Robust Execution for Stochastic Hybrid Systems

by

Lars James Christopher Blackmore

Submitted to the Department of Aeronautics and Astronautics in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2007

Author .... [illegible] ....
Department of Aeronautics and Astronautics
22nd May 2007

Certified by .... [illegible]
Brian C. Williams
Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .... [illegible]
Nicholas Roy
Assistant Professor of Aeronautics and Astronautics
Thesis Advisor

Certified by .... [illegible]
Jean-Jacques Slotine
Professor of Mech. Eng. and Inf. Sci., and Brain and Cog. Sci.
Thesis Advisor

Accepted by .... [illegible]
New Chair David Darmofal on behalf of
Jaime Peraire
Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

Robust Execution for Stochastic Hybrid Systems

by

Lars James Christopher Blackmore

Submitted to the Department of Aeronautics and Astronautics
on 22nd July 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Unmanned systems, such as Autonomous Underwater Vehicles (AUVs), planetary rovers and space probes, have enormous potential in areas such as reconnaissance and space exploration. However the effectiveness and robustness of these systems is currently restricted by a lack of autonomy. Previous work introduced the concept of a model-based executive, which increases the level of autonomy, elevating the level at which systems are commanded. This simplifies the operator's task and leaves degrees of freedom in the plan that allow the executive to optimize resources and ensure robustness to uncertainty. Uncertainty arises due to uncertain state estimation, disturbances, model uncertainty and component failures. This thesis develops a model-based executive that reasons explicitly from a stochastic hybrid discrete-continuous system model to find the optimal course of action, while ensuring the required level of robustness to uncertainty is achieved.

Our first contribution is a novel 'Particle Control' approach for robust execution of state plans with stochastic hybrid systems. We introduce the notion of chance-constrained state plan execution; this means that the executive ensures tasks in the state plan have at least a specified minimum probability of success. The minimum probabilities are specified by the operator, enabling conservatism to be traded against performance. In order to make optimal chance-constrained execution tractable, the Particle Control approach approximates the system's state distribution using samples or 'particles' and optimizes the evolution of these particles to achieve chance-constrained state plan execution. In this manner particle control solves a tractable deterministic approximation to the original stochastic problem; furthermore the approximation becomes exact as the number of particles approaches infinity. For an important class of hybrid discrete-continuous system known as Jump Markov Linear Systems, the resulting deterministic optimization can be posed as a Mixed Integer Linear Program and solved to global optimality using efficient commercially-available solvers.

Our second contribution is 'active' hybrid estimation subject to state plan constraints. Exact hybrid state estimation in stochastic hybrid systems is, in general, intractable. Tractable approximate hybrid estimation methods can lose track of the true hybrid state. In this thesis we develop an active hybrid estimation capability, which probes the system in order to reduce uncertainty in the hybrid state. This approach generates control sequences to minimize the probability of approximate hybrid estimation losing the true mode sequence, while ensuring that the state plan is satisfied subject to chance constraints. In order to make this problem tractable, we develop an analytic upper bound on the probability of losing the true mode sequence, and use a convex constraint tightening approach to approximate the chance constraints in the problem.

Our final contribution is a novel hybrid model-learning approach. Specifying accurate hybrid system models is essential for accurate estimation and control, but is also extremely challenging. The hybrid executive must therefore determine hybrid system models from observed data. In this thesis we present an approximate Expectation-Maximization method for hybrid model learning; this method extends prior approaches to deal with mode transitions that depend on the continuous state.

List of Figures

Chapter 1

Introduction

Unmanned systems such as Unmanned Air Vehicles (UAVs), Autonomous Underwater Vehicles (AUVs), planetary rovers and space probes have enormous potential in areas such as reconnaissance and space exploration. Unmanned missions can explore regions that are inhospitable to humans, such as outer space or the earth's oceans, with orders of magnitude less cost compared to manned missions, and without risk to human life. However current unmanned systems are controlled closely by teams of human operators. This lack of autonomy increases the cost of the mission and seriously restricts the effectiveness and robustness of the unmanned system. With humans in the loop at almost all decision points, communication limitations introduce serious problems; for example, causing unnecessary dead time in a planetary rover's science mission; or preventing a Mars lander's timely recovery from critical faults. Hence in order to realize their potential, unmanned systems require a dramatic increase in the level of autonomy they exhibit.

