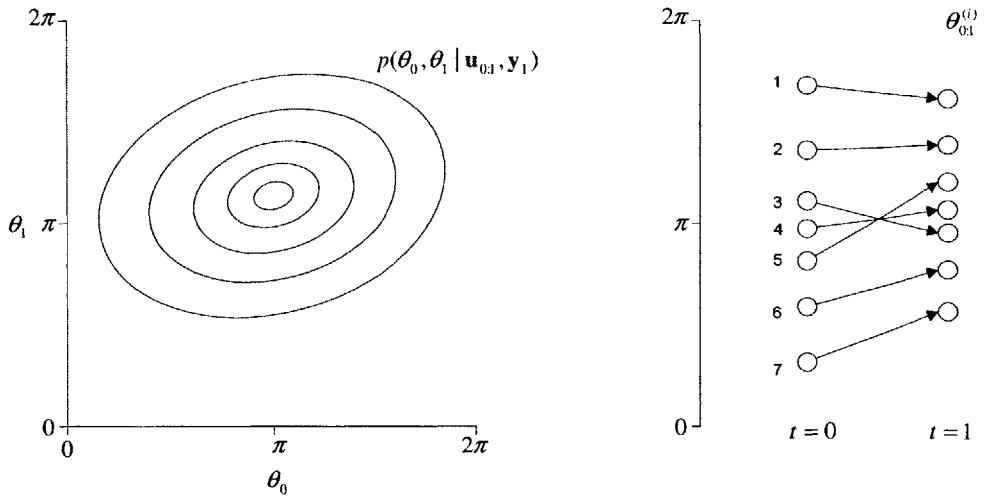

Figure 3-1: Samples from one-dimensional state space. The state at time 0 is positively correlated with the state at time 1, which is reflected in the samples.

distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ with a set of samples, evolved sequentially. The samples are evolved in two steps, importance sampling and selection, and can, in turn, be directly used to estimate the hidden state and other desired characteristics.

3.2.1 Concepts

One of the fundamental principles in particle filtering is a duality between samples and the distribution from which they are taken [9]: A distribution can generate random samples, which are random events from that distribution, and samples can, in some sense, approximate the distribution that generated them.

For example, consider a distribution $p(x)$ shown in Figure 3-2. Given this distribution, we can generate independent, identically distributed (i.i.d.) samples $x^{(i)}$. These samples have the highest occurence in the regions where the probability density function $p(x)$ is highest and tend to be sparse in the regions where $p(x)$ is low. Given the samples $x^{(i)}$, we can approximate the distribution, for example, as a histogram (Figure 3-2 on the right), in which we compute the number of samples over fixed intervals of the probability space. At a more fundamental level, the samples $x^{(i)}$

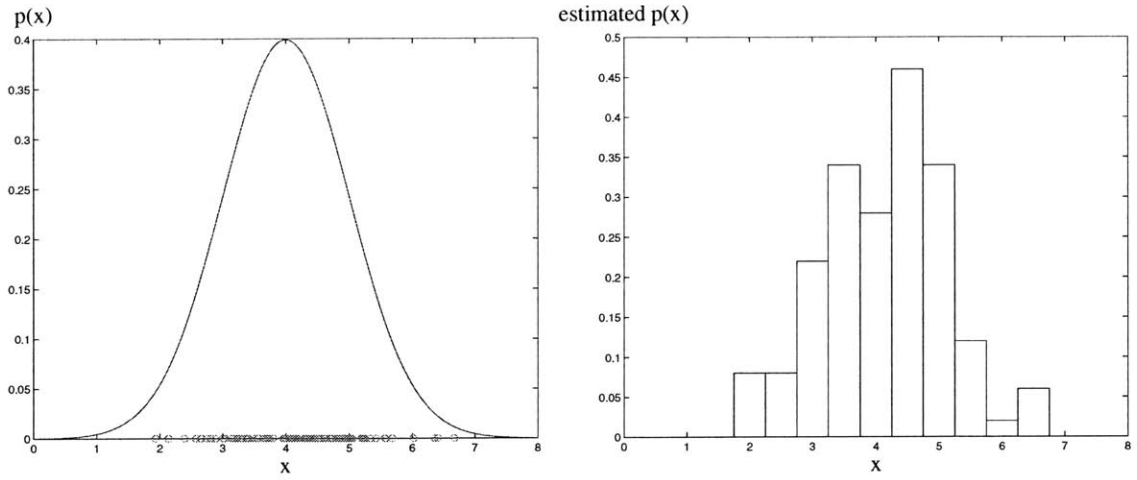

Figure 3-2: Left: A probability distribution and 100 i.i.d. samples taken from it (shown with circles). Right: Histogram of the samples (appropriately scaled).

approximate the distribution $p(x)$ in terms of the probability mass function

$$Pr[x = c] = \frac{1}{N}|i \in \{1, \ldots, N\} : x^{(i)} = c| \tag{3.2}$$

or, equivalently,

$$p_{\bar{N}}(x) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^{(i)}}(x), \tag{3.3}$$

where $\delta_{x^{(i)}}(x)$ is the Dirac delta function positioned at the i-th sample $x^{(i)}$ and $\frac{1}{N}$ is a normalizing factor. This approximation is evident in Figure 3-2 (left): the p.d.f. $p(x)$ can be approximated by the relative density of the random samples $x^{(i)}$.

We can approximate the expected value of any function $f(x)$ with respect to a given distribution $p(x)$ by taking independent samples $x^{(i)}$ from this distribution. Rather than computing the (possibly intractable) integral

$$\mathbb{E}[f(x)] \triangleq \int_x f(x)p(x)dx, \tag{3.4}$$

we only look at the points given by the samples $x^{(i)}$ and approximate the integral

with a finite summation:

$$\mathbb{E}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}) \triangleq \bar{I}_N(f). \tag{3.5}$$

For example, consider the case when we wish to estimate $\mathbb{E}_{p(x)}[x]$, the mean of the random variable x w.r.t. distribution $p(x)$. In this case, $f(x) = x$, and $\bar{I}_N(f) = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}$. This estimator is consistent with our intuition that the mean of i.i.d. samples should be a "good" approximation to the mean of the distribution that generated them.

Indeed, the estimator $\bar{I}_N(f)$ has several desirable properties. First, $\bar{I}_N(f)$ is unbiased:[1]

$$\mathbb{E}[\bar{I}_N(f)] = \mathbb{E}[\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[f(x^{(i)})] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[f(x)] = \mathbb{E}[f(x)]. \tag{3.6}$$

The third equality in Equation 3.6 holds, because the samples $x^{(i)}$ are drawn from the distribution $p(x)$ and thus, need to be themselves treated as random variables with distribution $p(x)$, hence $\mathbb{E}[f(x^{(i)})] = \mathbb{E}[f(x)]$. In other words, in repeated trials for a fixed N, the estimates will be centered around the estimated value $\mathbb{E}[f(x)]$. Furthermore, since the samples $x^{(i)}$ are i.i.d. random variables, the random variables $f(x^{(i)})$ are also i.i.d. From the strong law of large numbers, $\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})$ converges almost surely (a.s.) to $\mathbb{E}[f(x^{(i)})] = \mathbb{E}[f(x)]$ as $N \to +\infty$. In other words, the estimates $\bar{I}_N$, $N = 1, 2, \ldots$, converge almost surely to the estimated quantity as the number of samples increases (see Figure 3-3).[2]

Now, suppose that we are not able to easily take samples from the given distribution $p(x)$, but we can easily evaluate the p.d.f. $p(x)$ for any given x up to a constant. For example, the distribution $p(x)$ may have no closed-form solution for the inverse cumulative function and no efficient approximate sampling method, but it may have a

[1] An estimator is unbiased if its expected value is the same as the estimated quantity, that is, if, on average, the estimator is not offset from the estimated quantity.

[2] An even stronger statement can be made when the variance σ_f^2 of the estimated function $f(x)$ with respect to the distribution $p(x)$ is finite. In this case, the central limit theorem holds, and the expression $\sqrt{N}(\bar{I}_N(f) - \mathbb{E}_{p(x)}[f(x)])$ converges in distribution to the normal distribution $\mathcal{N}(0, \sigma_f^2)$.

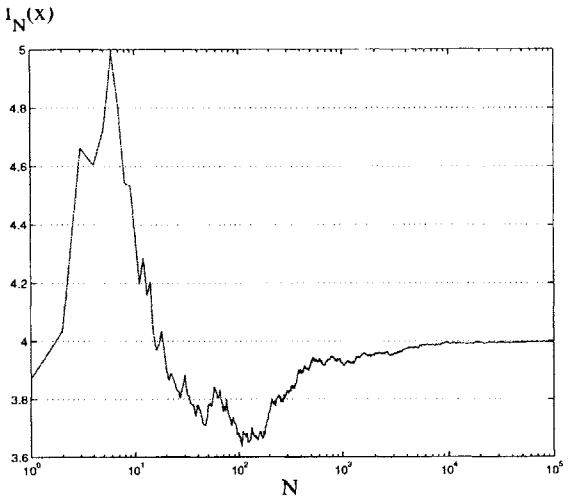

Figure 3-3: Estimates $I_N(x)$ of the mean of the distribution $p(x)$ from Figure 3-2. The estimates were computed from a single sequence of samples $x^{(i)}$ and converge to the true mean $\mathbb{E}[x] = 4$.

functional form of the p.d.f. Such distributions are common, for example, in nonlinear systems, in which the posterior distributions are often non-standard, but their p.d.f.s can be easily evaluated as a product of the prior model and the observation(s) up to the normalizing constant $\frac{1}{p(\mathbf{y}_{0:t})}$.[3] In these cases, we can apply the importance sampling method [9]. This method is based on the observation that even if we are unable to take samples from the target distribution $p(x)$, we might be able to take samples from a different distribution $q(x)$ and adjust for the difference by assigning a weight to each sample. In this manner, we can still approximate the target distribution $p(x)$ and any of its characteristics.

The process is illustrated in Figure 3-4. Suppose that the target distribution $p(x)$, shown on the left, is difficult to sample efficiently. Thus, we take samples from another distribution, $q(x)$, that can be easily sampled; we refer to this distribution as *the proposal distribution*, or simply *the proposal*.[4]. For example, in Figure 3-4, the samples were taken from a normal distribution with mean 5 and variance 1; hence, some of the samples fell at the tails at 3 and 7, even though $p(x) \approx 0$ there. In order

[3]For now, we do not explicitly condition the distribution $p(x)$ on the observations, since the concepts herein apply to arbitrary distributions. The conditioning on the observations will be introduced in the next subsection.

[4]In some literature, it is also referred to as the *importance sampling distribution*.

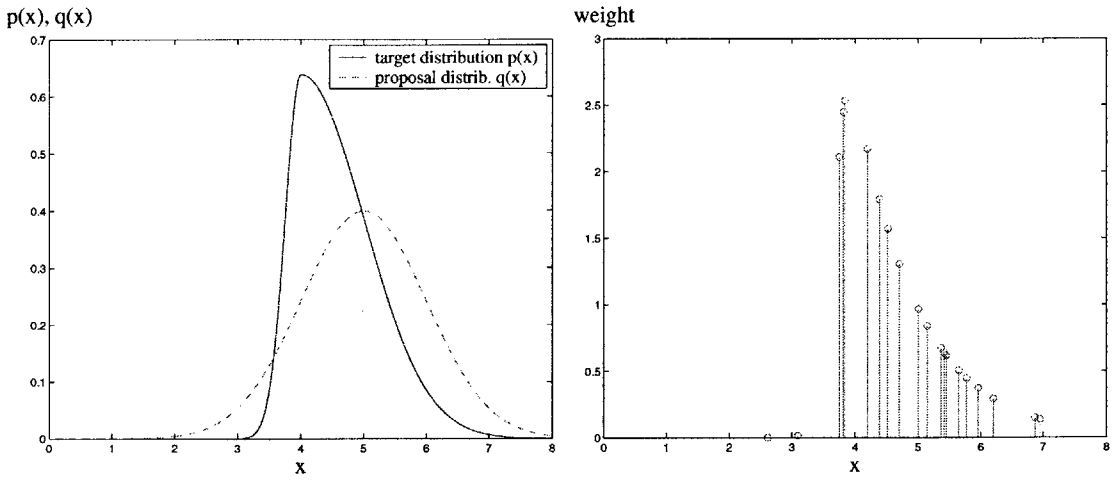

Figure 3-4: Left: The desired target distribution and the sampled proposal distribution. Right: Generated samples and their weights.

to account for the discrepancy between the two distributions, each sample has an associated weight, which is equal to the ratio between the target and the proposal p.d.f. at the sampled point. Intuitively, where the posterior distribution $p(x)$ is much lower than the sampled distribution $q(x)$, the weights will be low; where $p(x) \gg q(x)$, the weights will be high.

As before, the samples, along with their weights, approximate the target distribution $p(x)$. This fact is illustrated in Figure 3-5. Similarly, we can approximate the expected value of any function $f(x)$ with respect to $p(x)$ by taking a weighted average of the function at the sampled points:

$$\mathbb{E}_{p(x)}[f(x)] \approx \frac{\sum_{i=1}^{N} f(x^{(i)})w(x^{(i)})}{\sum_{i=1}^{N} w(x^{(i)})} = \sum_{i=1}^{N} f(x^{(i)})\tilde{w}(x^{(i)}) \triangleq \bar{I}_N^1(f), \tag{3.7}$$

where $w(x^{(i)}) \triangleq \frac{p(x^{(i)})}{q(x^{(i)})}$ is the weight of the i-th particle and $\tilde{w}(x^{(i)}) \triangleq \frac{w(x^{(i)})}{\sum_{i=1}^{N} w(x^{(i)})}$ is the normalized weight.

As an example, suppose that we wish to compute the probability that $x < c$ w.r.t. $p(x)$ for some constant c, given that $q(x)$ is uniform over the (bounded) domain of x. This problem is equivalent to finding the expected value of the decision variable

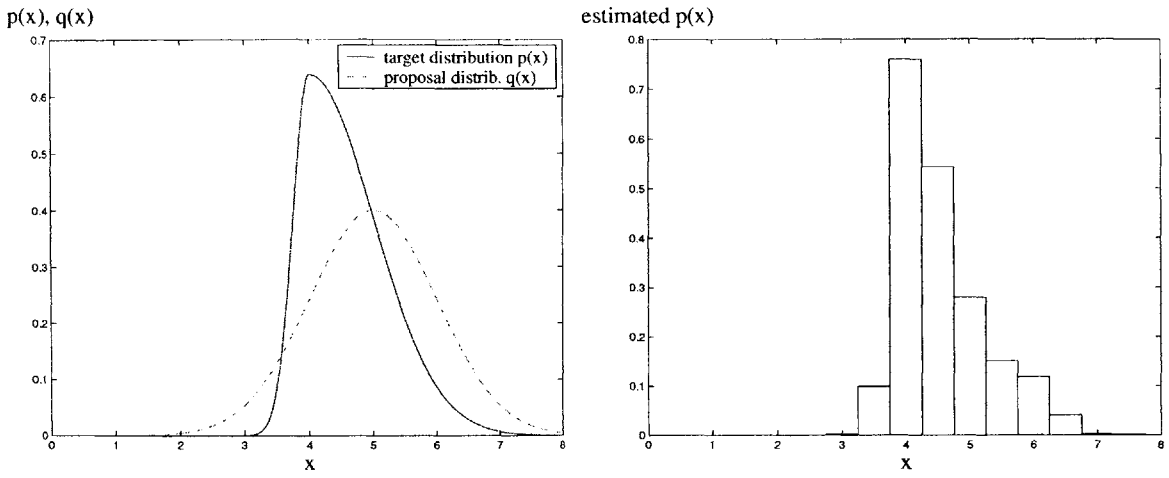

Figure 3-5: Left: Approximated distribution $p(x)$ and sampled distribution $q(x)$. Right: Histogram of 100 i.i.d. samples taken from $q(x)$, weighted according to $\frac{p(x)}{q(x)}$.

(function)

$$f = \begin{cases} 1 & \text{if } x < c, \\ 0 & \text{otherwise,} \end{cases} \tag{3.8}$$

because then $\mathbb{E}[f] = 1 \cdot Pr[x < c] + 0 \cdot Pr[x \geq c] = Pr[x < c]$. Since $q(x)$ is constant, the estimator 3.7 simplifies to

$$I_N^1(f) = \frac{\sum_{i=1}^N f(x^{(i)})p(x^{(i)})}{\sum_{i=1}^N p(x^{(i)})} = \frac{\sum_{i:f(x^{(i)}<c)} p(x^{(i)})}{\sum_{i=1}^N p(x^{(i)})}. \tag{3.9}$$

In other words, the estimator computes the relative density of those samples that fall in the region of interest $x < c$. This is analogous to approximating the integral $\int_{x<c} p(x)dx$ with a Riemann sum with fixed interval lengths.

The estimator 3.7 is not, in general, unbiased, but it does converge asymptotically a.s to the estimated quantity $\mathbb{E}_{p(x)}[f(x)]$.[5] First, note that the numerator is the Monte Carlo estimator I_N of the function $f(x)w(x)$ (see Equation 3.5) and the denominator

[5]For clarity, we sometimes explicitly denote the distribution w.r.t. which the expectation is taken using the subscript.

is the Monte Carlo estimator I_N of the function $w(x)$ w.r.t. the distribution $q(x)$:

$$\bar{I}_N^1(f) \triangleq \frac{\sum_{i=1}^N f(x^{(i)})w(x^{(i)})}{\sum_{i=1}^N w(x^{(i)})} = \frac{I_N(fw)}{I_N(w)}. \tag{3.10}$$

Although both estimators $I_N(fw)$ and $I_N(w)$ are unbiased, their ratio is not necessarily unbiased, because, in general, $\mathbb{E}[\frac{a}{b}] \neq \frac{\mathbb{E}[a]}{\mathbb{E}[b]}$. Thus, in repeated trials with different sets of samples, the estimator $\bar{I}_N$ will not be centered at $\mathbb{E}_{p(x)}[f(x)]$. However, since these estimators converge a.s. to $\mathbb{E}_{q(x)}[f(x)w(x)]$ and $\mathbb{E}_{q(x)}[w(x)]$, respectively, their ratio converges asymptotically a.s. to the estimated value $\mathbb{E}_{p(x)}[f(x)]$ under the following assumptions [25]:

1. the estimated value $\mathbb{E}_{p(x)}[f(x)]$ is finite
2. the support of $q(x)$ includes the support of $p(x)$[6] and
3. the expectations of w_t and $w_t f_t^2(x)$ are finite.

Intuitively, the second assumption ensures that the probability that importance sampling would miss a region of $p(x)$ with non-zero probability will tend to zero as $N \to +\infty$.

Finally, note that the importance sampling method subsumes the perfect (direct) sampling method, since we can always let $q(x) = p(x)$. In this case, the weights $w(x^{(i)})$ are always equal to $\frac{1}{N}$, and the estimator $\bar{I}_N^1(f)$ (Equation 3.7) simplifies to

$$\frac{\sum_{i=1}^N f(x^{(i)})\frac{1}{N}}{\sum_{i=1}^N \frac{1}{N}} = \frac{1}{N}\sum_{i=1}^N f(x^{(i)}) = I_N(f). \tag{3.11}$$

In reality, particle filters reason in terms of vectors of random variables, rather than a single variable x. Thus, rather than taking samples $x^{(i)}$ from a single variable x, we will take random samples $\mathbf{x}^{(i)}$ from a *vector* of variables $\mathbf{x}$ according to some distribution $p(\mathbf{x})$. For example, in order to localize a rover in a known environment, we may consider taking samples from a 2-dimensional space of coordinates $\langle x, y \rangle$, as shown in Figure 3-7. Furthermore, we will consider the evolution of the random vector

[6]Support of a function q is the closure of the set, for which $q(x) \neq 0$.

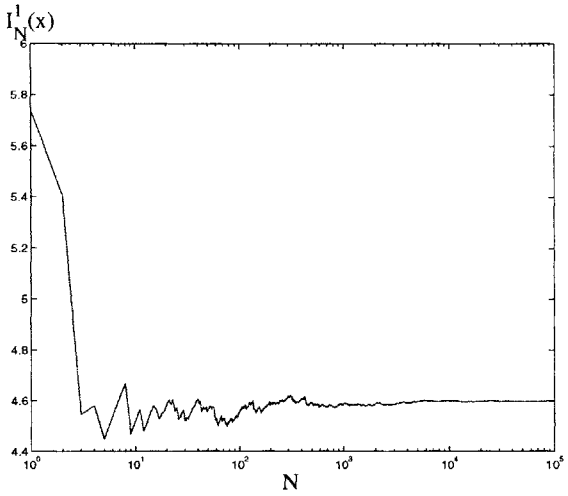

Figure 3-6: Estimates $I_N(x)$ of the mean of the distribution $p(x)$ when the samples were taken from the distribution $q(x)$ in Figure 3-4. The estimates were computed using a single sequence of samples $x^{(i)}$ and converge to the true mean $\mathbb{E}[x] = 4.5984$.

$\mathbf{x}$ in discrete time-steps. Conceptually, this evolution corresponds to introducing a vector of random variables at each time step up to the present:

$$\mathbf{x}_{0:t} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \quad \cdots \quad \begin{bmatrix} x_t \\ y_t \end{bmatrix}. \tag{3.12}$$

All the principles and results introduced above for a single random variable hold also in this more general setting. For example, there is a duality between any distribution $p(\mathbf{x}_{0:t})$ over random vectors $\mathbf{x}_0, \ldots, \mathbf{x}_t$ and the samples $\mathbf{x}_{0:t}^{(i)}$ taken from this distribution. This duality implies that one can generate (at least in principle) random samples from the distribution $p(\mathbf{x}_{0:t})$, and, in turn, the samples $\mathbf{x}_{0:t}^{(i)}$ can be used to reconstruct the original distribution. Similarly, as in the single-dimensional case, if we can take i.i.d. samples from a distribution $p(\mathbf{x}_{0:t})$, we can approximate the expected value of any function $\mathbf{f}(\mathbf{x})$ by taking a weighted sum of the function at the points given by the samples $\mathbf{x}_{0:t}^{(i)}$.

To summarize, we can take samples and use them to approximate the distribution that generated them. The samples can be used to approximate any characteristics of the distribution that can be expressed as $\mathbb{E}[\mathbf{f}(\mathbf{x})]$ for some function $\mathbf{f}$. Examples of such characteristics include the MMSE estimate and the variance of the distribution.

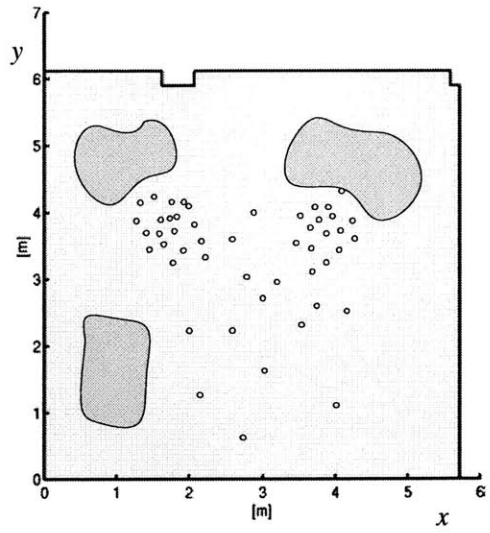

Figure 3-7: Two-dimensional state space representing the set of coordinates, where a robot can be located. Dark regions represent the obstacles; dots represent the sampled positions.

When we cannot take samples directly from the distribution $p(\mathbf{x})$, we can apply the importance sampling method and take samples from a different distribution $q(\mathbf{x})$. With sufficiently large number of samples N, one can obtain a good approximation of any characteristics $\mathbb{E}[\mathbf{f}(\mathbf{x})]$ by taking a weighted sum of the function $\mathbf{f}$ at the sampled points.

With these principles in mind, we turn to the algorithm that estimates the state of a dynamic system using importance sampling.

3.2.2 Sequential importance sampling

As discussed previously, our goal is to approximate the posterior $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ and its interesting characteristics, such as the minimum mean square error (MMSE) estimate $\mathbb{E}[\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}]$ and the marginal posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, also called the *filtering distribution*. In view of the previous section, we estimate these characteristics by taking random samples from the set of evolutions of the hidden state, $\mathbf{x}_{0:t} \triangleq \langle \mathbf{x}_0\mathbf{x}_1, \ldots, \mathbf{x}_t \rangle$. The desired characteristics, such as MMSE estimate, can be then approximated by taking the weighted average of these samples, as was done in the estimator $I_N^1(f)$ (Equation 3.5).

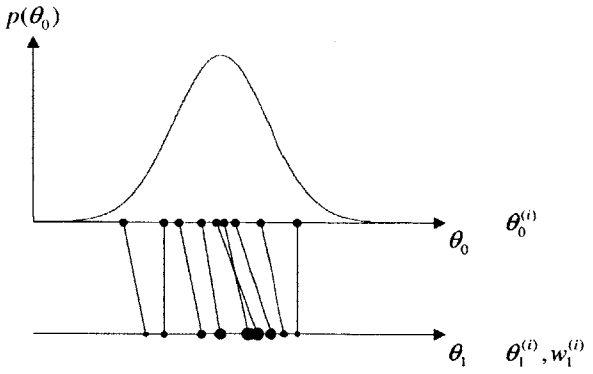

Figure 3-8: Evolution fo the samples $\theta_{0:t}$.

The key idea to make this computation suitable for recursive estimation it to evolve the samples sequentially, from one time step to another. This process is illustrated in Figure 3-8. At the beginning, we take N i.i.d. samples from the initial distribution $p(\mathbf{x}_0)$; hence approximating the posterior at time $t = 0$. Then, in each time step, we evolve each sample $\mathbf{x}_{0:t-1}^{(i)}$ according to some proposal distribution $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ and update its importance weight $w_t^{(i)}$. The weight $w_t^{(i)}$ reflects the discrepancy between the proposal q and the desired posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, such that the resulting weighted samples converge to the posterior.

The importance weights are chosen as

$$w_t^{(i)} = \begin{cases} w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})} & \text{if } t > 0 \\ 1 & \text{if } t = 0 \end{cases} \tag{3.13}$$

To see why this choice guarantees convergence, first note that the samples $\mathbf{x}_{0:t}^{(i)}$ are distributed according to the proposal distribution $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, where

$$q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) \triangleq p(\mathbf{x}_0) \prod_{k=1}^{t} q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}). \tag{3.14}$$

The weights $w_t^{(i)}$ satisfy the following equality:

$$w_t^{(i)} = w_0^{(i)} \prod_{k=1}^{t} \frac{p(\mathbf{y}_k|\mathbf{x}_{0:k}^{(i)}, \mathbf{y}_{0:k-1}, \mathbf{u}_{0:k}) p(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{0:k-1}, \mathbf{u}_{0:k})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{0:k}, \mathbf{u}_{0:k})} \tag{3.15}$$

$$= \frac{p(\mathbf{x}_0) \prod_{k=1}^{t} p(\mathbf{y}_k|\mathbf{x}_{0:k}^{(i)}, \mathbf{y}_{0:k-1}, \mathbf{u}_{0:k}) p(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{0:k-1}, \mathbf{u}_{0:k})}{p(\mathbf{x}_0) \prod_{k=1}^{t} q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{0:k}, \mathbf{u}_{0:k})} \tag{3.16}$$

$$= \frac{p(\mathbf{y}_{1:t}, \mathbf{x}_{0:t}|\mathbf{u}_{0:t})}{q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})} \tag{3.17}$$

$$= \frac{p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})}{q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})} \frac{1}{p(\mathbf{y}_{1:t}|\mathbf{u}_{0:t})} \tag{3.18}$$

Hence, the algorithm is just a special case of importance sampling, discussed in the previous section, with the proposal distribution $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ equal to the product of the proposal distributions $q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{0:k}, \mathbf{u}_{0:k})$ at individual time step, and the approximated distribution equal the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$. The factor $\frac{1}{p(\mathbf{y}_{1:t}|\mathbf{u}_{0:t})}$ in the weights is independent of the state variables $\mathbf{x}_{0:t}$ and thus, does not affect the desired estimate $\bar{I}_N^1(f)$ (see Equation 3.7). The resulting sequential importance sampling (SIS) algorithm is shown in Figure 3-9.

Choice of the proposal distribution

The proposal distribution is chosen as necessary to allow efficient sampling for the given problem or the domain. In its simplest form, the proposal distribution takes the form of the prior transition distribution $p(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{u}_{0:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$. In this case, the samples are evolved according to the transition model and the incremental weights simplify to the observation likelihood $p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) = p(\mathbf{y}_t|\mathbf{x}_t^{(i)}, \mathbf{u}_t)$:[7]

$$w_t^{(i)} \triangleq w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})}$$

$$= w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{p(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}$$

$$= w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) \tag{3.20}$$

[7]We choose not to simplify the transition and observation distributions in the derivations to $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ and $p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{u}_t)$, because particle filters apply also models that are not strictly Markovian. This more general non-Markovian form will be used later in Chapter 4.

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_0^{(i)}$ from the prior distribution $p(\mathbf{x}_0)$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_t^{(i)}$ from the proposal $q(\mathbf{x}_t | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})$
     - let $\mathbf{x}_{0:t}^{(i)} \leftarrow (\mathbf{x}_{0:t-1}^{(i)}, \mathbf{x}_t^{(i)})$
   - For $i = 1, \ldots, N$, compute the importance weights:

$$w_t^{(i)} \leftarrow w_{t-1}^{(i)} \frac{p(\mathbf{y}_t | \mathbf{x}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_t^{(i)} | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\mathbf{x}_t^{(i)} | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})} \tag{3.19}$$

   - For $i = 1, \ldots, N$ normalize the importance weights $w_t^{(i)}$

Figure 3-9: Sequential importance sampling algorithm.

The importance sampling step is then entirely analogous to the prediction-update sequence in other filtering methods: first, we predict the state $\tilde{\mathbf{x}}_t$ using the estimate at the previous time step and then we adjust the prediction using the newest observation.

The reason why the proposal distribution $p(\mathbf{x}_t | \mathbf{x}_{t-1})$ is popular [26, 40] is that stochastic models are often written in the form

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) + \mathbf{v}_x \tag{3.21}$$

$$\mathbf{y}_t = \mathbf{g}(\mathbf{x}_t, \mathbf{u}_{t-1}) + \mathbf{v}_y, \tag{3.22}$$

where $\mathbf{v}_x$ is a noise variable with a Gaussian or other distribution that can be sampled efficiently, and $\mathbf{v}_y$ is a noise variable, whose p.d.f. can be evaluated easily. In this case, it is easy to sample from the proposal distribution by propagating the sample $\mathbf{x}_{t-1}^{(i)}$ through the function $\mathbf{f}$ and sampling from the distribution $\mathbf{f}(\mathbf{x}_{t-1}^{(i)}, \mathbf{u}_{t-1}) + \mathbf{v}_x$.

It has been shown that the distribution $q(\mathbf{x}_t^{(i)} | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t}) = p(\mathbf{x}_t | \mathbf{x}_{t-1})$ is optimal in the sense that it minimizes the variance of the importance weights. [20]

Unforunately, this distribution often results in untractable integrals. Therefore, it is instead common to use a proposal that is "close" to the optimal proposal distribution. See [57] for an example.

Degeneracy of sequential importance sampling

Unfortunately, while the SIS algorithm guarantees convergence for a fixed t, it is not usable in practice. After a while, the fittest sample will tend to have a normalized weight of 1, while all the other ones will be nearly zero. Thus, the filter will degenerate to tracking a single sequence in the continuous state, and will no longer approximate the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ well. The following proposition is taken from [41]:

Proposition 1 *The unconditional variance of the weights $\tilde{w}_t$ (that is, the variance of the weights $\tilde{w}_t$, when the observations are regarded as random) increases over time.[41]*

To see how this relates to a single run of the filter and the performance of the SIS algorithm, note that

$$var(\tilde{w}_t) = \mathbb{E}[var(\tilde{w}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})] + var(\mathbb{E}[\tilde{w}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}]) \tag{3.23}$$

Since $\mathbb{E}[\tilde{w}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}] = 1$, its variance is 0, and the mean (conditional) variance of the importance weights

$$\mathbb{E}[var(\tilde{w}_t|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})] = var(\tilde{w}_t) \tag{3.24}$$

will increase over time. This means that the variance of the weights will typically increase, and most of the density in the distribution $\sum_{i=1}^{N} \tilde{w}_t^{(i)} \delta(\mathbf{x}_{0:t}^{(i)})$ will be concentrated in a small region of the state space.

3.2.3 Selection

In order to avoid the degeneracy of the Sequential importance sampling method, an additional resampling (selection) step is needed (see Figure 3-10). This step multiplies the particles with high weight and removes the ones with the low weight, so that in the

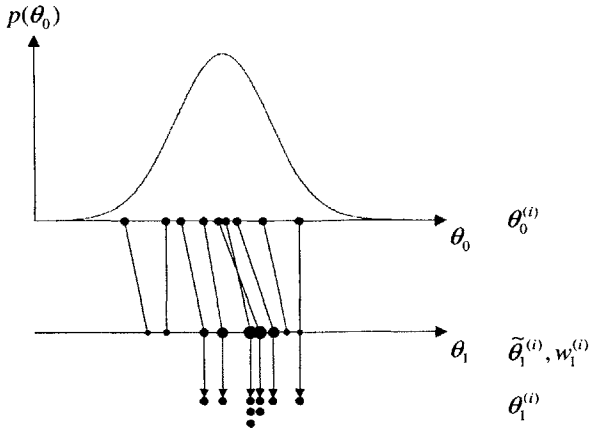

Figure 3-10: Importance sampling with an additional selection step. After the samples $\theta_t^{(i)}$ are evolved, they are resampled according to their importance weights.

next time step, the high-likelihood particles contribute more to the sampling process. Intuitively, the particles with high importance ratio are more likely to contribute to the regions of high posterior probability.

A selection scheme associates with each particle i a number of off-springs, denoted as N_i, such that $\sum_{i=1}^{N} N_i = N$. The off-springs are *unweighted*, that is, the selection scheme replaces a weighted particle $\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}$ with N_i unweighted particles $w_t^{(i)} = N^{-1}$. Several selection schemes have been proposed in the literature. An early example of such a strategy is sampling-importance resampling (SIR), which selects N random i.i.d. samples (with repetition) from $\{\mathbf{x}_{0:t}^{(i)}\}$ according to the weights, $w_t^{(i)}$. [26] Examples of other strategies include residual resampling [31, 46] and stratified/systematic sampling [40]. All of these strategies satisfy the equality $\mathbb{E}[N_i] = N w_t^{(i)}$, but vary in terms of how much variance they introduce in the number of offsprings N_i. See [13] for the theoretical treatment of the subject and converge proofs for several selection schemes.

The final algorithm is shown in Figure 3-11. To distinguish between the samples before and after the selection step, the samples in the importance sampling step are marked with a tilde ($\tilde{\mathbf{x}}_{0:t}^{(i)}$).

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_0^{(i)}$ from the prior distribution $p(\mathbf{x}_0)$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - draw a random sample $\tilde{\mathbf{x}}_t^{(i)}$ from the proposal $q(\mathbf{x}_t | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})$
     - let $\tilde{\mathbf{x}}_{0:t}^{(i)} \leftarrow (\mathbf{x}_{0:t-1}^{(i)}, \tilde{\mathbf{x}}_t^{(i)})$
   - For $i = 1, \ldots, N$, compute the importance weights:

$$w_t^{(i)} \leftarrow \frac{p(\mathbf{y}_t | \tilde{\mathbf{x}}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\tilde{\mathbf{x}}_t^{(i)} | \tilde{\mathbf{x}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\tilde{\mathbf{x}}_t^{(i)} | \tilde{\mathbf{x}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})} \tag{3.25}$$

   - For $i = 1, \ldots, N$ normalize the importance weights $w_t^{(i)}$

   (b) Selection step

   - Select N samples (with replacement) from $\{\tilde{\mathbf{x}}_{0:t}^{(i)}\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain samples $\{\mathbf{x}_{0:t}^{(i)}\}$

Figure 3-11: Generic particle filter.

3.3 Filter implementation for PHA

Sequential Monte Carlo methods, which were overviewed in the previous section, apply to a wide range of discrete and continuous stochastic processes that satisfy the Markovian property and whose observations are conditionally independent given the process. Their adaptation to Probabilistic Hybrid Automata is straightforward, provided that we sample the complete hybrid state and evolve the samples according to the transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$. In this section, we present a simple particle filter, which directly applies the concepts presented in the previous section to PHA. While more advanced particle filtering algorithms exist for estimating state with hybrid models, including the Risk-sensitive paritcle filter [55], Variable resolution particle filter [58], and a Rao-Blackwellised particle filter in [22], we introduce this algorithm here to motivate the discussion in the following chapter. For a discussion of other particle filtering approaches and their comparison to our Gaussian particle filter, see Section 4.4.

Figure 3-12 illustrates the algorithm on a model with one discrete variable domain $\{\texttt{on}, \texttt{off}\}$ and one continuous variable x_c. The algorithm maintains samples over the complete hybrid state space; each particle $\mathbf{x}_{0:t}^{(i)}$ consists of a discrete sample $\mathbf{x}_{d,0:t}^{(i)}$ and a continuous sample $\mathbf{x}_{c,0:t}^{(i)}$. The algorithm starts by drawing samples from the initial distribution $p(\mathbf{x}_0)$; thus, effectively approximating the posterior at time $t = 0$. The initial state distribution in PHA is specified as a distribution over the mode variables $p(\mathbf{x}_{d,0})$ and, for each mode $\mathbf{m}$, the associated normal distribution over the continuous state $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0} = \mathbf{m})$. Thus, the initial samples $\mathbf{x}_0^{(i)}$ are generated by first taking a random samples $\mathbf{x}_{d,0}^{(i)}$ according to the prior distribution $p(\mathbf{x}_{d,0})$ and then, for each discrete sample $\mathbf{x}_{d,0}^{(i)}$, by taking a corresponding continuous sample $\mathbf{x}_{c,0}$ according to the distribution $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$.

Given our choice of the proposal distribution

$$q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t}) = p(\mathbf{x}_{d,t}, \mathbf{x}_{c,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1}), \tag{3.26}$$

the algorithm evolves the particles in two steps (see Figure 3-13):

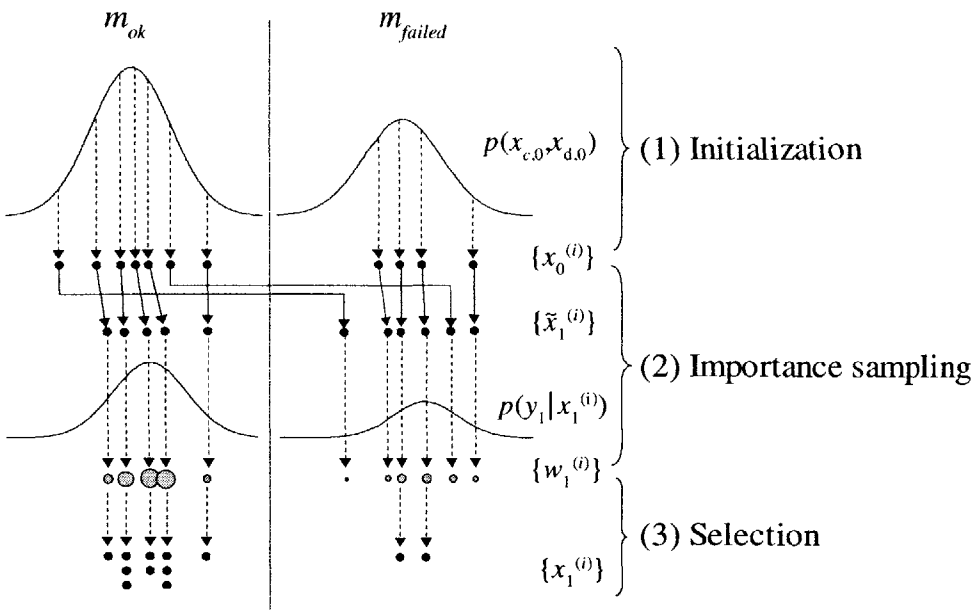

Figure 3-12: Bootstrap filter for a hybrid model with one mode variable and one continuous variable.

1. **discrete (mode) evolution**, which generates a random *discrete* sample $\mathbf{x}_{d,t}$ according to the discrete transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1})$ and

2. **continuous evolution**, which generate a random *continuous* sample according to $p(\mathbf{x}_{c,t}|\mathbf{x}_{d,t}^{(i)}, \mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1})$, using the new discrete sample $\mathbf{x}_{d,t}^{(i)}$.

Recall that in PHA, discrete transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1})$ is specified as a set of pairs $\{\langle p_\tau, c\rangle\}$ of transition distributions p_τ and the associated transition guards c (see Figure 2-1). The guards determine, which transition distribution applies to the current discrete state, and may depend on the current continuous state $\mathbf{x}_c$ and the input $\mathbf{u}$. For example, when `has-ball=yes` and $\theta_1 = 0.75$ in the model in Figure 2-1, the system is as likely to transition to the mode `no` as it is to stay in the mode `yes`. It follows that, in order to generate the discrete sample $\mathbf{x}_{d,t}^{(i)}$, we need to determine which transition guard in $T(\mathbf{x}_{d,t-1}^{(i)})$ is satisfied by the continuous sample $\mathbf{x}_{c,t-1}^{(i)}$ and the input $\mathbf{u}_{t-1}$. This guard gives us a (unique) distribution $p_\tau(\mathbf{x}_{d,t})$ to generate the sample $\mathbf{x}_{d,t}^{(i)}$.

Given a mode assignment at time t, the continuous evolution and observation of the system is determined by the equations associated with that mode. For exam-

ple, if the acrobatic robot is in the mode $\mathbf{x}_{d,t}^{(i)} = \langle \texttt{actuator=ok}, \texttt{has-ball=yes} \rangle$, its continuous state evolves and is observed according to the equations

$$\mathbf{x}_{c,t} = \mathbf{f}(\mathbf{x}_{c,t-1}, \mathbf{u}_{c,t-1}; \mathbf{x}_{d,t}^{(i)}) + \mathbf{v}_x(\mathbf{x}_{d,t}) \tag{3.27}$$

$$\mathbf{y}_{c,t} = \mathbf{g}(\mathbf{x}_{c,t}, \mathbf{u}_{c,t}; \mathbf{x}_{d,t}^{(i)}) + \mathbf{v}_y(\mathbf{x}_{d,t}) \tag{3.28}$$

that take into account the higher weight m_2 at the robot's legs. Hence, it is easy to evolve the continuous sample $\mathbf{x}_{c,t-1}^{(i)}$ once we condition on the new mode of the system $\mathbf{x}_{d,t}^{(i)}$ by taking a random sample from the distribution

$$\mathcal{N}(\mathbf{f}(\mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1}; \mathbf{x}_{d,t}^{(i)}), cov(\mathbf{v}_x(\mathbf{x}_{d,t}^{(i)}))). \tag{3.29}$$

Finally, since the samples are evolved according to the transition prior, the weights simplify to the observation likelihood $p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{u}_t)$, as shown in Equation 3.20. From Equation 3.28, the observation likelihood is given by

$$\mathcal{N}(\mathbf{y}_t, g(\mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1}; \mathbf{x}_{d,t}^{(i)}), cov(\mathbf{v}_y(\mathbf{x}_{d,t}^{(i)}))), \tag{3.30}$$

where $N(\mathbf{y}, \mu, \Lambda)$ is the p.d.f. of the normal distribution with the mean μ and covariance Λ, evaluated at $\mathbf{y}$. The complete filter algorithm for PHA is shown in Figure 3-13.