One approach to achieving this increase in autonomy is to elevate the level at which systems are commanded from low-level system commands to desired states. For example, a UAV mission should be specified in terms of the desired regions of the state space that the UAV must reach, instead of in terms of the actuator effort to be applied. This elevated level of commanding brings a number of advantages. First, the mission planner's task is simplified. Second, commanding at the level of desired state leaves degrees of freedom in the plan that allow the autonomous system to plan for uncertainty and to optimize resources. In the UAV case, there are an infinite number of low-level control sequences that will take the UAV from its initial state to a specified goal region; we would like to choose from this set the sequence that minimizes fuel use, or time, while being robust to uncertainty such as wind disturbances. Prior work introduced the concept of a model-based executive[133], which takes as its input a state plan and issues low-level system commands such that the state plan is achieved. [133] introduced the Titan model-based executive, which controls discrete-event, under-actuated systems according to a *Qualitative State Plan* comprised of a sequence of constraints on goal states. This is insufficient for autonomous vehicles such as UAVs, since these are not discrete-event systems; they have continuous dynamics as well as discrete changes in operational state.

More recent work has developed executives that handle systems with continuous dynamics. Sulu [76] addresses the problem of model-based execution of continuous dynamic systems, while Chekov [63] deals with hybrid discrete-continuous systems whose mode transitions are deterministic. A key challenge facing these executives is uncertainty, which arises due to several factors. First, partial observability and noisy sensors mean that the system state must be estimated with resulting uncertainty in the system state estimate. Second, disturbances such as wind or currents act on a system, making the future state deviate from its planned value in a manner which is a priori uncertain. Third, component failures may occur, causing unexpected changes in the system dynamics. Fourth, the system dynamics may not be known exactly, and finally, the location of obstacles in the environment may be uncertain. These forms of uncertainty are best captured using stochastic models; state and model parameter estimators give stochastic representations of state and parameter uncertainty, respectively; wind disturbances are best characterized using stochastic processes, and component failures occur at random.

Sulu assumes no uncertainty or hidden state. Sulu responds to uncertainty reactively, by re-planning continuously based on the most recent observations of the system state. Chekov handles uncertainly reactively, as well as using a heuristic approach to robustness that keeps the system state in the middle of the region for which success can be guaranteed. Neither Sulu nor Chekov select a robust course of action by reasoning explicitly about stochastic uncertainty.

This thesis develops a model-based executive that reasons explicitly from a stochastic hybrid model to find the optimal course of action while ensuring the required level of robustness to uncertainty is achieved. Stochastic hybrid models explicitly encode all the forms of uncertainty previously mentioned. Development of this executive poses three different sub-problems:

1. The problem of generating a complete, robust, optimal control sequence in order to execute the input qualitative state plan, based on the plant model and estimates of the plant state. This task is performed by the *robust hybrid controller*

2. The problem of estimating the hybrid state given the plant model, past observations and control inputs. This is performed by the *hybrid state estimator*

3. The problem of estimating the parameters of the hybrid plant model given past observations and control inputs. This is performed by the *hybrid parameter estimator.*

In order to solve these problems we present three main contributions. First, we present a novel method for robust execution of state plans for using stochastic hybrid models. Second, we introduce state-plan constrained active estimation. This enables the hybrid state estimator to issue control inputs in order to disambiguate optimally between different modes, while ensuring robust state plan execution. Third, we develop a novel hybrid model learning algorithm that enables the executive to adapt its model parameters according to observed data to account for changing system characteristics and inaccurate model specifications. These contributions are summarized in Sections 1.1 through 1.3.

1.1 Chance-Constrained Execution of Qualitative State Plans

1.1.1 Chance-Constrained Execution Capability

Our first contribution is a novel method for robust execution of state plans for stochastic hybrid systems, which we call the PC-QSP algorithm. We introduce the notion of chance-constrained state plan execution; the executive issues commands that are robust, in the sense that they ensure the state plan is satisfied with a probability above a specified minimum. Many stochastic uncertainty models have infinite support; for example in the case of a Gaussian noise model. This means there is always a finite probability of an arbitrarily large noise value occurring (in contrast to set-bounded uncertainty, where a hard upper bound exists on the magnitude of the uncertainty). This means that a plan can typically not be generated that has a 100% probability of success. We can however require that the overall state plan is satisfied with a certain probability less than 1. In addition, minimum probabilities of satisfaction can be imposed on individual tasks, or groups of tasks, within the state plan. These probabilities are specified by the operator, enabling conservatism to be traded against performance in a meaningful manner; a plan constrained to have a low probability of failure gives the operator high confidence in success, but will typically be more expensive in terms of time or fuel, for example.

Figure 1-1 shows a very simple AUV mission and the corresponding Qualitative State Plan. An informal description of the Qualitative State Plan is:

> While minimizing fuel usage, end in the goal region 4 time units after the start of the mission and avoid obstacles. Mission failure, defined as collision with an obstacle or failure to be at the goal at 10 time units, can occur with a probability at most 0.1.

Figure 1-2(left) shows the plan generated by the PC-QSP algorithm. Note how the expected path is some distance away from the obstacles, ensuring that the probability of mission failure is at most 0.1. In this example, the chance constraint is tight, meaning that the probability of mission failure is exactly 0.1. For the sake of comparison, Figure 1-2(right) shows a typical plan generated without taking into account uncertainty. Here, the most likely state is required to avoid obstacles and reach the goal region, as is the case with the Sulu executive. Note that the probability of mission failure is very high; failure to reason explicitly from a stochastic model leads to a plan that is highly brittle to uncertainty. We now compare Figure 1-2 with a more conservative plan. We change the chance

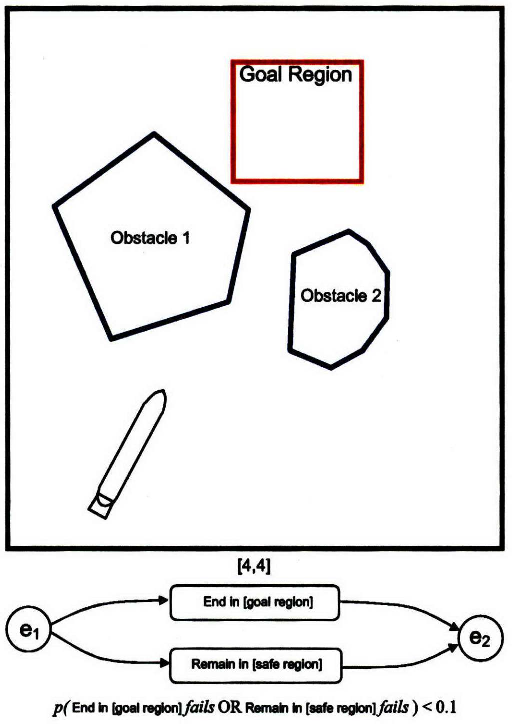


Figure 1-1. Simple AUV mission and corresponding Qualitative State Plan. The AUV must end in the goal region at $t = 4$ and must avoid obstacles (remain in the safe region). Mission failure occurs if either the AUV fails to reach the goal region, or the AUV collides with an obstacle. The Qualitative State Plan requires failure to occur with a probability at most 0.1. Uncertain localization means that the initial state of the aircraft is uncertain. In addition, the tailcone actuators can fail, in which case they have no effect on the vehicle.

constraint in the Qualitative State Plan to allow a probability of mission failure of at most 0.01. Figure 1-3 shows the plan generated by the PC-QSP algorithm. Note that the planned path no longer goes through the narrow corridor, but instead takes a longer path around both obstacles. This plan requires more fuel than the less conservative one, but has a lower probability of failure.

1.1.2 Chance-Constrained Particle Control Approach

Optimal chance-constrained task plan execution for stochastic hybrid systems is an extremely challenging constrained stochastic optimization problem for a number of reasons. First, while previous executives have planned with regard to future hybrid state trajectories that are deterministic, we must plan with regard to the future hybrid state trajectory *distributions*; this space of distributions is far larger than that of deterministic trajectories. Second, there are many decision variables over which we must optimize, namely the sequence of commands applied to the hybrid system, and furthermore the commands can take on values from a continuous (uncountable) set. By contrast planning with purely discrete decision variables is typically much more straightforward for comparable problem sizes. Finally, we must optimize with constraints on the probability of plan failure. Even evaluating the probability of plan failure is intractable in closed form, hence constraining this