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_{d,0}^{(i)}$ from the prior distribution $p(\mathbf{x}_{d,0})$
     - draw a random sample $\mathbf{x}_{c,0}^{(i)}$ from the normal distribution $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - draw a random discrete sample $\tilde{\mathbf{x}}_{d,t}^{(i)}$ from the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1})$
     - draw a random continuous sample $\tilde{\mathbf{x}}_{c,t}^{(i)}$ from the normal distribution $\mathcal{N}(\mathbf{f}(\mathbf{x}_{c,t-1}^{(i)}, \mathbf{u}_{t-1}; \tilde{\mathbf{x}}_{d,t}^{(i)}), cov(\mathbf{v}_x(\tilde{\mathbf{x}}_{d,t}^{(i)})))$
     - let $\tilde{\mathbf{x}}_{0:t}^{(i)} \leftarrow (\mathbf{x}_{0:t-1}^{(i)}, \langle \tilde{\mathbf{x}}_{d,t}^{(i)}, \tilde{\mathbf{x}}_{c,t}^{(i)} \rangle)$
   - For $i = 1, \ldots, N$, compute the importance weights:

$$w_t^{(i)} \leftarrow \mathcal{N}(\mathbf{y}_t, g(\tilde{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{u}_{t-1}; \tilde{\mathbf{x}}_{d,t}^{(i)}), cov(\mathbf{v}_y(\tilde{\mathbf{x}}_{d,t}^{(i)}))) \tag{3.31}$$

   - For $i = 1, \ldots, N$ normalize the importance weights $w_t^{(i)}$

   (b) Selection step

   - Select N samples (with replacement) from $\{\tilde{\mathbf{x}}_{0:t}^{(i)}\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain samples $\{\mathbf{x}_{0:t}^{(i)}\}$

Figure 3-13: Bootstrap particle filter for PHA.

Chapter 4

Gaussian Particle Filtering for PHA

In practice, sampling in high-dimensional spaces can be inefficient, since many particles may be needed to cover the probability space and attain a sufficiently accurate estimate. Several methods have been developed to reduce the variance of the estimates, including antithetic sampling [28, 27], control variates [5, 27], and, more recently, decomposition [50] and abstraction [58]. In this chapter, we apply the technique of Rao-Blackwellisation [6, 12, 21] to particle filtering in Probabilistic Hybrid Automata (PHA). This technique is based on a fundamental observation that if the model has a tractable substructure, we may be able to factor it out with an efficient solution only sample the remaining variables. In this manner, fewer samples are needed to attain a given accuracy of the estimate.

The key contribution of this chapter is a an approximate Rao-Blackwellised particle filtering (RBPF) algorithm, which handles the nonlinearities and autonomous transitions (mode transitions dependent on the continuous state), present in PHA models. In the spirit of prior approaches in RBPF [6, 48] and k-best filtering [44, 32], our algorithm samples the mode sequences and, conditioned on each sampled sequence, estimates the associated continuous state with an Extended [7] or an Unscented Kalman Filter [39]. The key insight to addressing autonomous transitions is to reuse the continuous estimates in the importance sampling step of the filter.

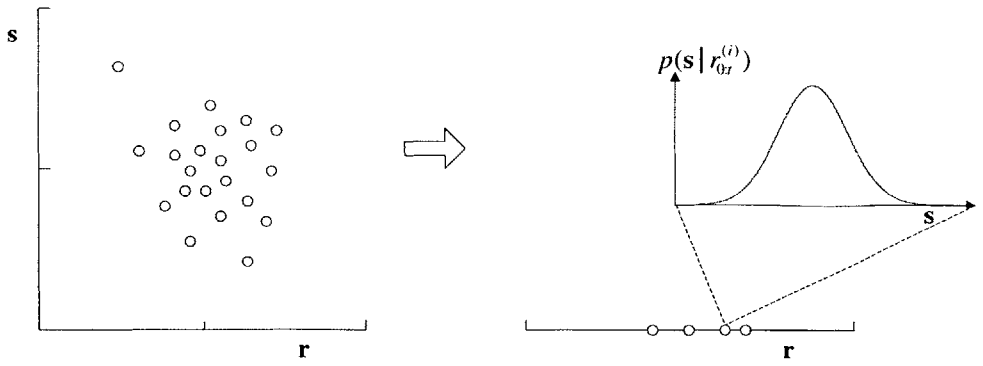

Figure 4-1: Rao-Blackwellised particle filtering.

We extend the class of autonomous transitions that can be addressed efficiently and demonstrate, how the algorithm bridges the prior work in Rao-Blackwellised particle filtering and hybrid model-based diagnosis.

4.1 Rao-Blackwellised Particle Filtering

Rao-Blackwellised particle filtering (RBPF) [6, 21] is an extension to the generic particle filtering algorithm, described in Section 3.2.3. If we partition the state variables into two sets, $\mathbf{r}$ and $\mathbf{s}$ (see Figure 4-1), we can use the chain rule to express the posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ as

$$\begin{aligned} p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) &= p(\mathbf{s}_{0:t}, \mathbf{r}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) \\ &= p(\mathbf{s}_{0:t}|\mathbf{r}_{0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) p(\mathbf{r}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) \end{aligned} \tag{4.1}$$

Thus, we expand the posterior in terms of the sequence of random variables $\mathbf{r}_{0:t}$ and in terms of the sequence $\mathbf{s}_{0:t}$ conditioned on $\mathbf{r}_{0:t}$. The key to this formulation is that if we can compute the conditional distribution $p(\mathbf{s}_{0:t}|\mathbf{r}_{0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ or its marginal $p(\mathbf{s}_t|\mathbf{r}_{0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ analytically, we only need to sample the sequences of variables $\mathbf{r}_{0:t}$. Intuitively, fewer particles will be needed in this way to reach a given precision of the estimate, since for each sampled sequence $\mathbf{r}_{0:t}$, the corresponding state space $\mathbf{s}$ is covered by an analytical solution rather than a finite number of samples.

In RBPF, each particle holds not only the samples $\mathbf{r}_{0:t}^{(i)}$, but also a *parametric*

representation of the distribution $p(\mathbf{s}_t|\mathbf{r}_{0:t}^{(i)}, \mathbf{y}_{1:t})$, which we denote by $\alpha_t^{(i)}$. This representation holds sufficient statistics for $p(\mathbf{s}_t|\mathbf{r}_{0:t}^{(i)}, \mathbf{y}_{1:t})$, such as the mean vector and the covariance matrix of the distribution. The posterior is thus approximated as a mixture of the distributions $\alpha_t^{(i)}$ at the sampled points $\mathbf{r}_{0:t}^{(i)}$:

$$p(\mathbf{s}_{0:t}, \mathbf{r}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t}) \approx \sum_{1}^{N} \alpha_t(i)\delta_{\mathbf{r}_{0:t}^{(i)}}(\mathbf{r}_{0:t}). \tag{4.2}$$

A generic RBPF method is outlined in Figure 4-2, and, except for the initialization and the exact step, it is identical to the generic particle filter from Section 3.2.3. In each time step, we evolve the samples $\mathbf{r}_{0:t}^{(i)}$, according to a suitable proposal distribution $q(\mathbf{r}_t|\mathbf{r}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})$. This distribution depends entirely on the structure of the problem, and is discussed at a greater depth in Section 4.3.2. In order to account for the discrepancy between the proposal distribution and the desired posterior distribtion $p(\mathbf{r}_{0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, we assign importance weights

$$w_t^{(i)} = \frac{p(\mathbf{y}_t|\tilde{\mathbf{r}}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})p(\tilde{\mathbf{r}}_t^{(i)}|\tilde{\mathbf{r}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\tilde{\mathbf{r}}_t^{(i)}|\tilde{\mathbf{r}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})}, \tag{4.3}$$

to the particles i (see Equation 3.13). Then, we multiply/discard the samples according to their weights $w_t^{(i)}$, using one of the selection schemes, as discussed in Section 3.2.3. The last, exact step updates the posterior distribution.$\alpha_t^{(i)}$ over the variable $\mathbf{r}_t$ using the newly evolved sample $r_t^{(i)}$ and the latest observation $\mathbf{y}_t$ and inputs.

The RBPF method places no restriction how the state space should be partitioned, other than that it must be possible to update the conditional distribution $\alpha_t^{(i)} = p(\mathbf{s}_{0:t}|\mathbf{r}_{0:t}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ efficiently. A common practice in the field is to factor out as much of the tractable space as possible. Murphy and Russell [49] advocate an iterative procedure for dynamic Bayesian networks [14], whereby the set of variables $\mathbf{r}$ gets expanded until the set of remaining variables $\mathbf{s} = \mathbf{x} \setminus \mathbf{r}$ can be updated exactly and efficiently. For switching linear models, where the continuous dynamics of the system is linear conditioned on the discrete mode variables (see Section 4.2.1), it is common

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{r}_0^{(i)}$ from the prior distribution $p(\mathbf{r}_0)$
     - let $\alpha_0^{(i)} \leftarrow p(\mathbf{s}_0|\mathbf{r}_0^{(i)})$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - draw a random sample $\tilde{\mathbf{r}}_t^{(i)}$ from the proposal $q(\mathbf{r}_t|\mathbf{r}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})$
     - let $\tilde{\mathbf{r}}_{0:t}^{(i)} \leftarrow (\mathbf{r}_{0:t-1}^{(i)}, \tilde{\mathbf{r}}_t^{(i)})$
   - For $i = 1, \ldots, N$, compute the importance weights:

$$w_t^{(i)} \leftarrow \frac{p(\mathbf{y}_t|\tilde{\mathbf{r}}_{0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) p(\tilde{\mathbf{r}}_t^{(i)}|\tilde{\mathbf{r}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t})}{q(\tilde{\mathbf{r}}_t^{(i)}|\tilde{\mathbf{r}}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})} \tag{4.4}$$

   - For $i = 1, \ldots, N$ normalize the importance weights $w_t^{(i)}$

   (b) Selection step

   - Select N particles (with replacement) from $\{\tilde{\mathbf{r}}_{0:t}^{(i)}\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain samples $\{\mathbf{r}_{0:t}^{(i)}\}$

   (c) Exact step

   - Update $\alpha_t^{(i)}$ given $\alpha_{t-1}^{(i)}$, $r_t^{(i)}$, $r_{t-1}^{(i)}$, $\mathbf{y}_t$, $\mathbf{u}_{t-1}$, and $\mathbf{u}_t$ with a domain-specific procedure (such as a Kalman Filter)

Figure 4-2: Generic RBPF algorithm. [49]

to factor out the continuous space altogether [6, 21].

This strategy can be understood in the light of the following proposition: [17]

Proposition 2 *The variances of the importance weights and the numerator and the denominator in Equation 3.7 obtained by Rao-Blackwellisation are smaller than those obtained using the generic particle filtering method.*

Furthermore, under weak assumptions, the Rao-Blackwellised estimate converges to the estimated value as $N \rightarrow +\infty$, with a variance smaller than the generic particle filtering method [18]. Therefore, at least based on a fixed number of samples, it is benefitial to sample as small as a subset of the state space as possible. In practice, the run-time performance of the filter will depend on the relative cost of the exact update for $\alpha_t^{(i)}$.

4.2 Tractable substructure in PHA

In the previous section, we have outlined the technique of Rao-Blackwellisation, which reduces the variance of the estimates by factoring out a tractable substructure. Although, in general, inference in hybrid models is NP-hard [45], many hybrid modeling formalisms contain a tractable substructure, which, for HMM-based models, typically takes the form of the continuous state conditioned on sequence of mode assignments [6, 32]. In this Section, we review the structure in switching linear models, in which the posterior over the continuous state is a Gaussian when conditioned on a mode sequence. We then elaborate on the tractable substructure in Probabilistic Hybrid Automata, which exhibit additional challenges, including nonlinearities and autonomous transitions.

4.2.1 Structure in switching linear systems

Switching linear dynamical systems (SLDS), also known as jump Markov linear Gaussian models, are a special form of hybrid models, in which $\mathbf{x}_d$ is finite Markov chain

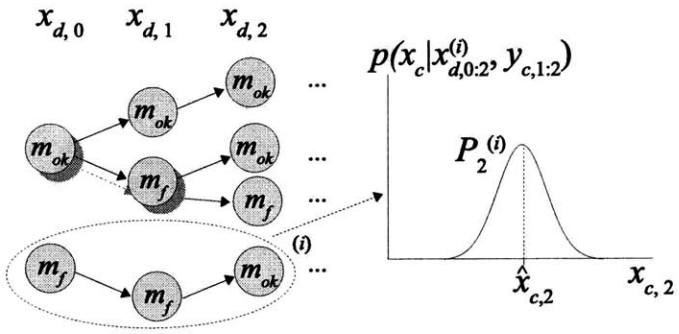

Figure 4-3: Structure in switching linear dynamical systems. Once we fix the mode up to time t, we can estimate the continuous state at time t analytically.

with transition distribution $p(\mathbf{x}_{d,t}\mathbf{x}_{d,t-1})$ and the continuous state evolves as

$$\mathbf{x}_{c,t} = \mathbf{A}(\mathbf{x}_{d,t})\mathbf{x}_{c,t-1} + \mathbf{B}(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1} + \mathbf{v}_x(\mathbf{x}_{d,t}) \tag{4.5}$$

$$\mathbf{y}_t = \mathbf{C}(\mathbf{x}_{d,t})\mathbf{x}_{c,t} + \mathbf{D}(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1} + \mathbf{v}_y(\mathbf{x}_{d,t}), \tag{4.6}$$

where A, B, C, D are mode-dependent system and observation matrices, and $\mathbf{v}_x(\mathbf{x}_{d,t})$ and $\mathbf{v}_y(\mathbf{x}_{d,t})$ are normally-distributed noise variables. Therefore, they can be viewed as a special form of PHA without autonomous transitions, in which the system function $\mathbf{f}$ and the observation function $\mathbf{g}$ (Equations 2.10) are linear.

SLDS models have attractive properties that make them particularly amenable to Rao-Blackwellisation. If we take an arbitrary (but fixed) assignment $d_{0:t} \triangleq d_0, d_1, \ldots, d_t$ to the mode variables $\mathbf{x}_{d,0:t}$, the initial distribution $p(\mathbf{x}_{c,0})$, the system matrices $\mathbf{A}, \mathbf{B}$, the observation matrices $\mathbf{C}, \mathbf{D}$, and the noise models will be fixed for all $t' = 1, \ldots, t$. This means that once we fix the mode variables $\mathbf{x}_{d,0:t} =$, we can construct an analytical estimate of the continuous state of the system up to time t [6] (see Figure 4-3).

From the probabilistic point of view, fixing the mode variables $\mathbf{x}_{d,0:t}$ to $d_{0:t}$ amounts to conditioning on the event that $\mathbf{x}_{d,0:t} = d_{0:t}$ and computing the posterior probability $p(\mathbf{x}_{c,t}|\mathbf{x}_{d,0:t} = d_{0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$. Since the transition and observation functions are linear, the posterior probability $p(\mathbf{x}_{c,t}|\mathbf{x}_{d,0:t} = d_{0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ can be computed *exactly* with a Kalman Filter [7].

4.2.2 Structure in PHA

For a PHA, continuous behavior of the system may change at each time step, in a similar way as it does in SLDS models. However, PHA pose two additional challenges to continuous state estimation: non-linearities and autonomous mode transitions. These two challenges translate to two approximations:

1. When nonlinear functions are used in the transition or an observation function, posterior is typically nonlinear and non-Gaussian. This means, that the continuous state tracking will typically incur error whenver the system is propagated through such equations.

2. When the PHA model has autonomous transitions, the posterior will be biased right after the transition towards the regions of those guards c_j, which have a higher associated transition probability $p_{\tau_j}(\mathbf{x}_{d,t}^{(i)})$.

The first approximation will be accurate to the first degree if an Extended Kalman Filter is used, or to the second degree if the Unscented Kalman Filter is used.

In order to understand the second approximation, consider the example in Figure 4-4. This figure shows the distribution $N(0,1)$ of variable x, when it is first propagated through a constraint $x > 0$ (upper right-hand corner) and then is evolved according to the continuous model $x' = x + \mathcal{N}(0,1)$ (lower right-hand corner). We see that by conditioning the variable on the event $x > 0$, its distribution is slightly skewed to the right and has a smaller variance. Nevertheless, the disitribution is still approximated well by a Gaussian, due to the normal noise added after the distribution is propagated through the constraint.

In our current algorithm, we make no special arrangements in order to account for the bias introduced by autonomnous transitions. It may be possible, however, to compute the true mean and variance of the distribution after it has been propagated through the continuous constraint by numerical methods.

Having described the structure in PHA, we turn to describing an algorithm, which uses this structure to perform Rao-Blackwellised particle filtering for PHA.

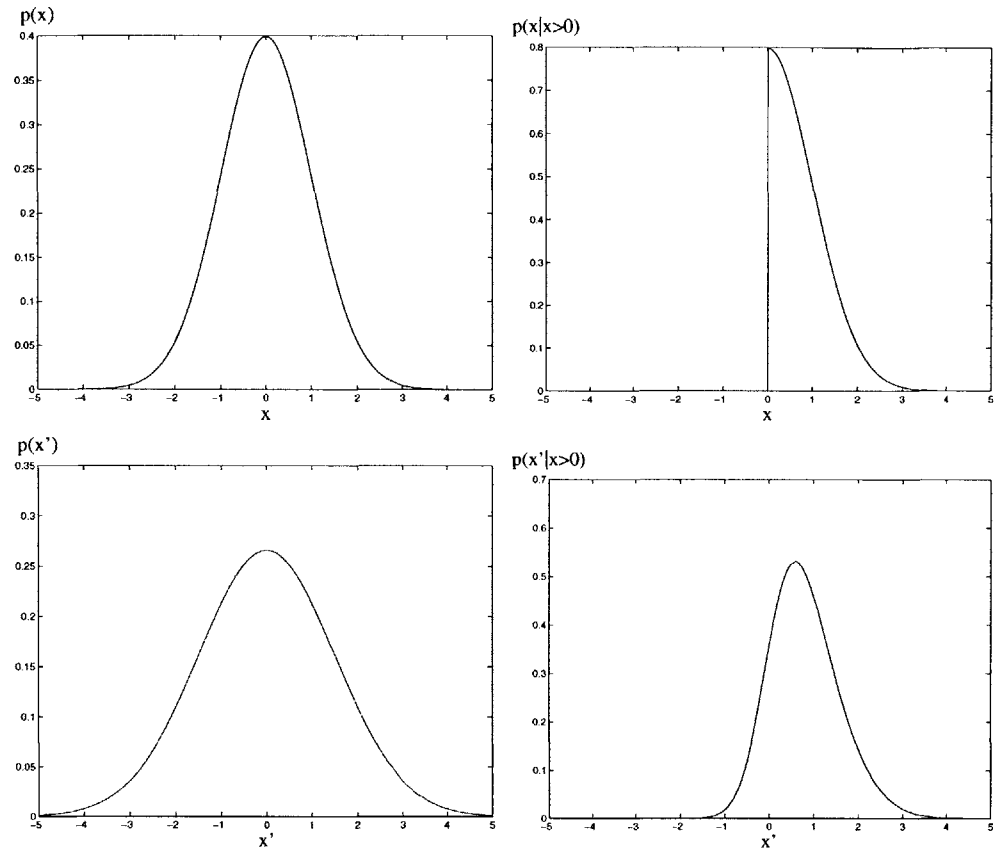

Figure 4-4: The top two graphs show a Gaussian distribution $p(x)$ (left) and a graph of this distribution when it is propagated through a constraint $x > 0$ (right). The bottom left graph shows the distribution when it is propagated through the model $x' = x + \mathcal{N}(0, 1)$ but ignores the constraint, while the bottom right graph shows the true distribution when the constraint is accounted for (obtained by importance sampling with a large number of samples).

4.3 Gaussian Particle Filtering for PHA

4.3.1 Overview of the algorithm

Since we can compute (or approximate) the posterior distribution $p(\mathbf{x}_{c,t}|\mathbf{x}_{d,0:t}, \mathbf{u}_{0:t})$ efficiently in an analytical form, we can apply Rao-Blackwellisation to this problem by taking $\mathbf{r} = \mathbf{x}_d$ and $\mathbf{s} = \mathbf{x}_c$. In other words, we will sample the mode sequences $\mathbf{x}_{d,0:t}^{(i)}$ with a particle filter and, for each sampled sequence $\mathbf{x}_{d,0:t}^{(i)}$, we estimate the continuous state with a Kalman Filter. The result of Kalman Filtering for each sampled sequence $\mathbf{x}_{d,0:t}^{(i)}$ will be the estimated mean $\hat{\mathbf{x}}_{c,t}^{(i)}$ and the error covariance matrix $\mathbf{P}_t^{(i)}$. The samples $\mathbf{x}_{d,0:t}^{(i)}$ will serve as an approximation of the posterior distribution over the mode sequences, $p(\mathbf{x}_{d,0:t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, while each continuous estimate $\langle \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle$ will serve as a Gaussian approximation of the conditional distribution $p(\mathbf{x}_{c,t}|\mathbf{x}_{d,0:t} = \mathbf{x}_{d,0:t}^{(i)}, \mathbf{y}_{0:t}; \mathbf{u}_{0:t}) \triangleq \alpha_t^i$. Since the estimate $\langle \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle$ merely approximates α_t^i, we will not be performing a strict Rao-Blackwellisation; nevertheless, the results will be accurate up to the approximations in the Extended or the Unscented Kalman Filter.

Our algorithm is illustrated in Figure 4-5. Each particle now holds a sample sequence $\mathbf{x}_{d,0:t}^{(i)}$ and the corresponding continuous estimate $\hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)}$. The algorithm starts by taking a fixed number of random samples from the initial distribution over the mode variables $p(\mathbf{x}_{d,0})$ (Step 1). For each sampled mode $\mathbf{x}_{d,0}^{(i)}$, the corresponding continuous distribution $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$ is specified by the PHA model.

The algorithm then proceeds to expand the mode sequences and updates the corresponding continuous estimates (see Figure 4-5, Step 2). In each time step, we first evolve each particle by taking one random sample $\mathbf{x}_{d,t}^{(i)}$ from the proposal distribution $q(\mathbf{x}_{d,t}; \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$. This distribution takes into account the transition model for the PHA in the mode $\mathbf{x}_{d,t-1}^{(i)}$ and can be efficiently computed from the transition model and the continuous estimates, as described in the next section. For each new mode sequence $\mathbf{x}_{d,0:t}^{(i)}$, we compute the importance weight $w_t^{(i)}$. These importance weights take into account the latest observation $\mathbf{y}_t$ and are akin to the observation function $p(\mathbf{y}_t|\mathbf{x}_t)$ in a hidden Markov model. After we compute the importance weights, we

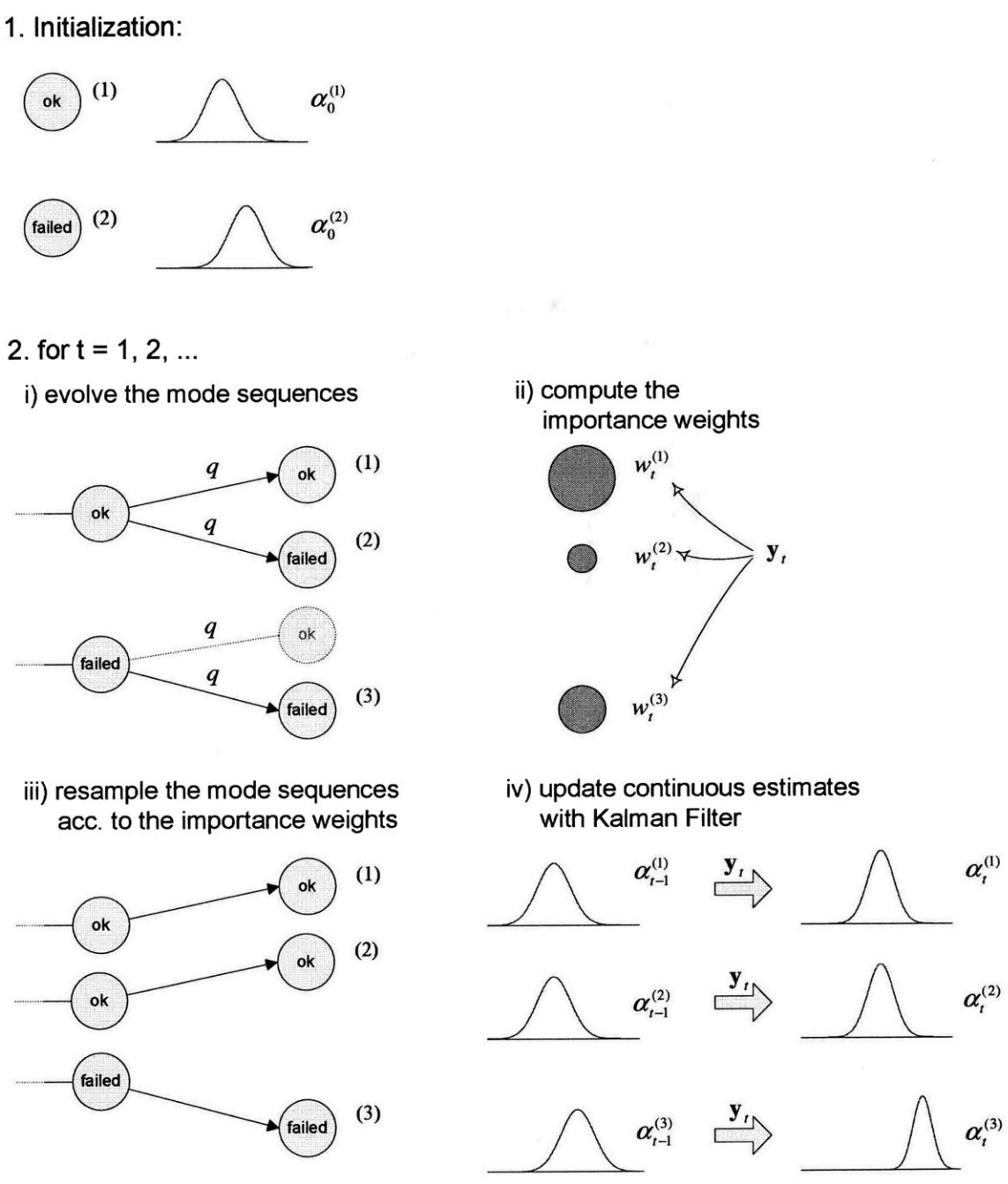

Figure 4-5: Gaussian particle filter for PHA.

resample the trajectories according to their importance weights using one of the selection schemes described in Section 3.2.3, such as residual resampling. This step will, in effect, direct the future expansion of the mode sequences into relevant regions of the state space.

The final step in Figure 4-5 updates the continuous estimate for each new mode sequence $\mathbf{x}_{d,0:t}^{(i)}$. Since in PHA, each mode assignment $\mathbf{d}$ over the variables $\mathbf{x}_d$ is associated with a transition and observation distributions

$$\mathbf{x}_{c,t} = \mathbf{f}(\mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}; \mathbf{d}) + \mathbf{v}_x(\mathbf{d}) \tag{4.7}$$

$$\mathbf{y}_{c,t} = \mathbf{g}(\mathbf{x}_{c,t}, \mathbf{u}_t; \mathbf{d}) + \mathbf{v}_y(\mathbf{d}), \tag{4.8}$$

we update each estimate $\hat{\mathbf{x}}_{c,t-1}^{(i)}, P_{t-1}^{(i)}$ with a Kalman Filter, using the transition function $\mathbf{f}(\mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}; \mathbf{d})$, observation function $\mathbf{g}(\mathbf{x}_{c,t}, \mathbf{u}_t; \mathbf{d})$, and noise variables $\mathbf{v}_x(\mathbf{x}_{d,t}^{(i)})$ and $\mathbf{v}_y(\mathbf{x}_{d,t}^{(i)})$, to obtain a new estimate $\hat{\mathbf{x}}_{c,t}^{(i)}, P_t^{(i)}$.

4.3.2 Proposal distribution

In order to complete the algorithm outlined above, we need to specify the proposal distribution $q(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$, which determines, how the mode sequences are evolved. For simplicity we choose the distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t-1} = \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$. This distribution expresses the probability of the transition from the mode $\mathbf{x}_{d,0:t-1}^{(i)}$ to each mode $\mathbf{d} \in \mathcal{X}_d$ and is similar in its form to the transition distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ in a Markov process. However, it is conditioned on a complete discrete state sequence and all previous observations and control actions, rather than simply on the previous state. This is because $\{\mathbf{x}_{d,t}\}$ alone is *not* an HMM process: due to the autonomous transitions, knowing $\mathbf{x}_{d,t-1}$ alone does not tell us what the distribution of $\mathbf{x}_{d,t}$ is. The distribution of $\mathbf{x}_{d,t}$ is known only when conditioned on the mode *and* the continuous state in the previous time step (see Figure 4-6).

Our key insight is to compute the proposal distribution for each tracked mode sequence $\mathbf{x}_{d,0:t-1}^{(i)}$ using the corresponding continuous estimate $\langle \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle$. Since the estimate $\langle \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle$ captures the posterior distribution of the continuous state con-

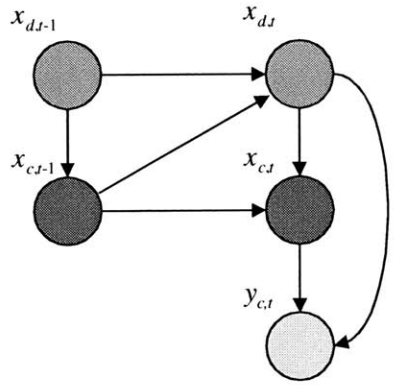

Figure 4-6: Conditional dependencies among the state variables $\mathbf{x}_c, \mathbf{x}_d$ and the output $\mathbf{y}_c$ expressed as a dynamic Bayesian network [14]. The edge from $\mathbf{x}_{c,t-1}$ to $\mathbf{x}_{d,t}$ represents the dependence of $\mathbf{x}_{d,t}$ on $\mathbf{x}_{c,t-1}$, that is, autonomous transitions.

ditioned on the i-th sequence, $p(\mathbf{x}_{c,t-1}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$, we can integrate it out to obtain a transition distribution conditioned on $\mathbf{x}^{(i)}_{d,0:t-1}$, $\mathbf{y}_{0:t-1}$ and $\mathbf{u}_{0:t}$ alone:

$$
\begin{aligned}
& p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) \\
&= \int_{\mathbf{x}_{c,t-1}} p(\mathbf{x}_{d,t}, \mathbf{x}_{c,t-1}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})\ d\mathbf{x}_{c,t-1} \\
&= \int_{\mathbf{x}_{c,t-1}} p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{x}_{c,t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_{c,t-1}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})\ d\mathbf{x}_{c,t-1} \\
&= \int_{\mathbf{x}_{c,t-1}} p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,t-1}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{c,t-1}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})\ d\mathbf{x}_{c,t-1} \quad (4.9)
\end{aligned}
$$

The first equality follows from the total probability theorem. The second equality comes from the independence assumptions made in the model: the distribution of $\mathbf{x}_{d,t}$ is independent of the observations $\mathbf{y}_{1:t-1}$ and mode assignments prior to time $t-1$, given the state at time $t-1$.

Typically when performing Rao-Blackwellisation, the integral in Equation 4.9 is difficult to evaluate efficiently, as noted in [49], since the integral 4.9 often does not have a closed form. For PHA, however, efficient evaluation of this integral is possible.

Recall that the distribution of the discrete evolution of a PHA is specified as a finite set of guards c and the associated transition probabilities p_τ. Each guard specifies a region over the continuous state and automaton's input/output variables, for which the transition distribution p_τ holds. For example, the acrobot model in Figure 2-1 has

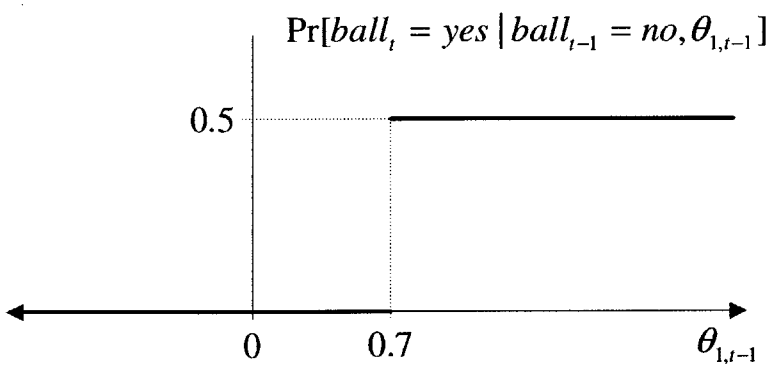

Figure 4-7: Probability of a mode transition `ball=no` to `ball=yes` as a function of $\theta_{1,t-1}$.

two guard conditions for the mode `has-ball=no`: $\theta_1 < 0.7$ and $\theta_1 > 0.7$, with associated transition probabilities back to `no` of 1.0 and 0.5, respectively. Since the transition distributions p_τ are fixed, the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1})$ takes a finite number of values for *varying* $\mathbf{x}_{c,t-1}$.

As an example, consider computing the probability of transitioning from mode $\texttt{ball}_{t-1} = \texttt{no}$ to mode $\texttt{ball}_t = \texttt{yes}$, as a function of $\theta_{1,t-1}$ (Figure 4-7). When $\theta_{1,t-1} > 0.7$, the probability of transitioning from $\texttt{ball}_{t-1} = \texttt{no}$ to $\texttt{ball}_t = \texttt{yes}$ is equal to 0.5. When $\theta_{1,t-1} < 0$, the transition probability is given by the distribution associated with the guard $\theta_1 < 0.7$, and is equal to 0. In general, the mode transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1})$ will be constant over the regions $X_j \subset \mathbb{R}^{n_x}$ that satisfy the corresponding guards, $c(\mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}, \mathbf{y}_{t-1})$. In each region X_j, the distribution will be determined by the corresponding distribution $p_{\tau j}$ from the PHA model.

To see how, this insight aids in the evaluation of the proposal distribution, consider the left term in in the integral in Equation 4.9, $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1})$. Since this term in takes only a finite number of distributions $p_{\tau j}$, we can split the integral domain into the sets X_j that satisfy the constraints c_j and factor out the transition

probability $p_{\tau j}$:

$$\begin{aligned}
&\int_{\mathbf{x}_{c,t-1}} p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})\ d\mathbf{x}_{c,t-1} \\
&\quad = \sum_j \int_{X_j} p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})\ d\mathbf{x}_{c,t-1} \\
&\quad = \sum_j p_{\tau j} \int_{X_j} p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})\ d\mathbf{x}_{c,t-1} \\
&\quad = \sum_j p_{\tau j} \Pr_{\alpha_t^{(i)}}[X_j] \qquad (4.10)
\end{aligned}$$

The second equality holds because, for the region X_j, the conditional distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,t-1}^{(i)}, \mathbf{x}_{c,t-1}, \mathbf{u}_{t-1})$ is fixed and equal to $p_{\tau j}$. Therefore, in each summed term, we multiply the transition distribution $p_{\tau j}$ by the probability of satisfying the guard c_j in the distribution $\alpha_t^{(i)}$.

4.3.3 Evaluating the probability of a transition guard

Given the derivation in the previous section, the remaining challenge in computing the proposal distribution is evaluating the probability of statisfying the guard c_j in the distribution $\alpha_t^{(i)} = p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})$. For simplicity, assume that c_j is only over continuous state variables. The guards of the form $c_d(\mathbf{u}_{d,)} \wedge c_c(\mathbf{x}_{c,)}$, which include both the guard over input variables and over the continuous state space, can be handled by defining $Pr[c_u(\mathbf{u}_{t-1})] \equiv 1$ whenever the input satisfies the guard c_u and $Pr[c_u(\mathbf{u}_{t-1})] \equiv 0$ otherwise.

As suggested in Section 3.2, computing the posterior distribution of $\mathbf{x}_c$ or its characteristics exactly is, in general, intractable. While it would be possible to use a particle filter to estimate the distribution $\alpha_t^{(i)}$, doing so would be prohibitively expensive and would defeat the purpose of applying Rao-Blackwellisation to this problem. Instead, we use Gaussian distributions with the estimated mean $\hat{\mathbf{x}}_{c,t-1}^{(i)}$ and covariance $\mathbf{P}_{t-1}^{(i)}$ in place of the true posterior distribution $\alpha_t^{(i)}$. While this approximation will introduce estimation error in the proposal, it allows us to compute the proposal distribution efficiently, since the problem simplifies to computing an integral over a

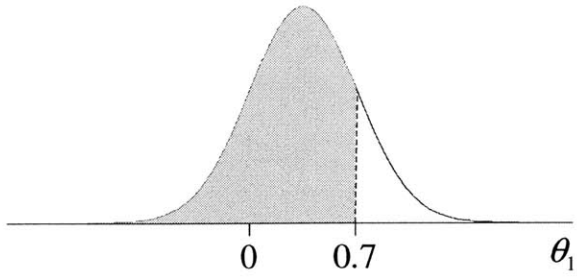

Figure 4-8: Evaluating simple guard conditions.

Gaussian distribution with mean $\hat{\mathbf{x}}_{c,t-1}^{(i)}$ and covariance matrix $\mathbf{P}_{t-1}^{(i)}$:

$$\Pr_{\alpha_t^{(i)}}[X_j] \approx \frac{1}{(2\pi)^{n_c/2}|\mathbf{P}_{t-1}^{(i)}|^{1/2}} \int_{X_j} e^{-\frac{1}{2}(\mathbf{x}_c-\hat{\mathbf{x}}_{c,t-1}^{(i)})^T {\mathbf{P}_{t-1}^{(i)}}^{-1}(\mathbf{x}_c-\hat{\mathbf{x}}_{c,t-1}^{(i)})} \, d\mathbf{x}_c \tag{4.11}$$

This approach was suggested in [32] for single-variate guards of the form $x < c$ and $x > c$, where $x \in \mathbf{x}$ is a continuous state variable and c is a real constant. In this section, we first summarize their procedure and then show, how it generalizes to multi-variate linear conditions.

Interval single-variate guards

When the guards are of the form $x < c$ or $x \leq c$ for some constant c, such as $\theta_1 < 0.7$, the integral in Equation 4.11 simplifies to evaluating the cumulative density function of the normal variable $\mathcal{N}(\mu, \sigma^2)$, where $\mu = (\hat{\mathbf{x}}_{c,t-1}^{(i)})_x$ is the mean of variable x in $\hat{\mathbf{x}}_{c,t-1}^{(i)}$ and $\sigma^2 = (\mathbf{P}_t^{(i)})_x$ is its variance (Figure 4-8):

$$D(c) \triangleq \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{c} e^{-(x-\mu)^2/(2\sigma^2)} dx. \tag{4.12}$$

The cumulative density function $D(c)$ can be evaluated using standard numerical methods, such as trapezoidal approximation or using a table lookup. In order to evaluate the probability of the complementary guards $x > c$ or $x \geq c$, we take the complement of the cumulative density function, $1 - D(c)$.

The above forms of guard conditions can be viewed as a special case of a more general form, in which x falls into an interval $[l; u]$.[1], where l, u are in the extended

[1]Whether the interval is closed or open matters only if x can have a zero variance. It is straight-

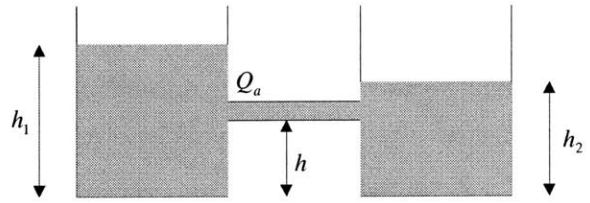

Figure 4-9: A two-tank system.

set of real numbers $\mathbb{R}^+ \triangleq \mathbb{R} \cup \{-\infty, +\infty\}$ that includes positive and negative infinity. In these cases, the probability of satisfying a guard condition can be expressed as the difference of the c.d.f at the endpoints of the interval, $D(u) - D(l)$. Such guards are thus slightly more expressive, while maintaining the same computational complexity.

Rectangular multi-variate guards

Multivariate guards are often needed to represent more complex constraints on transitions. For example, in a two-tank system connected by a pipe at height h (Figure 4-9) [42], the transition between the four flow modes between the two tanks are constrained by the how heights h_1 and h_2 compare to h. In this system, the transition into the `no-flow` mode with $Q_a = 0$ would be conditioned on the guard $(h_1 < h) \wedge (h_2 < h)$.

In general, the rectangular multi-variate guards will take the form $\bigwedge_{i \in I}(x_i \in [l_i; u_i])$, where x_i are distinct continuous state variables and $I \triangleq \{i_1, \ldots, i_n\}$ are their indices. Evaluating the probability of such a multi-variate guard amounts to evaluating the multi-dimensional (hyper)rectangular integral over a Gaussian distribution (see Figure 4-10):

$$\Pr_{\alpha_t^{(i)}}[X_j] \approx \frac{1}{(2\pi)^{n_c/2}|\mathbf{P}_I|^{1/2}} \int_{l_{i_1}}^{u_{i_1}} \int_{l_{i_2}}^{u_{i_2}} \cdots \int_{l_{i_n}}^{u_{i_n}} e^{-\frac{1}{2}(\mathbf{x}_c - \hat{\mathbf{x}}_{c,I})^T \mathbf{P}_I{}^{-1}(\mathbf{x}_c - \hat{\mathbf{x}}_{c,I})} \, d\mathbf{x}_c, \quad (4.13)$$

where $\hat{\mathbf{x}}_{c,I}$ is the mean of guard values, selected from the continuous state estimate $\hat{\mathbf{x}}_{c,t-1}^{(i)}$, and $\mathbf{P}_I$ is the covariance matrix of guard values, selected from the estimate covariance $\mathbf{P}_{t-1}^{(i)}$. Rectangular integrals over Gaussian distributions can be evaluated efficiently using numerical methods, using numerical methods, such as those presented in [38, 24]. As an alternative, one could use Monte Carlo methods to evaluate the

forward to generalize the discussion here to open and half-open intervals.

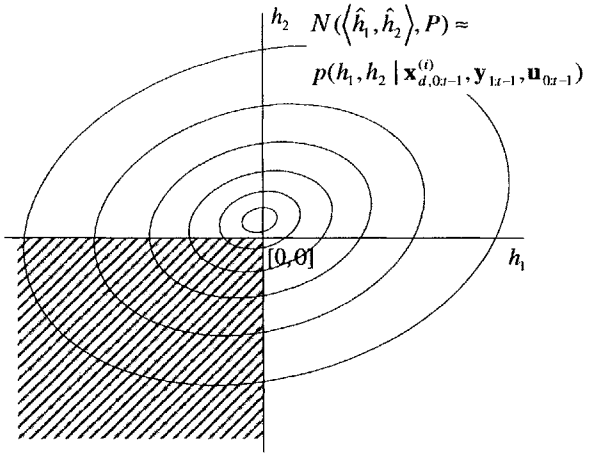

Figure 4-10: Rectangular integral over a Gaussian approximation of the posterior density of h_1 and h_2, $p(h_1, h_2|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})$.

integral 4.13; however, numerical methods tend to perform better.

Linear multi-variate guards

Sometimes, linear combination of continuous variables best represents the transition guard. For example, in the two-tank system in Figure 4-9, the direction of the flow Q_a depends on the heights in the two tanks: if $h_1 > h_2$, the flow will be positive, while if $h_1 < h_2$, the flow will be negative. If we were to have a mode variable in our model that represents the direction of the flow, the transitions for this mode variable would be guarded by the linear guards $h_1 > h_2$ and $h_1 < h_2$, or equivalently, $h_1 - h_2 > 0$ and $h_1 - h_2 < 0$ (see Figure 4-11). Such guards cannot be handled by directly applying the procedure for rectangular constrained described above. One remedy would be to include Q_a in our model, as a derived state variable. However, this would increase the computational complexity of the Kalman Filter. Even worse, introducing the derived state variable would make the covariance matrix singular and prevent efficient implementation of the inversion and matrix square root operations in the Unscented Kalman Filter. Instead, the key idea is to apply a linear transform to the variables and reduce the computation to one of the previous two cases.

For example, consider the guard $h_2 < h_1$ in Figure 4-11. Suppose that the random vector $[h_1\ h_2]^T$ is distributed as $\mathcal{N}(\hat{\mathbf{x}}_{c,I}, \mathbf{P}_I)$, where $\hat{\mathbf{x}}_{c,I}$ and $\mathbf{P}_I$ are the mean and the

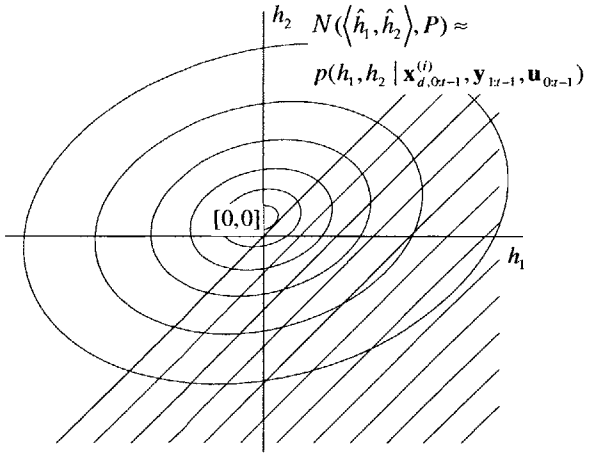

Figure 4-11: Linear guard $h_2 < h_1$ over the Gaussian approximation of the posterior density of h_1 and h_2.

covariance matrix of h_1 and h_2 in the continuous state estimate $\langle \hat{\mathbf{x}}^{(i)}_{c,t-1}, \mathbf{P}^{(i)}_{t-1} \rangle$. Then the random variable $h_2 - h_1 = [-1\ 1][h_1\ h_2]^T$ is normal with mean $[-1\ 1]\hat{\mathbf{x}}_{c,I}$ and variance $[-1\ 1]\mathbf{P}_I[-1\ 1]^T$. Therefore, we can evaluate the probability of the guard by computing an integral over the Gaussian (single-variate) distribution with mean $[-1\ 1]\hat{\mathbf{x}}_{c,I}$ and variance $[-1\ 1]\mathbf{P}_I[-1\ 1]^T$, as it was done in the previous subsections.

In general, suppose that the guard condition c is expressed as a conjunction of clauses $\bigwedge_{i=1}^{n} l_i < \mathbf{a}_i \mathbf{x}_{c,<} u_i$. With two continuous state variables, such conditions correspond to a polygon in the plane that is formed as an intersection of half-planes $l_i < a_{1,i}x_{c1} + a_{2,i}x_{c2}$ and $a_{1,i}x_{c1} + a_{2,i}x_{c2} < u_i$. In higher-dimensional space, these guard conditions correspond to a convex space that is formed as an intersection of hyper-planes $l_i < \mathbf{a}_i \mathbf{x}_c$ and $\mathbf{a}_i \mathbf{x}_c < u_i$.

Let

$$\mathbf{A} \triangleq \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} \tag{4.14}$$

Then $\mathbf{z} \triangleq \mathbf{A}\mathbf{x}_c$ defines a random vector with n elements. Furthermore, the guard $c \triangleq \bigwedge_{i=1}^{n} l_i < \mathbf{a}_i \mathbf{x}_{c,<} u_i$ is equivalent to the guard $\bigwedge_{i=1}^{n} l_i < \mathbf{z}_i < u_i$. Therefore, if we knew the posterior distribution $p(\mathbf{z}_t | \mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})$ of the derived vector $\mathbf{z}$ and

could evaluate the integral

$$\int_{l_{i_1}}^{u_{i_1}} \int_{l_{i_2}}^{u_{i_2}} \cdots \int_{l_{i_n}}^{u_{i_n}} p(\mathbf{z}_t|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})\ d\mathbf{x}_c, \tag{4.15}$$

over the rectangular region $[l_1; u_1] \times [l_2; u_2] \times \cdots \times [l_n; u_n]$, this integral would be the desired probability of the guard c.

In general, the posterior distribution of $\mathbf{z}$ will be as intractable as the posterior distribution $\alpha_{t-1}^{(i)}$ of $\mathbf{x}_c$. Nevertheless, if we approximate $\alpha_{t-1}^{(i)}$ with the continuous estimate as $\mathcal{N}(\hat{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{P}_{t-1}^{(i)})$, the distribution of $\mathbf{z}$ will be Gaussian with a mean $\mathbf{A}\hat{\mathbf{x}}_{c,t-1}^{(i)}$ and a covariance $\mathbf{A}\mathbf{P}_{t-1}^{(i)}\mathbf{A}^T$. Therefore, in order to compute the probability of the guard $\bigwedge_{i=1}^{n} l_i < \mathbf{a}_i\mathbf{x}_{c,<}u_i$, we can compute the mean and covariance of $\mathbf{z}$, and use the rectangular integration methods discussed in the previous subsections.

4.3.4 Importance weights

Given our choice of the proposal distribution, the weights $w_t^{(i)}$ in Equation 4.3 simplify to

$$w_t^{(i)} \triangleq \frac{p(\mathbf{y}_t|\tilde{\mathbf{x}}_{d,0:t}^{(i)}, \mathbf{y}_{0:t-1})p(\tilde{\mathbf{x}}_{d,t}^{(i)}|\tilde{\mathbf{x}}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t-1})}{q(\tilde{\mathbf{x}}_{d,t}^{(i)}; \tilde{\mathbf{x}}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t})} = p(\mathbf{y}_t|\tilde{\mathbf{x}}_{d,0:t}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t}) \tag{4.16}$$

This expression represents the likelihood of the observation $\mathbf{y}_t$, given a complete mode sequence, inputs, and previous observations. PHA, like mode hybrid models, do not directly provide this likelihood and only provide the probability of an observation $\mathbf{y}$, conditioned on the discrete and continuous state (see Figure 4-6). The closest quantity is the posterior distribution of x conditioned on the mode sequence, $p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t-1}, \mathbf{u}_{0:t-1})$. The key idea is to use the system transition and observation model (Equations 4.8), to compute a prediction of $\mathbf{x}_c$ and $\mathbf{y}$. This leads to the well-known Kalman Filter measurement innovation, which has been used extensively in SLDS models. [10] A similar technique applies to nonlinear models, such as PHA.

Observation likelihood in linear switching models

Let $\langle \hat{\mathbf{x}}_{c,t-1}(i), P_{t-1}^{(i)} \rangle$ be the continuous estimate for the i-th mode sequence at time $t-1$. As discussed in Section 4.2.1, in linear systems, this estimate represents the posterior distribution $\alpha_{t-1}^{(i)} \equiv p(\mathbf{x}_{c,t-1}|\mathbf{x}_{d,0:t}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$ *exactly*, that is, $\alpha_{t-1}^{(i)} = \mathcal{N}(\hat{\mathbf{x}}_{c,t}(i), \mathbf{P}_t^{(i)})$. Let

$$\mathbf{x}_{c,t} = \mathbf{A}(\mathbf{x}_{d,t})\mathbf{x}_{c,t-1} + \mathbf{B}(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1} + \mathbf{v}_x(\mathbf{x}_{d,t}) \tag{4.17}$$

$$\mathbf{y}_t = \mathbf{C}(\mathbf{x}_{d,t})\mathbf{x}_{c,t} + \mathbf{D}(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1} + \mathbf{v}_y(\mathbf{x}_{d,t}), \tag{4.18}$$

be the equations for continuous evolution of the model, where A, B, C, D are mode-dependent system and observation matrices, and $\mathbf{v}_x(\mathbf{x}_{d,t})$ and $\mathbf{v}_y(\mathbf{x}_{d,t})$ are normally-distributed noise variables with covariance matrices Q and R, respectively.

Since $\alpha_{t-1}^{(i)}$ is Gaussian, the predicted distribution $\alpha_t^{(i)-} \triangleq p(\mathbf{x}_{c,t}|\mathbf{y}_{1:t-1}, \mathbf{x}_{d,0:t}^{(i)}, \mathbf{u}_{0:t})$ is also Gaussian, because it is a linear comination of Gaussian variables. The mean of $\alpha_t^{(i)-}$ is

$$\begin{aligned}
\mathbb{E}[\alpha_t^{(i)-}] &= \mathbb{E}[A(\mathbf{x}_{d,t}^{(i)})\mathbf{x}_{c,t-1} + B(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1} + \mathbf{v}_{x,t}|\mathbf{y}_{1:t-1}, \mathbf{x}_{d,0:t}^{(i)}, \mathbf{u}_{0:t}] \\
&= A(\mathbf{x}_{d,t})\mathbb{E}[\mathbf{x}_{c,t-1}|\mathbf{y}_{1:t-1}, \mathbf{x}_{d,0:t}^{(i)}, \mathbf{u}_{0:t}] + B(\mathbf{x}_{d,t}^{(i)})\mathbf{u}_{c,t-1} + \mathbb{E}[\mathbf{v}_{x,(}\mathbf{x}_{d,t})] \\
&= A(\mathbf{x}_{d,t})\hat{\mathbf{x}}_{c,t-1}i + B(\mathbf{x}_{d,t})\mathbf{u}_{c,t-1}
\end{aligned} \tag{4.19}$$

and its covariance is

$$\Lambda_x, \mathbf{x}_{d,0:t}^{(i)}, \mathbf{u}_{0:t} = A(\mathbf{x}_{d,t}^{(i)})P_{t-1}^{(i)}A(\mathbf{x}_{d,t}^{(i)}))^T + Q. \tag{4.20}$$

The first equation follows from additive properties for mean, $\mathbb{E}[ax+b] = a\mathbb{E}[x] + \mathbb{E}[b]$, while the second follows from the additive properties for uncorrelated variables.

Similarly, the predited distribution of $\mathbf{y}_{c,t}$ given a mode sequence $\mathbf{x}_{d,0:t}^{(i)}$ and prior

observations $\mathbf{y}_{c,0:t-1}$ is Gaussian, and its first two moments are

$$\begin{aligned}\mathbb{E}[\mathbf{y}_t|\mathbf{x}^{(i)}_{d,0:t}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}] &= C(\mathbf{x}^{(i)}_{d,t})\mathbb{E}[\alpha_t^{(i)-}] + D\mathbf{x}_{d,t}\mathbf{u}_{c,t} + R \\ &\triangleq \mathbf{y}_p. \end{aligned} \tag{4.21}$$

and

$$\Lambda_y = C(\mathbf{x}^{(i)}_{d,t})\Lambda_x C(\mathbf{x}^{(i)}_{d,t})^T + R \triangleq \mathbf{S}_t^{(i)} \tag{4.22}$$

Thus, $\mathbf{y}_t$ is distributed as $\mathcal{N}(\mathbf{y}_p, \mathbf{S})$ when conditioned on mode sequence i and prior observations. This allows us to compute the observation likelihood using the normal p.d.f.:

$$w_t^{(i)} = \frac{1}{(2\pi)^{N/2}|\mathbf{S}_t^{(i)}|^{1/2}} e^{-0.5\mathbf{r}^T(\mathbf{S}_t^{(i)})^{-1}\mathbf{r}}, \tag{4.23}$$

where $\mathbf{r} = \mathbf{y}_t - \mathbf{y}_p$ is the *measurement residual* (innovation). The residual $\mathbf{r}$ and its covariance $\mathbf{S}$ are precisely the values computed in the update step of a Kalman Filter.

Observation likelihood in PHA

Since PHA may, in general, contain nonlinear dynamics and autonomous transitions, the distribution $\alpha_t^{(i)}$ will no longer be Gaussian. Nevertheless, it is possible to approximate the weight with a procedure similar to the one described in the previous section. The key is to use the result of measurement innovation in the Extended or the Unscented Kalman Filter.

For example, in the case of the Extended Kalman Filter, the prediction $\alpha_t^{(i)-}$ is computed by first, propagating the estimated mean through the nonlinear model and predicting the the estimate covariance

$$\hat{\mathbf{x}}^{(i-)}_{c,t} = \mathbf{f}(\hat{\mathbf{x}}^{(i)}_{c,t-1}, \mathbf{u}_{t01}) \tag{4.24}$$

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}|\hat{\mathbf{x}}^{(i)}_{c,t} \tag{4.25}$$

$$\mathbf{P}_t^{(i-)} = \mathbf{A}\mathbf{P}_t^{(i-)}\mathbf{A}^T + \mathbf{Q}. \tag{4.26}$$

This leads to the observation prediction $\mathbf{y}_p$ with covariance $\mathbf{S}_t^{(i)}$:

$$\mathbf{y}_p = \mathbf{g}(\hat{\mathbf{x}}_{c,t}^{(i-)}, \mathbf{u}_t) \tag{4.27}$$

$$\mathbf{C} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}_c} | \hat{\mathbf{x}}_{c,t}^{(i-)} \tag{4.28}$$

$$\mathbf{S}_t^{(i)} = \mathbf{C}\mathbf{P}_t^{(i-)}\mathbf{C}^T + \mathbf{R}. \tag{4.29}$$

Therefore, the likelihood can be approximated again as

$$w_t^{(i)} = \frac{1}{(2\pi)^{N/2}|\mathbf{S}_t^{(i)}|^{1/2}} e^{-0.5\mathbf{r}^T(\mathbf{S}_t^{(i)})^{-1}\mathbf{r}}. \tag{4.30}$$

4.3.5 Putting it all together

The final algorithm is shown in Figure 4-12. Note that the order of the Exact and Selection step of the generic RBPF algorithm in Figure 4-2 has been switched, because the innovation mean and covariance, computed in the Kalman Filter update step, are used to compute the importance weight.

Several straightforward optimizations can be employed to further improve the performance of the algorithm. First, since the algorithm is recursive and only depends on the latest state estimate and the latest mode assignment in a mode sequence, it is sufficient to maintain only the latest mode assignment $\mathbf{x}_{d,t}^{(i)}$, rather than complete mode sequences $\mathbf{x}_{d,0:t}^{(i)}$. Furthermore, the algorithm would compute the transition probability $P_{\mathcal{T}}$ in the Importance Sampling step only once for each unique sample. This can be accomplished by maintaining the number of off-springs N_i generated in the Selection step, and taking N_i random samples from the computed transition probability $P_{\mathcal{T}}$. We implemnted our algorithm in C++, using both of these optimizations.

4.3.6 Sampling from the posterior

One problem with using the transition distribution as a proposal in fault diagnosis domain is that fault transitions typically have a low prior probability, and many particles may be needed to sample and detect the fault. If there is a significant

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_{d,0}^{(i)}$ from the prior distribution $p(\mathbf{x}_{d,0})$
     - initialize the estimate mean $\hat{\mathbf{x}}_{c,0}^{(i)} \leftarrow \mathbb{E}[\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)}]$
     - initialize the estimate covariance $\mathbf{P}_0^{(i)} \leftarrow Cov(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - compute the transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$
     - sample $\tilde{\mathbf{x}}_{d,t}^{(i)} \sim p(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$
     - let $\tilde{\mathbf{x}}_{d,0:t}^{(i)} \leftarrow (\mathbf{x}_{d,0:t-1}^{(i)}, \tilde{\mathbf{x}}_{d,t}^{(i)})$

   (b) Exact step

   - For $i = 1, \ldots, N$
     - perform a KF update: $\tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)}, \mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)} \leftarrow UKF(\hat{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{P}_{t-1}^{(i)}, \tilde{\mathbf{x}}_{d,t}^{(i)})$
     - compute the importance weight:

$$w_t^{(i)} \leftarrow \mathcal{N}(\mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)}) \tag{4.31}$$

   (c) Selection step

   - normalize the importance weights $w_t^{(i)}$
   - Select N particles (with replacement) from $\{\langle \tilde{\mathbf{x}}_{d,0:t}^{(i)}, \tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)} \rangle\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain particles $\{\langle \mathbf{x}_{d,0:t}^{(i)}, \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle\}$

Figure 4-12: Gaussian particle filer for PHA.

amount of information in observations, it may be useful to incorporate it into the proposal, so that modes with high probability in the posterior distribution are also likely in the proposal. It may be possible to use domain-specific heuristic to guide the sampling process [15]; however, such heuristics are difficult to construct and very fragile. One systematic solution is to use the optimal proposal distribution, $q = p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{0:t-1}, \mathbf{y}_{0:t}, \mathbf{u}_{0:t})$ [6, 18]. This distribution was first described by Akashi [6] and is optimal in the sense that it minimizes the variance of importance weights [18]. Unlike the proposal distribution discussed in Section 4.3.2, it is also conditioned on the latest observation. This modification is a double-edged sword: although the performance will improve on a per-sample basis, significantly more computation will need to be performed to compute the proposal distribution. In practice, the trade-off will depend on the amount of information in the observations and on the prior probabilities of the faults.

In order to understand this distribution, let us expand it in terms of the hybrid transition function and the observation likelihood:

$$p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) = \tag{4.32}$$

$$= \frac{p(\mathbf{x}_{d,t}, \mathbf{y}_t|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})}{p(\mathbf{y}_t|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})} \tag{4.33}$$

$$= \frac{p(\mathbf{y}_t|\mathbf{x}_{d,t}, \mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})}{\sum_{d \in \mathcal{X}_d} p(\mathbf{y}_t, \mathbf{x}_{d,t} = d|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})} \tag{4.34}$$

$$= \frac{P^{(i)}_{\mathcal{T},t-1}(\mathbf{x}_{d,t}) p(\mathbf{y}_t|\mathbf{x}_{d,t}, \mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})}{\sum_{d \in \mathcal{X}_d} P^{(i)}_{\mathcal{T},t-1}(d) p(\mathbf{y}_t|d, \mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})} \tag{4.35}$$

Thus, the distribution represents the "increment" in the posterior distribution of the mode sequence from time step $t-1$ to time step t, among all the sequences extending $\mathbf{x}^{(i)}_{d,0:t-1}$. The sum in the denominator ensures the proper normalization of the proposal distribution.

Given this choice of the proposal distribution, the importance weights in Equa-

tion 4.3 simplify to

$$w_t^{(i)} = \frac{p(\mathbf{y}_t|\tilde{\mathbf{x}}_{d,0:t}^{(i)}, \mathbf{y}_{0:t-1}) p(\tilde{\mathbf{x}}_{d,t}^{(i)}|\tilde{\mathbf{x}}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t})}{q(\tilde{\mathbf{x}}_{d,t}^{(i)}; \tilde{\mathbf{x}}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t})} \tag{4.36}$$

$$= \frac{P_{\mathcal{T},t-1}^{(i)}(\tilde{\mathbf{x}}_{d,t}^{(i)}) p(\mathbf{y}_t|\mathbf{x}_{d,0:t}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})}{q(\tilde{\mathbf{x}}_{d,t}^{(i)}; \tilde{\mathbf{x}}_{d,0:t-1}^{(i)}, \mathbf{y}_{0:t})} \tag{4.37}$$

$$= \sum_{d \in \mathcal{X}_d} P_{\mathcal{T},t-1}^{(i)}(d)\ p(\mathbf{y}_t|d, \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) \tag{4.38}$$

Thus, the weight represents the total "increment" in the posterior distribution for the mode sequences that extend $\mathbf{x}_{d,0:t-1}^{(i)}$.

Note that the weights $w_t^{(i)}$ do not depend on the latest mode $\mathbf{x}_{d,t}^{(i)}$. Therefore, it is possible to move the Selection before the Importance sampling step, an idea that was first introduced in [48]. In this manner the number of evolved samples that stem from $\mathbf{x}_{d,t-1}^{(i)}$ will remain unchanged; however, their variety will be increased at no further computational cost.

An important consequence of sampling from the optimal proposal is that the algorithm needs to evaluate the observation likelihood for each successor mode, unless additional approximations are used. This means that its computational complexity per sample increases form $O(T + K)$ when sampling from the prior to $O(T + |\mathcal{X}_d|K)$ when sampling from the posterior, where T is the computational complexity of computing the transition function, $\mathcal{X}_d$ is the number of discrete mode assignments, and K is the cost of the Kalman Filter update. Thus, the trade-off between the two methods will depend on whether there is enough information in the observation likelihood to justify the additional cost of $|\mathcal{X}_d| - 1$ Kalman Filter updates. One heuristic is to take the ratio r between the observation likelihoods $P_{\mathcal{O}}$ in the fault mode and the observation likelihood in nominal modes when a fault occurs. If $r > |\mathcal{X}_d| - 1$, the algorithm will benefit from sampling from the posterior; otherwise, it will not.

4.4 Discussion

In the previous sections, we have derived an efficient Gaussian particle filtering algorithm that can handle single-component systems with autonomous transitions and nonlinear dynamics. In the following two subsections, we relate our algorithm to the prior art in hybrid model-based diagnosis and particle filtering.

4.4.1 Comparison with prior approaches in hybrid model-based diagnosis

Several algorithms have addressed the problem of the exponential growth of the Gaussian mixtures. One class of solutions are multiple-model estimation schemes, which maintain a pre-determined number of mode sequences. These include the generalized pseudo-Bayesian algorithm (GPB) [4], the Detection/Estimation Algorithm (DEA) [56], the popular Interacting Multiple Model (IMM) algorithm [10], and residual correlation Kalman filter bank [29]. All of these techniques have a fixed, deterministic strategy for pruning the mode sequences.

More recently, Lerner et al. [44] proposed a k-best filtering solution for switching linear dynamical models. In addition to pruning, their algorithm implements several techniques not present in our algorithm, including collapsing of the mode sequences, smoothing, and weak decomposition. Lerner extended this approach later in [43] to the setting of hybrid dynamic Bayesian networks with SoftMax transitions, using numerical integration techniques instead of the Kalman Filter. Similarly to ours, their algorithm provides an any-time solution to the hybrid state estimation problem.

Hofbaur and Williams [32] introduced autonomous transitions to the models in the context of Concurrent Probabilistic Hybrid Automata. They introduced an any-time k-best filtering algorithm for concurrent systems. Their algorithm extracts a leading set of sequences in the order of their priors using a combination of branching and A* algorithm that exploits preferential independence and guarantees to find the next set of k leading sequences at each time step.

4.4.2 Comparison with prior particle filtering approaches

Several papers [8, 40] have proposed to use the bootstrap filter to perform state estimation in hybrid models. Dearden [15] demonstrated the application of this method in the rover fault diagnosis domain.

An early application of Rao-Blackwellisation method to reduce the variance of sampling in SLDS models was introduced by Akashi and Kumamoto [6]. Their algorithm, named Random Sampling Algorithm (RSA), sampled the sequences of mode assignments using the distribution $p(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$. Doucet [17, 21] introduced the Selection Step, which is crucial for the convergence of sequential Monte Carlo methods and framed the problem in the general particle filtering framework. In addition, he proved several properties on the convergence and variance reduction of Rao-Blackwellisation schemes. Doucet et al. [19] further extended the work of [17] and described an algorithm for fixed-lag smoothing with MCMC steps. Finally, Morales-Menndez et. al [48] introduced the look-ahead procedure, described in Section 4.3.6. This increases the variety of the particles and, in general, improves the performance of the particle filter. All of these techniques were designed for linear switching models without autonomous transitions.

Hutter and Dearden [36] combined the look-ahead Rao-Blackwellised particle filter with an Unscented Kalman Filter, in order to improve the accuracy of the continuous estimates. In our work [23], we have introduced autonomous transitions and drew parallels to prior approaches in hybrid model-based reasoning.

Two complementary approaches for improving the performance of particle filters were proposed by Thrun [55] and Verma [58]. The first one, the risk sensitive particle filter, incorporates a model of cost into the sampling process. The cost is implemented automatically using an MDP value function tracking. The second approach improves the performance of particle filtering by automatically choosing an appropriate level of abstraction in a multiple-resolution hybrid model. Maintaining samples at a lower resolution prevents hypotheses from being eliminated due to a lack of samples.

Chapter 5

Gaussian Particle Filtering for CPHA

In the previous chapter, we described a particle filtering algorithm for PHA models. In practice, a model will be composed of several concurrently operating automata that represent individual components of the underlying system. In this manner, the design of the models can be split on a component-by-component basis, thus enhancing the reusabiliy of the models and reducing modeling costs.

In this chapter we extend our Gaussian particle filter to handle Concurrent Probabilistic Hybrid Automata (CPHA), a modeling formalism that defines the overall hybrid model as a set of PHA, connected through continuous input and output variables, see Chapter 2 or [32, 35] for a description. In CPHA, components transition independently, conditioned on the current discrete and continuous state. Therefore, it is possible to compute the transition probabilities $P_{\mathcal{T}}^{(i)}$ for each tracked mode sequence component-wise [32]. This property is exploited by our algorithm in the importance sampling step, whereby the samples are evolved according to the transition distribution Pt on a component-by-component basis. In order to adapt the second version of the algorithm, which samples from the posterior, we evaluate the observation function for mode transitions independently. This results in an improved proposal q that incorporates some information in the latest observations, but does not need to evaluate the observation likelihood $P_{\mathcal{O}}$ for an excessive number of successor modes.

5.1 Sampling from the prior

Recall that the algorithm in Section 4.3 sampled the mode sequences according to the proposal distribution $q(\mathbf{x}_{d,}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) = p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) \triangleq P^{(i)}_{\mathcal{T},t}$. This represents the probability of being in the mode $\mathbf{x}_{d,t}$, conditioned on the previous sequence of modes $\mathbf{x}^{(i)}_{d,0:t-1}$ and observations $\mathbf{y}_{1:t-1}$ leading to that mode. Given this choice of the proposal, the importance weights simplify to

$$w^{(i)}_t = p(\mathbf{y}_t|\mathbf{x}_{d,0:t}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) \triangleq P^{(i)}_{\mathcal{O},t}. \tag{5.1}$$

When sampling mode sequences in CPHA, we use the same proposal distribution. The only difference is that now, instead of computing the transition probability for every value in the domain $\mathcal{X}_d$ of the discrete variables $\mathbf{x}_d$, we evaluate it only for the individual component's discrete domain $\mathcal{X}_{d\ k}$, and obtain the joint transition distribution $p(\mathbf{x}_{d,t}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$ as a product of component transition distributions $p(\mathbf{x}_{d\,k,t}|\mathbf{x}_{d\,k,0:t-1}, y1:t-1, \mathbf{u}_{0:t})$, for all components k in the model.

To illustrate this process, consider the discrete transition model for the acrobot, shown in Figure 5-1. In order to compute the transition probability from the mode $\mathbf{x}^{(i)}_{d,t-1} = \langle \texttt{actuator=ok}, \texttt{has-ball=no} \rangle$ to the mode $\mathbf{x}^{(i)}_{d,t} = \langle \texttt{actuator=ok}, \texttt{has-ball=yes} \rangle$, we multiply the component probailities

$$\begin{aligned} p_1 &= \Pr(\texttt{actuator=ok}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1}) \\ &= \Pr(\texttt{actuator=ok}|\texttt{actuator}_{t-1} = \texttt{ok}, \mathbf{x}^{(i)}_{d,0:t-2}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1}) \end{aligned} \tag{5.2}$$

and

$$\begin{aligned} p_2 &= \Pr(\texttt{has-ball=yes}|\mathbf{x}^{(i)}_{d,0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1}) \\ &= \Pr(\texttt{has-ball=yes}|\texttt{has-ball}_{t-1} = \texttt{no}, \mathbf{x}^{(i)}_{d,0:t-2}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1}). \end{aligned} \tag{5.3}$$

Since the actuator transition distribution is not conditioned on the continuous state, its evolution satisfies the Markovian property, and p_1 simplifies to $p(\texttt{actuator=no}$ |

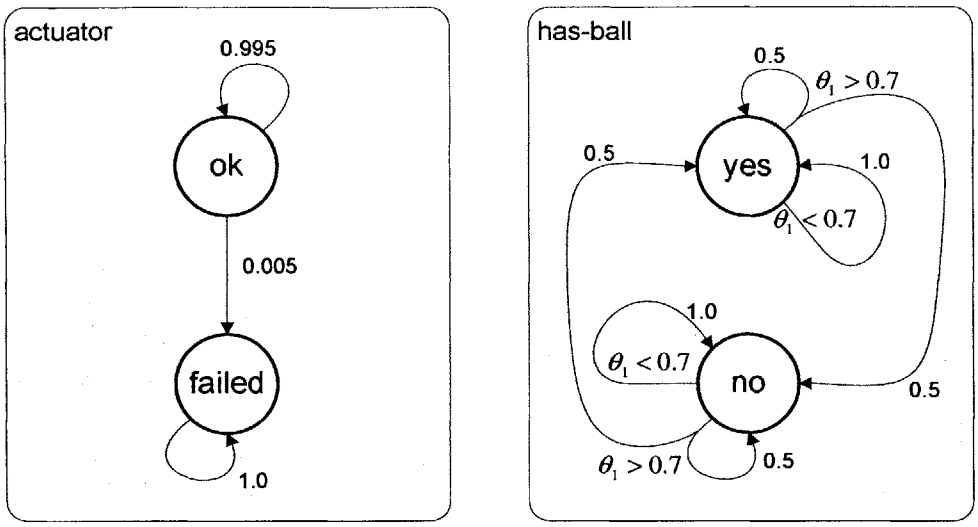

Figure 5-1: The discrete transition model for the acrobatic robot. Due to the independence assumptions made in the model, the joint probability distribution for the two components (`actuator`, `has-ball`) is obtained as a product of the two component distributions, when conditioned on the continuous state.

$\texttt{actuator}_{t-1} = \texttt{ok}$), which can be read off directly from the model. The transitions for the `has-ball` component, on the other hand, do depend on the continuous state. Hence, the transition probability for this component is computed from the continuous estimate $\langle \hat{\mathbf{x}}_{ct,i}^{(.)} P_t^{(i)} \rangle$, as described in Sections 4.3.2 and 4.3.3.

Figure 5-2 shows the pseudocode for the resulting algorithm. The algorithm is based on the algorithm presented in Section 4.3, except that in the importance sampling step, we compute the transition distribution and evolve the sampled mode sequences on a component-by-component basis.

5.2 Sampling from partial posterior

As discussed in Section 4.3.6, the performance of a particle filter can be increased by incorporating the latest observations into the proposal distribution q. In the case of the Gaussian particle filter, these observations can be incorporated by evaluating the observation function at each possible successor mode [6, 48], and sampling according to the distribution $q(\mathbf{x}_{d,t}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) = P_{T,t}^{(i)} P_{\mathcal{O},t}^{(i)}$.

While this approach may work in a single-component system that has only a

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_{d,0}^{(i)}$ from the prior distribution $p(\mathbf{x}_{d,0})$
     - initialize the estimate mean $\hat{\mathbf{x}}_{c,0}^{(i)} \leftarrow \mathbb{E}[\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)}]$
     - initialize the estimate covariance $\mathbf{P}_0^{(i)} \leftarrow Cov(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - For each component k
       * compute the transition distribution $p(\mathbf{x}_{d\,k,t}|\mathbf{x}_{d\,k,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$
       * sample $\tilde{\mathbf{x}}_{d,t\ k}^{(i)} \sim p(\mathbf{x}_{d\,k,t}|\mathbf{x}_{d\,k,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t})$
     - let $\tilde{\mathbf{x}}_{d,0:t}^{(i)} \leftarrow (\mathbf{x}_{d,0:t-1}^{(i)}, \langle \tilde{\mathbf{x}}_{d,t\ 1}^{(i)}, \ldots, \tilde{\mathbf{x}}_{d,t\ n_c}^{(i)} \rangle)$

   (b) Exact step

   - For $i = 1, \ldots, N$
     - perform a KF update: $\tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)}, \mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)} \leftarrow UKF(\hat{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{P}_{t-1}^{(i)}, \tilde{\mathbf{x}}_{d,t}^{(i)})$
     - compute the importance weight:

$$w_t^{(i)} \leftarrow \mathcal{N}(\mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)}) \tag{5.4}$$

   (c) Selection step

   - normalize the importance weights $w_t^{(i)}$
   - Select N particles (with replacement) from $\{\langle \tilde{\mathbf{x}}_{d,0:t}^{(i)}, \tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)} \rangle\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain particles $\{\langle \mathbf{x}_{d,0:t}^{(i)}, \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle\}$

Figure 5-2: Gaussian particle filer for CPHA.

few modes, its performance, as a function of the execution time, will degrade as the number of modes increases. The reason for this degradation is that computing the observation likelihood for each successor mode quickly becomes a burden – with 500,000 modes in the BIO-Plex model [34], the algorithm would have to perform 500,000 Kalman Filter updates *for each sample*. With one Kalman Filter update taking as much as 1ms for a four-variable continuous model (see Section 6.2), this approach would take at least ten hours to perform one iteration, when taking 70 samples for the model described in [34].

Instead, we describe an algorithm that incorporates the latest observations into the sampling process, but does not enumerate all the successor modes. The key idea is to compute the observation likelihood, $P_{\mathcal{O}}$, for each individual component transition, and combine these into the proposal distribution, q. After we sample the new mode, we compute the observation likelihood for the newly generated sample and reflect the discrepancy between the proposal distribution and the true posterior with the importance weights. Thus, the algorithm will still have guaranteed convergence properties as $N \to +\infty$. Although more computation will need to be performed in each time step, fewer samples will be needed to attain the same level of accuracy.

The pseudocode for the algorithm is shown in Figure 5-3. In the importance sampling step, the algorithm computes the observation likelihood for each component k and each mode m in that component. The transitions are treated independently: While we transition the component k into mode m and compute the observation likelihood new the newly evolved sequence, all of the other components remain in the same mode. Together with the transition distribution $P^{(i)}_{T,t\ k}$, this observation likelihood determines the probability of transitioning to mode m of the proposal distribution $q^{(i)}_{t\ k}$, for the component k. This proposal distribution is used to evolve the mode variables in the k-th component, in order to obtain the new sample $\tilde{\mathbf{x}}^{(i)}_{d,t\ k}$.

In the Exact step, the algorithm updates the continuous state, using the newly sampled mode, and computes the importance weights for the newly obtained samples. Recall from Section 3.2.2 that importance weights need to satisfy the following

1. Initialization

   - For $i = 1, \ldots, N$
     - draw a random sample $\mathbf{x}_{d,0}^{(i)}$ from the prior distribution $p(\mathbf{x}_{d,0})$
     - initialize the estimate mean $\hat{\mathbf{x}}_{c,0}^{(i)} \leftarrow \mathbb{E}[\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)}]$
     - initialize the estimate covariance $\mathbf{P}_0^{(i)} \leftarrow Cov(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$

2. For $t = 1, 2, \ldots$

   (a) Importance sampling step

   - For $i = 1, \ldots, N$
     - For each component k
       * compute the transition distribution:

$$P_{\mathcal{T},t\ k}^{(i)} \leftarrow p(\mathbf{x}_{d\,k,t}|\mathbf{x}_{d\,k,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) \tag{5.5}$$

       * For each mode m in component k
         · perform a KF update: $\mathbf{r}, \mathbf{S} \leftarrow UKF(\hat{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{P}_{t-1}^{(i)}, \mathbf{x}_{d,t-1}^{(i)}, m)$
         · compute the observation likelihood $P_{\mathcal{O}} \leftarrow \mathcal{N}(\mathbf{r}, \mathbf{S}_t^{(i)})$
         · let $q_{t\ k}^{(i)}(m) \leftarrow P_{\mathcal{T},t\ k}^{(i)} P_{\mathcal{O}}$
       * normalize $q(\mathbf{x}_{d\,k,t})$
       * sample $\tilde{\mathbf{x}}_{d,t\ k}^{(i)} \sim q(\mathbf{x}_{d\,k,t})$
     - let $\tilde{\mathbf{x}}_{d,0:t}^{(i)} \leftarrow (\mathbf{x}_{d,0:t-1}^{(i)}, \langle \tilde{\mathbf{x}}_{d,t\ 1}^{(i)}, \ldots, \tilde{\mathbf{x}}_{d,t\ n_c}^{(i)} \rangle)$

   (b) Exact step

   - For $i = 1, \ldots, N$
     - perform a KF update: $\tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)}, \mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)} \leftarrow UKF(\hat{\mathbf{x}}_{c,t-1}^{(i)}, \mathbf{P}_{t-1}^{(i)}, \tilde{\mathbf{x}}_{d,t}^{(i)})$
     - compute the importance weight:

$$w_t^{(i)} \leftarrow \frac{\mathcal{N}(\mathbf{r}_t^{(i)}, \mathbf{S}_t^{(i)}) \prod_k P_{\mathcal{T},t\ k}^{(i)}}{\prod_k q_{t\ k}^{(i)}} \tag{5.6}$$

   (c) Selection step

   - normalize the importance weights $w_t^{(i)}$
   - Select N particles (with replacement) from $\{\langle \tilde{\mathbf{x}}_{d,0:t}^{(i)}, \tilde{\mathbf{x}}_{c,t}^{(i)}, \tilde{\mathbf{P}}_t^{(i)} \rangle\}$ according to the normalized weights $\{\tilde{w}_t^{(i)}\}$ to obtain particles $\{\langle \mathbf{x}_{d,0:t}^{(i)}, \hat{\mathbf{x}}_{c,t}^{(i)}, \mathbf{P}_t^{(i)} \rangle\}$

Figure 5-3: Gaussian particle filer for CPHA with improved proposal.

equality:

$$w_t^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_{d,0:t}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t}) p(\mathbf{x}_{d,t}^{(i)}|\mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t-1}, \mathbf{u}_{0:t-1})}{q(\mathbf{x}_{d,t}^{(i)}; \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})}. \tag{5.7}$$

In this case, since the samples $\mathbf{x}_{d,t\ k}^{(i)}$ are evolved independently according to the proposal distribution $q_{t\ k}^{(i)}$ for each component k, the proposal distribution for the *vector* $\tilde{\mathbf{x}}_{d,t}^{(i)}$ becomes

$$q(\mathbf{x}_{d,t}^{(i)}; \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) = \prod_k q_{t\ k}^{(i)}. \tag{5.8}$$

Similarly, the prior transition distribution $P_t^{(i)}$ for the sample $\tilde{\mathbf{x}}_{d,t}^{(i)}$ becomes

$$P_t^{(i)} = \prod_k P_{\mathcal{T},t\ k}^{(i)}. \tag{5.9}$$

Therefore, the weights $w_t^{(i)}$ in Equation 5.7 reduce to

$$w_t^{(i)} = \frac{\mathcal{N}(\mathbf{r}, \mathbf{S}_t^{(i)}) \prod_k P_{\mathcal{T},t\ k}^{(i)}}{\prod_k q_{t\ k}^{(i)}} \tag{5.10}$$

.

The algorithm will work best when the effects of mode transitions are observed independently, or nearly independently, among the component mode variables. This occurs, for example, when the transitions occur in independent or weakly dependent components. In these cases, the observation likelihood in one component is independent of, or weakly-dependent on, the observation likelihoods in the other components. The proposal distribution q is then near the optimal proposal $q(\mathbf{x}_{d,t}^{(i)}; \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t}) = p(\mathbf{x}_{d,t}^{(i)}; \mathbf{x}_{d,0:t-1}^{(i)}, \mathbf{y}_{1:t}, \mathbf{u}_{0:t})$.

We have implemented both of these algorithms in C++. In the next Chapter, we demonstrate them and compare their performance against the k-best filering algorithm [32].

Chapter 6

Experimental Results

In the previous chapters, we described a Gaussian particle filtering algorithm for probabilistic hybrid models that handles the full expressivity of CPHA, including non-linearities, autonomous mode transitions, and concurrency. Through the technique of Rao-Blackwellised particle filtering [6, 21], our algorithm significantly reduces the sampled space. We have shown how the algorithm relates to the prior art in probabilistic hybrid diagnosis, including multiple-model estimation schemes [10], Gaussian k-best filtering [32, 44], and variable resolution particle filtering [58]. What is left open is how Gaussian particle filtering performs in relation to these methods in various domains. More generally, the questions that we are trying to address is how accurate the Gaussian representation is for tracking nonlinear systems, and how well sampling performs in relation to best-first enumeration algorithms. While significant progress has been made in both particle filtering and k-best filtering for hybrid systems, very little attention has been given so far to comparing their relative performance.

In this chapter, we consider the acrobatic robot example introduced in Chapter 2, see Figures 2-1 and 2-2. The discrete state of the hybrid model for this example consists of two variables, representing whether or not the robot holds a ball (variable `has-ball`) and whether or not its actuator has broken (variable `actuator`). The system's dynamics is represented by four continuous variables: θ_1, the angle that the robot holds with the horizontal plane, θ_2, the angle between the robot's torose and its legs, and the corresponding angular velocities ω_1, ω_2. The goal is to filter out the

robot's hybrid state from a sequence of noisy observations of θ_2.

While this example is small, it demonstrates interesting challenges for both tracking and hybrid state estimation. The dynamic model for two-link systems like the acrobatic robot is highly nonlinear. For example, the angular acceleration $\ddot{\theta}_1$ can be expressed as $\ddot{\theta}_1(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) =$

$$\frac{D_{12}T + \dot{\theta}_1^2(-D_{211}D_{12}) + \dot{\theta}_2^2(D_{122}D_{22}) + 2\dot{\theta}_1\dot{\theta}_2(D_{112}D_{22} - D_{212}D_{12}) + D_1D_{22} - D_2D_{12}}{D_{12}^2 - D_{11}D_{22}} \tag{6.1}$$

where the coefficients D involve trigonometric functions of θ_1 and θ_2 and mode-dependent parameters parameters of the robot's body, and T is the torque exerted by the actuator. [51] Furthermore, with four continuous state variables, as demonstrated below, the model appears to be already too large to be handled by particle filters in realtime, and even with 100,000 samples, both the Unscented Particle Filter (UPF) [57] and the Bootstrap filter [26]. Finally, the symptoms exhibited by mode changes are very subtle. Given our choice of the observation noise of $\sigma = 0.1\text{rad} \approx 5\text{deg}$, it takes at least ten time steps to differentiate between the `has-ball=yes` and the `has-ball=no` modes.

6.1 Scenarios

We considered the following three scenarios for tracking and hybrid estimation (see Figure 6-1). The robot was driven by input torque that made it swing back and forth. In the first scenario, the robot remains in the nominal mode `has-ball=no`, `actuator=ok` for the duration of the experiment. In the second one, the robot captures a ball at time $t = 1.3\text{s}$ and keeps it for the rest of the experiment. Capturing a ball increases its weight m_2 at the end of the legs and changes the resulting trajectory, as shown in Figure 6-1. In the final, third scenario, the robot's actuator breaks at $t = 1.8$. This event causes the robot to stop exerting any torque ($T = 0$), and alters its trajectory, as shown in Figure 6-1. In all of our experiments, the time was sampled at 100 Hz, and the continuous state was modeled to evolve with additive

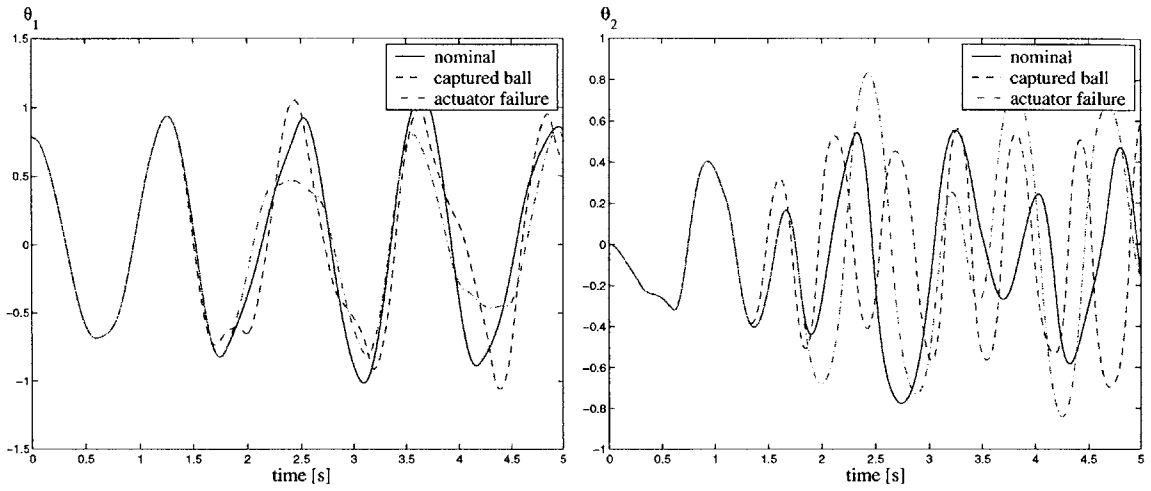

Figure 6-1: The evolution of θ_1 (left) and θ_2 (right) for the three scenarios.

white Gaussian noise with $\sigma_x = 0.01$rad.

As shown in Figure 6-1, the continuous state trajectories for these different scenarios are sufficiently different to be detected on a long-term basis, but are sufficiently subtle not to be observable from a single observation. Thus, the problem is well-suited for hybrid estimation.

6.2 Accuracy of the Gaussian representation

In the first set of experiments, we evaluated the accuracy of simple continuous state tracking when the discrete state was fully observable, in order to verify that our Gaussian representation is valid. We compared the performance of the unscented Kalman filter (UKF) [39] to the ground truth and to the unscented particle filter (UPF) [57], which has been shown to greatly outperform Bootstrap-based importance sampling schemes. The most commonly cited advantage of particle filters is their ability to cope with general nonlinear, non-Gaussian models. [57, 19]. Our experiments show that, even for a highly-nonlinear system like the acrobatic robot, the Gaussian estimates obtained with an UKF are sufficiently accurate for the given task. This result, along with the fact that an UKF is vastly more efficient than an UPF, justifies the use of Gaussian filtering in this domain.

Continuous state tracking

In order to evaluate the performance of continuous state tracking, we assumed that the mode is observable at each time step and temporarily removed the autonomous transitions from the model. The former assumption implies that we do not need to track several mode sequences, while the latter removes the bias from the estimate and allows a meaningful comparison in a system where the true posterior distribution is difficult to obtain.

Figure 6-2 shows the ground truth and continuous state estimates for θ_1 and θ_2, computed with an unscented Kalman filter and an unscented particle filter. The particle filter was using 100,000 samples in each time step and implemented systematic resampling [11] with MCMC moves, in an attempt to obtain an accurate approximation of the true posterior distribution. We see that while the UPF provides an accurate estimate of the continuous state at the beginning of the sequence, it suffers from particle depletion at $t = 3$ seconds, exhibited by a dense distribution of the particles that are clearly off from the true posterior, and eventually diverges from the posterior at $t = 4.2$s. The estimate obtained with the UKF, on the other hand, behaves consistently, even though it has higher variance and definite bias at the peaks of the motion ($t = 1.3$s, $t = 1.8$s, $t = 3.1$s, $t = 3.6$s, and $t = 4.2$s).

In order to obtain a better understanding of how well the Gaussian estimate represents the *joint* posterior distribution over the continuous state variables, we compared it to a close approximation of the posterior, obtained with a large number of samples ($N = 1,000,000$). Figure 6-3 shows a few representative samples from the posterior distributions of $\langle \theta_1, \theta_2 \rangle$ and $\langle \omega_1, \omega_2 \rangle$ at $t = 0.6$s. We see that while the covariance of the UKF estimate is much larger than the covariance of the true posterior distribution, the variables are at least properly correlated.

The table below shows the state estimation (mean square) error of continuous tracking using an UKF and an UPF in the three scenarios considered. The times in parenthesis show the average running time of the algorithm in each time step on a computer with a Pentium 4 processor. The algorithms were implemented within the

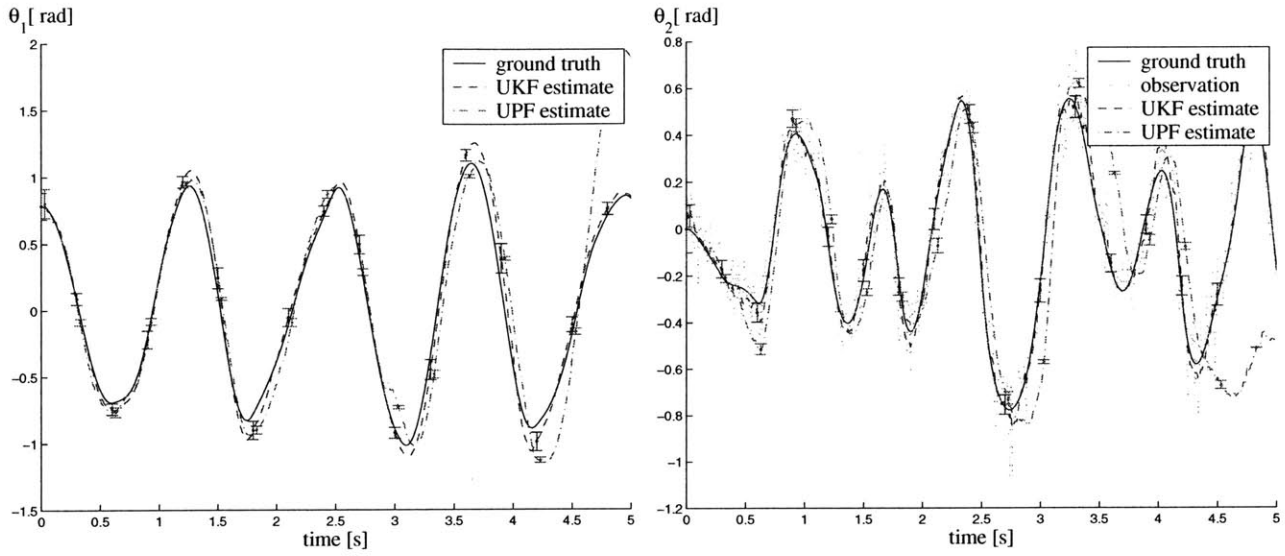

Figure 6-2: Ground truth and state estimates for θ_1 (above) and θ_2 (below). The standard deviations for the estimates were offset by 0.03 in order to clarify the figure.

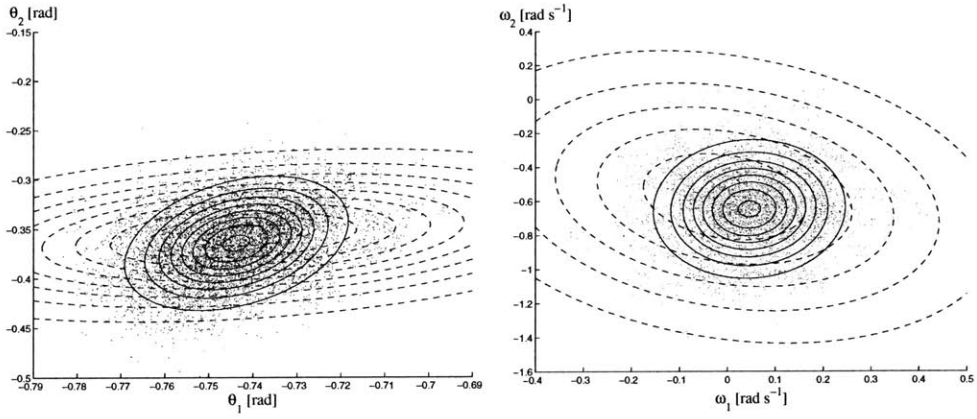

Figure 6-3: Estimated joint distribution over $\langle \theta_1, \theta_2 \rangle$ (left) and $\langle \omega_1, \omega_2 \rangle$ (right) at $t = 0.6$s. The solid lines represent the contours of equiprobability of the Gaussian distribution computed from the samples, while the dashed lines represent the contours of equiprobability for the Gaussian distribution obtained from the unscented Kalman filter.

Bayes++ Bayesian filtering framework. [54]

Filter	Nominal	Ball	Failure
UKF	0.562 (1.435ms)	0.497 (1.438ms)	0.700 (1.436ms)
UPF (10,000 samples)			

6.3 Hybrid estimation

Having evaluated the performance of the continuous tracking, we turn to evaluating our Gaussian particle filtering algorithm presented in Chapter 4 and to comparing its performance to the k-best filter [32].

6.3.1 Single executions

In order to gain insight into the operation of the Gaussian particle filter, we examined its performance on one sequence of observations for each of the three scenarios.

Figure 6-4, shows the maximum a posteriori (MAP) estimate of the filter for the nominal scenario, when the rover was swinging throughout its execution not holding a ball and without experiencing a failure. We see that the algorithm correctly estimates the mode `has-ball=no`, `actuator=ok` of the system (mode value 0 on the graph), except between $t = 4.3$ and $t = 4.8$ seconds, when (presumably) the noise in the observation leads to the wrong diagnosis. The regular spikes at $t \approx 0$, $t \approx 1.3$, $t \approx 2.5$, $t \approx 3.6$ and $t \approx 5$ correspond to the times when θ_1 reaches its maximum (see Figure 6-2). During these times, the autonomous transitions between `has-ball=no` and `has-ball=ues` guarded by the condition $t_1 > 0.7$ are enabled with high probability, and, according to our model, the probability of having the ball is approximately 0.5. This fact is correctly reflected by a decreased confidence in the correct diagnosis (Figure 6-4 right). The exact mode estimate will depend on the values of the received observations and on the outcomes of the probabilistic choices made by the filter.

Figure 6-5 shows the results for the scenario when the robot captures a ball at $t = 1.3$s. This result is very interesting; it shows that the Gaussian particle filter

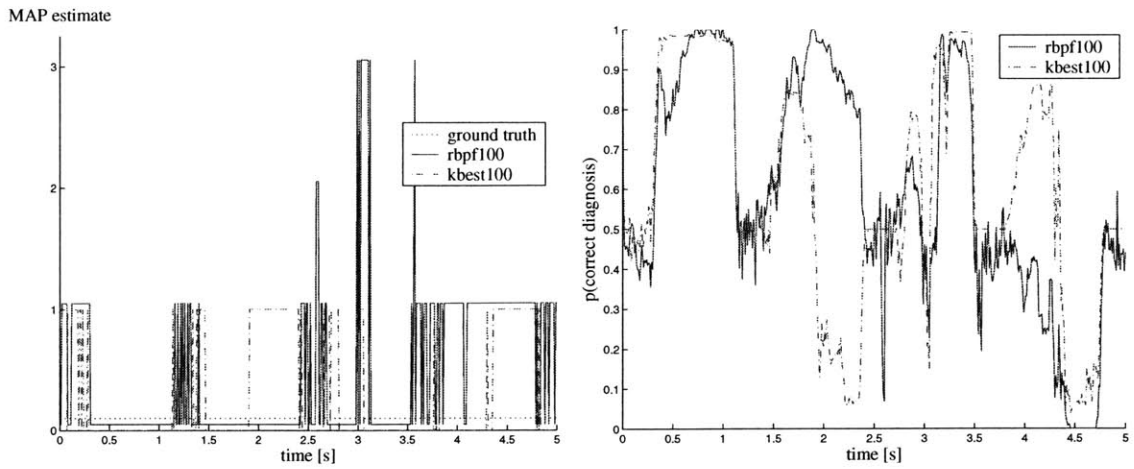

Figure 6-4: A single run for the nominal scenario using both the Gaussian particle filter and the Gaussian k-best filter. Left: MAP estimate computed by the Gaussian particle filter with 100 samples. Right: probability of the correct (nominal) diagnosis.

overcommits to the fault mode. One possible explanation of this phenomenon is that once a fault mode is sampled and if it supports the observations well, it will remain in the surviving set of the particles, because there are no outgoing transitions from a fault mode. This issue does not affect k-best filters to such a degree, because the fault sequence probabilities need to multiply over time to a high product to be included in the leading set of trajectories. Over time, the fault may be disproved by the observations, and, may even be dropped from the list of leading trajectories, because its posterior may not build up in time before the sequence is dropped from the leading set of hypotheses.

As shown in Figure 6-6, this problem is remedied to some extent by using a version of the Gaussian particle filter that samples from the posterior (see Section 4.3.6. In this case, the filter overcommits to the fault mode at a later time $t = 3.5$s.

In the last, actuator failure scenario, the Gaussian particle filter detected the failure, although it misclassified it as a double-transition to the mode `has-ball=true`, `actuator=failed`. Even so, the correct diagnosis was detected earlier than with the k-best filter. Looking at the plot in Figure 6-1, we see that the fault should have been detected early, since the evolution of θ_2 in the fault scenario is significantly different from the nominal scenario. Therefore, our Gaussian particle correctly detects the

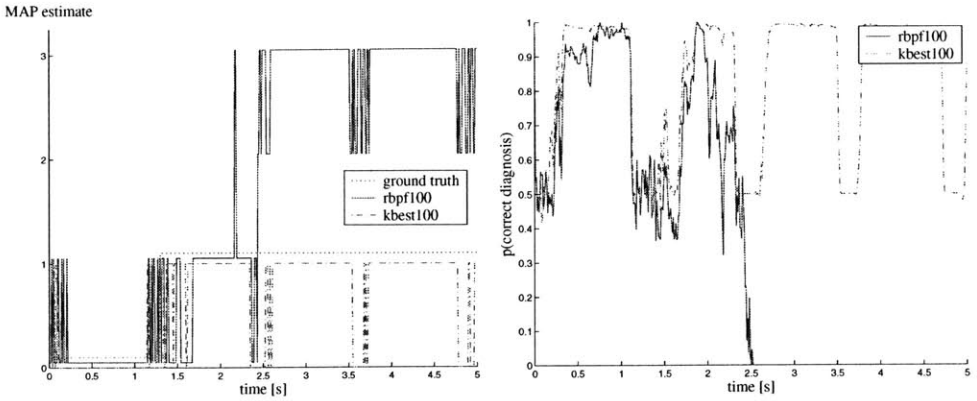

Figure 6-5: A single run for the ball capture scenario. Left: Maximum a posterior (MAP) estimate computed by RBPF and k-best filtering algorithm. Right: probability of the correct diagnosis `has-ball=yes` for $t \geq 1.3s$.

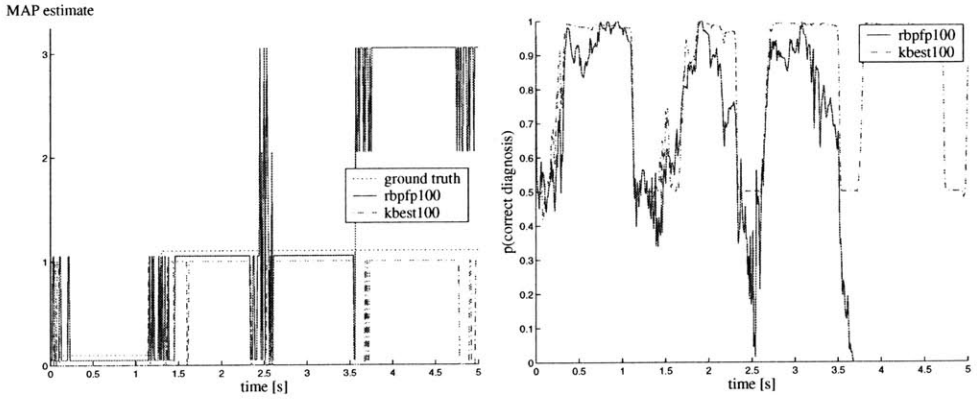

Figure 6-6: A single run for the ball capture scenario when sampling from posterior. Left: Maximum a posterior (MAP) estimate computed by RBPF and k-best filtering algorithm. Right: probability of the correct diagnosis `has-ball=yes` for $t \geq 1.3s$.

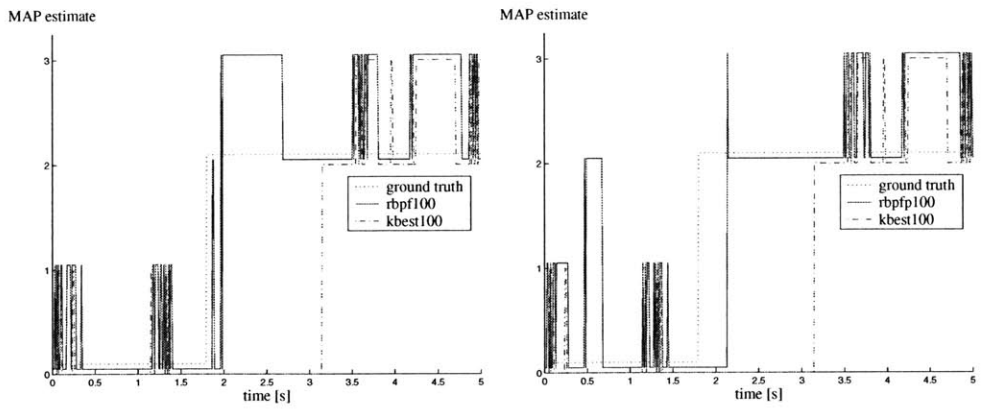

Figure 6-7: A single run for the actuator failure scenario, when sampling from the prior (left) and the posterior (right).

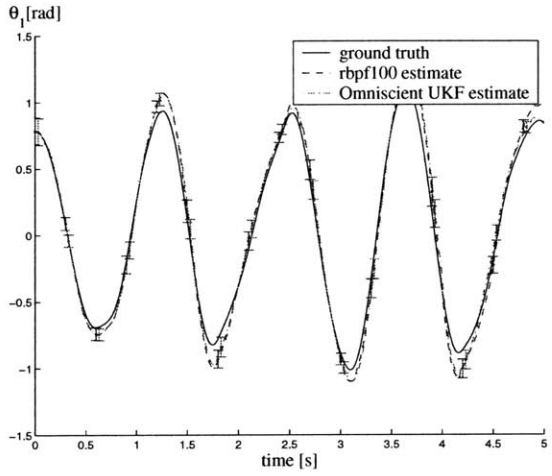

Figure 6-8: Filtered θ_1 for a single execution of the nominal scenario.

fault earlier in this test run.

In all the scenarios, the Gaussian particle filter tracked the continuous state very well. For example, Figure 6-8 shows the tracking for the nominal scenario. The continuous state estimate is very close to the omniscient UKF.[1]

6.3.2 Performance metrics

One of the biggest obstacles in evaluting the performance of hybrid state estimation algorithms is that inference with hybrid models is, in general, NP-hard [45], and it is

[1]Omniscient filter observes the discrete state directly and thus, can provide a more accurate continuous state estimate.

very difficult to obtain the true posterior distribution $p(\mathbf{x}_{c,t}, \mathbf{x}_{d,t}|\mathbf{y}_{1:t}, \mathbf{u}_{0:t})$. Sometimes, this distribution can be approximated by a particle filter with a large number of samples; however, the accuracy of such approximations may not be bounded tightly enough and has little chance of succeeding with our model, which could not be tracked reliably even when the discrete state was fully observed.

Instead, we use the following two metrics: percentage of the diagnostic faults, defined as $\frac{\#\ of\ wrong\ diagnoses}{\#\ time\ steps}$, and the mean square estimation error $((\hat{\mathbf{x}}_{c,t}-\mathbf{x}_{c,t})^T(\hat{\mathbf{x}}_{c,t}-\mathbf{x}_{c,t}))^{1/2}$, averaged out over all of the time steps and experiments. These metrics are far from perfect. For example, it is possible that the correct diagnosis may not be the most likely one. Furthermore, due to integration errors and the process and the observation noise, the optimal estimate might not be close to the ground truth. Nevertheless, these statistics do, in general, produce the correct results and have been employed in prior literature. [36] An alternative would be to use as a measure the likelihood of the correct diagnosis for discrete estimates and the KL divergence from the omniscient Kalman Filter for the continuous estimates. [43]

6.3.3 Average performance

Figures 6-9 through 6-14 show the percentage of diagnostic errors and the mean square tracking error for the three scenarios considered above. For each of the scenario, the algorithms were run on 10 random observation sequences with fixed mode assignments.

For the first two scenarios, the k-best filtering provides consistently better estimates. This is true especially for the percentage of diagnostic errors, which goes down by as much as 45 per cent.

For the actuator failure scenario, the Gaussian particle filter performs better than the k-best filter with 50 samples or more, and had produced consistently better continuous state estimates. The latter metric suggests that this result is not an accidental consequence of our choice of the diagnostic error metric: if the non-faulty mode estimates were indeed more likely, they should result in smaller continuous state error. Nevertheless, more testing will need to be performed to confirm this result.

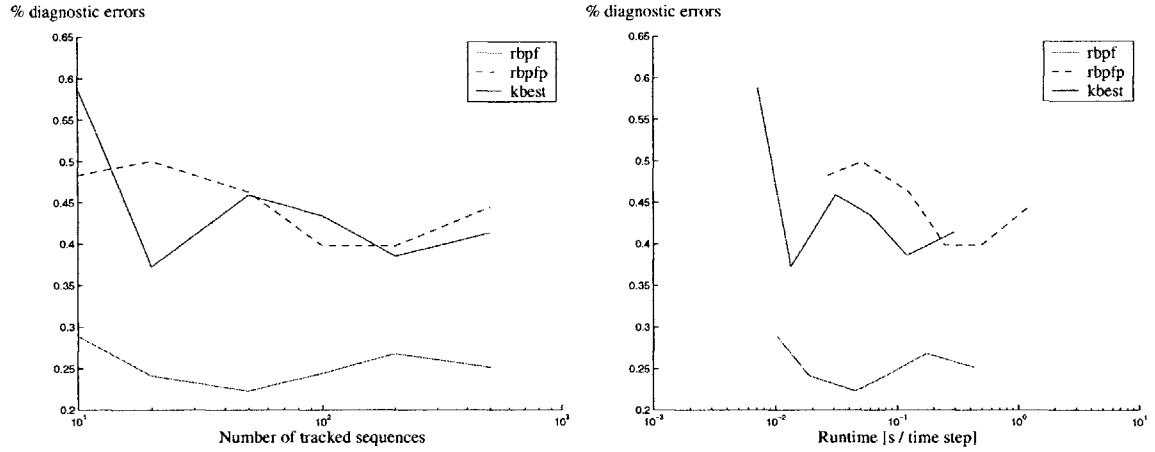

Figure 6-9: Percentage of diagnostic errors for the nominal scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

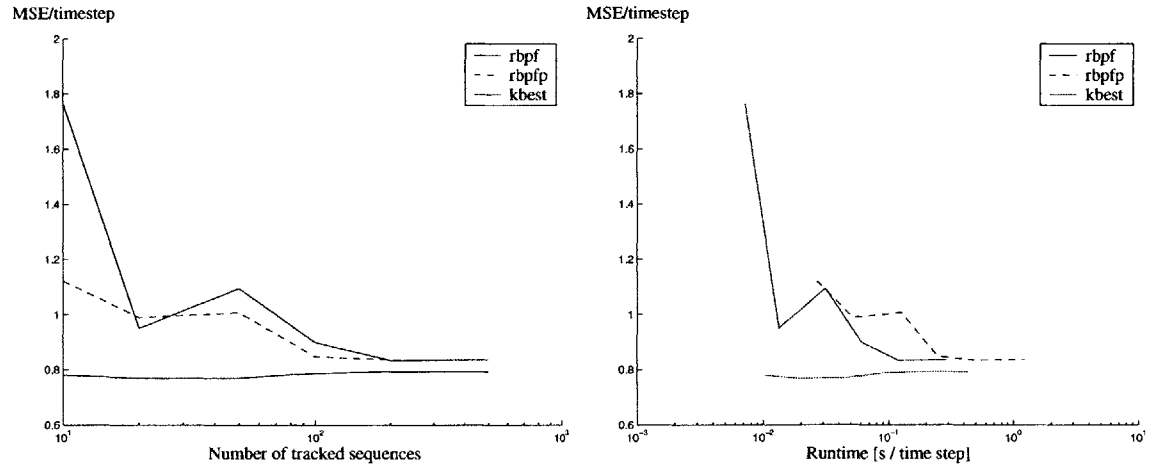

Figure 6-10: Mean square estimation error of the continuous state for the nominal scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

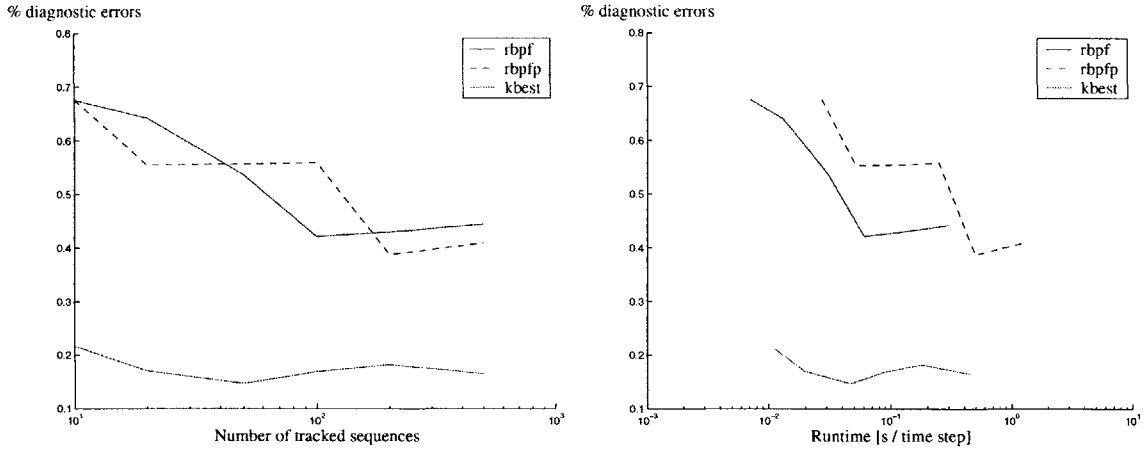

Figure 6-11: Percentage of diagnostic errors for the ball capture scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

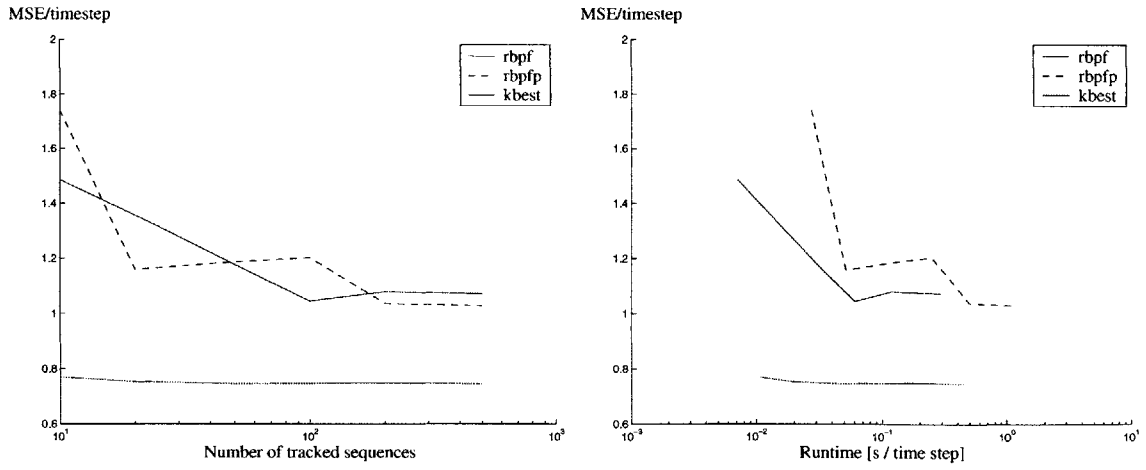

Figure 6-12: Mean square estimation error of the continuous state for the ball capture scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

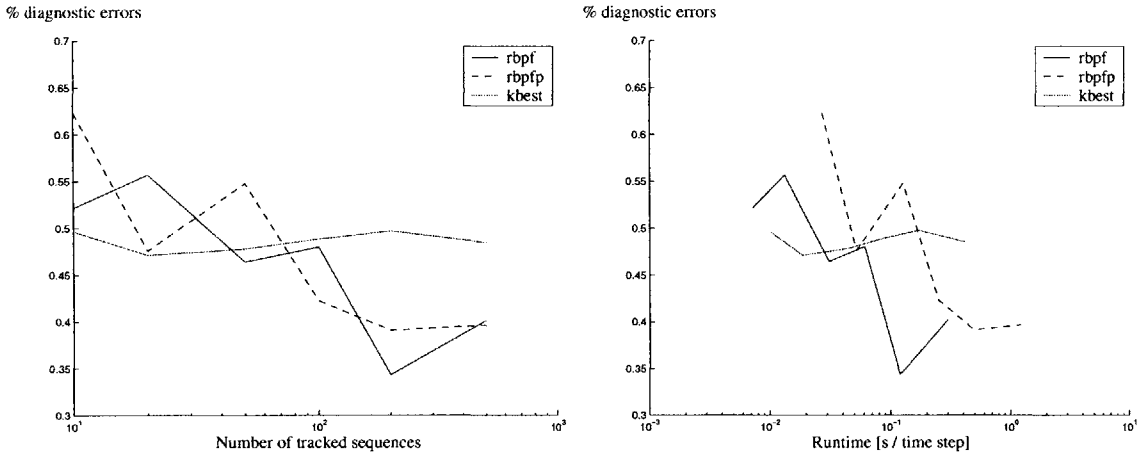

Figure 6-13: Percentage of diagnostic errors for the actuator failure scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

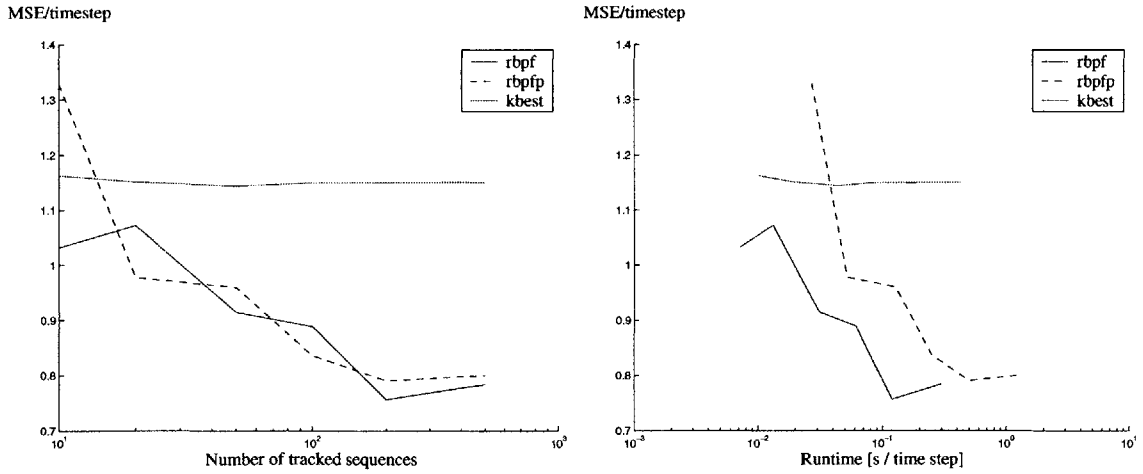

Figure 6-14: Mean square estimation error of the continuous state for the actuator failure scenario, as a function of number of tracked mode sequences (left) and running time per time step (right).

6.4 Discussion

In this demonstration, we focused on comparing the Gaussian particle filtering and k-best filtering techniques for selecting mode sequences. Based on our results, the Gaussian k-best filter tends to outperform the Gaussian particle filter in the cases when the likelihood of the correct diagnosis is sufficiently high. There are two reasons for this result. There are several reasons for this. One reason is that sampling introduces variance in the estimate by replacing the mode sequence weights with a set of discrete samples. By performing repeated sampling and resampling, the algorithm discards some information in the sequence that would otherwise eventually be noticable. Second reason is that k-best filtering focus the expansion of sequences to the space with the highest likelihood, thus performing better when high-likelihood events occur.

Nevertheless, there are instances when our Gaussian particle filter outperforms the k-best filter. In the actuator failure scenario, our Gaussian particle filter was able to detect the change much earlier than the Gaussian k-best filter. One possible explanation for this phenomenon is that either the fault diagnosis has too low a probability to be included in the leading set of sequences or does not accumulate large enough probability to survive the addition of higher-probability nominal sequences. The Gaussian particle filter, on the other hand, does not suffer from these effects, because it samples the mode sequences fairly, and will, sooner or later, sample the diagnosis, even if it has a low probability.

In this Chapter, we focused on evaluating the Gaussian representation and compared the performance of Gaussian particle filtering and k-best filtering in detecting high-likelihood sequences. Both evaluated algorithms could be immediately improved upon by using several techniques, including collapsing of the mode sequences [44], decomposition [50, 33, 44], and abstraction [58]. These issues are, however, orthogonal to the problem of selecting correct mode sequences, and our results should generalize when applying these optimizations.

Chapter 7

Summary and Future Work

7.1 Summary

In this thesis, we investigated the problem of estimating the state of system represented with probabilistic hybrid models. Our main accomplishment is an efficient Gaussian particle filtering algorithm, developed in Chapters 4 and 5, that handles autonomous mode transitions, concurrency, and nonlinearities present in Concurrent Probabilistic Hybrid Automata (CPHA). Through the technique of Rao-Blackwellised particle filtering, our algorithm significantly reduces the dimensionality of the sampled space and improves the performance of particle filtering. The key insight to addressing the autonomous transitions was reuse the continuous estimates associated with the tracked mode sequences.

In Chapter 4, we presented significant contributions related to discrete state transitions that depend on the continuous state (autonomous mode transitions). We extended the class of models, for which transition probabilities can be computed efficiently and explored the approximations that occur in the posterior of the continuous space when autonomous transitions are present. Due to the similarities in the theoretical development of both Gaussian particle filtering and k-best filtering, our results translate directly to prior k-best filtering algorithms that use sharp transition guards in the models. [32, 33].

Our contributions are, however, not merely theoretical. In Chapter 6, we have

demonstrated our algorithm on one a simulated highly nonlinear system, and empirically compared its performance with k-best filtering method [32, 44]. Our results [23] indicate that Gaussian particle filtering outperforms non-Rao-Blackwellised particle filtering approaches. For the cases when the correct diagnosis is repeatedly left out from the leading set of mode sequences, our Rao-Blackwellised particle filter outperforms k-best filtering, although more experimentation may be needed to confirm this result. For the nominal, high-likelihood sequences, k-best filtering is a clear winner. This development suggests that it may be possible to unify the two approaches in a stochastic search that shares the strengths of both methods.

7.2 Future work

We believe that hybrid estimation and fault diagnosis is an important field of research in artificial intelligence. Our hope is that this thesis will help drive further research in this area. The following research problems could be addressed using the results in this thesis.

7.2.1 Modeling

Semantics of the continuous state evolution

While we have clarified the semantics of discrete transition distribution in CPHA formalism slightly, the compatibility and determinedness properties of the continuous state evolution in CPHA are still left open. For example, the modeling formalism needs to provide theoretical provisions that disallow models with a set of conflicting equations, such as $y = \theta_1$ and $y = \theta_1 + 1$. Similarly, provisions need to be made regarding the semantics of models, in which some variables do not have their evolution completely specified. For example, it may be possible to gradually increase the variance of these variables [33] or follow a common engineering practice of setting their variances to a sufficiently high value to represent the high level of uncertainty that these variables have in the belief state. Alternatively, it may be possible to merge

the notions of continuous state constraints in to the formalism and make no assumptions about the evolution of the variables when no model is provided, although the semantics of and reasoning with such models may prove to be difficult.

Timed models

A very useful extension to probabilistic hybrid models would be to allow the mode transitions to depend on time. Allowing timed transition constraints would enable the use of model-based hybrid estimation capability in time-critical procedures, such as the entry, descent, and landing (EDL) sequence for a Mars lander.

The simplest way to extend the hybrid models with timed transitions is to allow a specially designated time variable in the transition guards. This is a common practice in the hybrid systems community, and would require minimal changes to estimation algorithm and the underlying modeling framework. Time can be represented as a continuous input variable, updated externally. Provided that a hybrid estimation algorithm is run at higher rate than the time difference between two consecutive timed transition, and provided that the transition constraints partition the $\mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times Time$ space, the algorithm will consider the transition in its computation of the belief state.

A more comprehensive time extension to hybrid models would involve allowing the transition distribution depend on the time continuously. This extension would allow the modelers to specify models, for which the transition probabilities change continuously as a function of time. A good starting point for this work would be the work in the context of Timed Concurrent Constraint Automata (TCCA). [37] The algorithm would span theads of mode sequences at different distinct time points. A collapsing strategy [44] could then be used to reduce the branching factor of the sequences.

7.2.2 State estimation

Errors due to autonomous transitions

As indicated in Chapter 4, a Gaussian can serve as a good approximation to the posterior even in the presence autonomous transitions, provided that there is sufficient amount of noise in the model. It would be useful to derive a bound on this error, as a part of a long-term effort to derive error bounds in hybrid estimates.

Improved proposal distribution through abstraction

The performance of a Gaussian particle filter could be greatly improved if it was possible to reduce the number of Kalman Filter updates needed to incorporate the latest observations into the proposal distribution. The techniques of abstraction [58], qualitative abstraction [53] could be used to guide sampling based on qualitative similarities or distinctions in the model. For particle filters, such heuristic is particularly easy to obtain, since it does not need to take a form of a strict upper- or lower-bound. The heuristic can also be more optimistic in directing the mode sequence expansion.

7.2.3 Hybrid model learning

A key enabler for hybrid estimation and fault diagnosis methods in practice is a hybrid model learning capability. Since the primary application of probabilistic hybrid models is to detect subtle symptoms from noisy data, hybrid models need to to be sufficiently accurate in order to be effective. Currently, the only way to obtain such models is by derivation from first principles of physics or by extensive experimentation. While these approaches work well for small systems, for large systems, such as the NASA JSC Advanced Life Support system [2], modeling costs could become prohibitively large.

An efficient hybrid state estimation algorithm can be directly used within the Expectation Maximization (EM) algorithm [16] to iteratively learn model parameters. The Expectation step of the EM algorithm uses the current model to label the data with the most likely mode and, as such, can be directly implemented using the algo-

rithm described in this thesis. The labeled data can then be used in the Maximization step to improve its estimates the model parameters. A preliminary application of this method was developed in [30]. Techniques, such as overlapping decomposition [60], could be used to reduce the dimensionality of the learned models and scale up the algorithm.

7.3 Conclusion

We believe that hybrid models are growing field of research with enticing challenges and numerous opportunity. Our hope is that this thesis shed some light on the green field of reasoning with hybrid models, and seeded some ideas that might grow to become beautiful, strong flowers in the future.

Bibliography

[1] Jupiter icy moons orbiter. http://www.jpl.nasa.gov/jimo/.

[2] Nasa's advanced life support. http://advlifesupport.jsc.nasa.gov/.

[3] Nasa's Mars exploration program. http://marsprogram.jpl.nasa.gov/missions/future/msl.html.

[4] G.A. Ackerson and K.S. Fu. On state estimation in switching environments. *IEEE Transactions on Automatic Control*, 15:10–17, 1970.

[5] H. Akashi and H. Kumamoto. Construction of discrete-time nonlinear filter by monte carlo methods with variance-reducing techniques. *International Journal of Control*, 19(4):211–221, 1975. (in Japanese).

[6] H. Akashi and H. Kumamoto. Random sampling approach to state estimation in switching environments. *Automatica*, 13:429–434, 1977.

[7] B. Anderson and J. Moore. *Optimal Filtering.* Information and System Sciences Series. Prentice Hall, 1979.

[8] D. Avitzour. A stochastic simulation Bayesian approach to multitarget tracking. *IEE Proceedings on Radar Sonar and Navigation*, 142(2):41–44, April 1995.

[9] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* Wiley Series in Applied Probability and Statistics. John Wiley & sons, 1994.

[10] H.A.P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33, 1988.

[11] J. Carpenter, P. Clifford, and P. Fearnhead. Building robust simulation-based filter for evolving data sets. Technical report, Technical Report Unversity of Oxford, 1999.

[12] G. Casella and C. P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[13] D. Crisan. Particle filters - a theoretical perspective. In A. Doucet, N. Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, chapter 2, pages 17–41. Springer-Verlag, 2001.

[14] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[15] R. Dearden and D. Clancy. Particle filters for real-time fault detection in planetary rovers. In *Proceedings of the 13th International Workshop on Principles of Diagnosis (DX02)*, pages 1–6, May 2002.

[16] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data. *Journal of the Royal Statistical Society Ser. B*, 39:1–38, 1977.

[17] A. Doucet. *Monte Carlo Methods for Bayesian Estimation of Hybrid Markov Models. Application to Radiation Signals.* PhD thesis, Universit Paris-Sud, Orsay, 1997. (in French with Chapters 4 and 5 in English).

[18] A. Doucet, J.F.G. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2000.

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

[20] A. Doucet, N. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump markov linear systems. Technical Report CUED/FINFENG/TR 359, Cambridge University Department of Engineering, 1999.

[21] Arnaud Doucet. On sequential simulation-based methods for bayesian filtering. Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.

[22] N. Freitas. Rao-Blackwellised particle filtering for fault diagnosis. *IEEE Aerospace*, 2002.

[23] S. Funiak and B. Williams. Multi-modal particle filtering for hybrid systems with autonomous mode transitions. In *Proceedings of SafeProcess 2003 (also published in DX-2003*, June 2003.

[24] Alan Genz and Koon-Shing Kwong. Numerical evaluation of singular multivariate normal distributions. *J. Stat. Comp. Simul*, 68(1-21), 2000.

[25] J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 24:1317–1399, 1989.

[26] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state stimation. In *IEE Proceedings-F*, volume 140, pages 107–113, 1993.

[27] J. E. Handschin. Monte carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6:555–563, 1970.

[28] J. E. Handschin and D. Q. Mayne. Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International Journal of Control*, 9(5):547–559, 1969.

[29] P. Hanlon and P. Maybeck. Multiple-model adaptive estimation using a residual correlation kalman filter bank. *IEEE Transactions on Aerospace and Electronic Systems*, 36(2):393–406, 2000.

[30] M. Henry. Model-based estimation of probabilistic hybrid automata. Master's thesis, Massachusetts Institute of Technology, May 2002.

[31] T. Higuchi. Monte Carlo filter using the genetic algorithm operators. *Journal of Statistical Computation and Simulation*, 59(1):1–23, 1996.

[32] M. Hofbaur and B. C. Williams. Mode estimation of probabilistic hybrid systems. In *Intl. Conf. on Hybrid Systems: Computation and Control*, May 2002.

[33] M. W. Hofbaur and B. C. Williams. Hybrid diagnosis with unknown behavioral modes. In *Proceedings of the 13th International Workshop on Principles of Diagnosis (DX02)*, pages 97–105, May 2002.

[34] M. W. Hofbaur and B. C. Williams. Hybrid estimation of complex systems. *accepted for publication in IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 2004.

[35] M.W. Hofbaur. Hybrid estimation and its role in automation. Habilitationsschrift, Faculty of Electrical Engineering, Graz University of Technology, Austria, September 2003.

[36] Frank Hutter and Richard Dearden. The gaussian particle filter for diagnosis of non-linear systems. In *Proceedings of the 14th International Conference on Principles of Diagnosis(DX'03)*, pages 65–70, Washington, DC, USA, June 2003.

[37] M.D. Ingham. *Timed model-based programming: Executable specifications for robust mission-critical sequences.* PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge, MA, June 2003.

[38] H. Joe. Approximations to multivariate normal rectangle probabilities based on conditional expectations. *Journal of the American Statistical Association*, 90(431):957–964, September 1995.

[39] S. J. Julier and J. K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Proceedings of AeroSense: The 11th Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 1997.

[40] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Comptuational and Graphical Statistics*, 5:1–25, 1996.

[41] A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[42] X. Koutsoukos, J. Kurien, and F. Zhao. Monitoring and diagnosis of hybrid systems using particle filtering methods. In *MTNS 2002*, August 2002.

[43] U. Lerner. *Hybrid Bayesian Networks for Reasoning About Complex Systems.* PhD thesis, Stanford University, October 2002.

[44] U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proc. of the 17th National Conference on A. I.*, pages 531–537, July 2000.

[45] Uri Lerner and Ronald Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 310–318, Seattle, Washington, August 2001.

[46] J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[47] N. A. Lynch, R. Segala, and F. W. Vaandrager. Hybrid i/o automata. *Information and Computation*, 185(1):105–157, August 2003.

[48] Rubn Morales-Menndez, Nando de Freitas, and David Poole. Real-time monitoring of complex industrial processes with particle filters. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2002.

[49] K. Murphy and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In A. Doucet, N. Freitas, and N. Gordon, editors, *Sequential*

Monte Carlo Methods in Practice, chapter 24, pages 499–515. Springer-Verlag, 2001.

[50] Brenda Ng, Leonid Peshkin, and Avi Pfeffer. Factored particles for scalable monitoring. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 2002.

[51] R. P. Paul. *Robot Manipulators.* MIT Press, August 1982.

[52] Stuart J. Russell and Peter Norvig, editors. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 2003.

[53] M. Sachenbacher and P. Struss. Automated qualitative domain abstraction. In *Proceedings IJCAI'03*, pages 382–387, 2003.

[54] Michael Stevens. Bayes++ open source bayesian filtering classes. http://bayesclasses.sourceforge.net/Bayes++.html.

[55] Sebastian Thrun, John Langford, and Vandi Verma. Risk sensitive particle filters. In *Neural Information Processing Systems (NIPS)*, December 2001.

[56] J. Tugnait. Detection and estimation for abruptly changing systems. *Automatica*, 18:607–615, 1982.

[57] Rudolph van der Merwe, Nando de Freitas, Arnaud Doucet, and Eric Wan. The Unscented Particel Filter. In *Advances in Neural Information Processing Systems 13*, 2000.

[58] Vandi Verma, Sebastian Thrun, and Reid Simmons. Variable resolution particle filter. In *In Proceedings of International Joint Conference on Artificial Intelligence.* AAAI, August 2003.

[59] E. A. Wan and R. van der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC)*, 2000.

[60] B. C. Williams and B. Millar. Decompositional, model-based learning and its analogy to model-based diagnosis. In *Proceedings of AAAI-98*, pages 197–203, 1998.

[61] B. C. Williams and P. Nayak. A model-based approach to reactive self-configuring systems. In *Proceedings of AAAI-96*, pages 971–978, 1996.