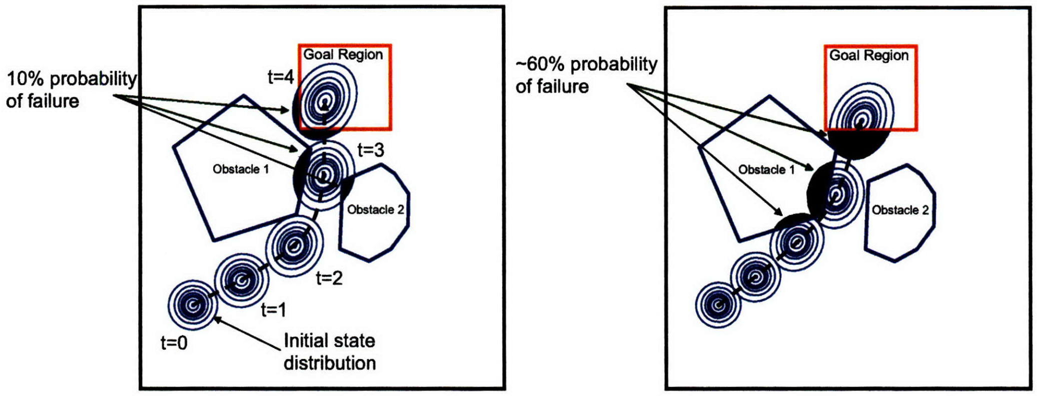

10% probability of failure
Goal Region
t=4
t=3
Obstacle 1
Obstacle 2
t=2
t=1
t=0
Initial state distribution
~60% probability of failure
Goal Region
Obstacle 1
Obstacle 2

Figure 1-2. Left: Plan generated by PC-QSP algorithm for AUV mission with maximum probability of failure set to 0.1. The predicted distribution of the system state at each time step is shown. The expected path, shown as the dashed line, is some distance away from the obstacles. The plan ensures that the probability of mission failure is at most 0.1. **Right:** Plan generated without taking into account uncertainty. Although the most likely (expected) state avoids the obstacles and reaches the goal region, the probability of mission failure is high.

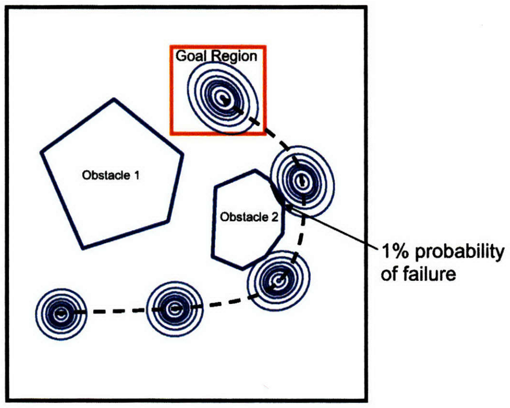


Figure 1-3. Plan generated by PC-QSP algorithm for AUV mission with maximum probability of failure set to 0.01. The optimal plan no longer takes the AUV between the obstacles, but instead takes the AUV around both obstacles to achieve a lower probability of failure.

value in an optimization is particularly difficult.

In order to make the problem of optimal chance-constrained state plan execution tractable, we introduce a novel chance-constrained particle control approach, illustrated in Figure 1-4. This approach draws from prior work in estimation, which uses samples or 'particles' to approximate the system state distribution. Particles are particularly useful as they can approximate arbitrary probability distributions, even the multimodal state distributions that occur in hybrid discrete-continuous models. The particle approximation also gives approximations to important values, namely the probability of events such as plan failure, and expected values such as the expected plan cost. Furthermore, as the number of particles tends to infinity, the approximations become exact. The PC-QSP algorithm works by sampling all uncertain variables to get a finite set of particles, whose trajectory depends explicitly on the control input sequence. The algorithm then approximates the chance-constrained execution problem in terms of these particles, thereby approximating an intractable stochastic optimization problem as a tractable deterministic one. For an important class of hybrid discrete-continuous system known as Jump Markov Linear Systems, the resulting deterministic optimization problem can be posed as a Mixed Integer Linear Program and solved to global optimality using efficient commercially-available solvers. Furthermore, as the number of particles tends to infinity, the approximate deterministic optimization problem converges to the true stochastic optimization problem. This gives an *any-time* solution to the execution problem, in that the number of particles is dictated by the available computational resources, and the more particles that are used, the more accurate the results.

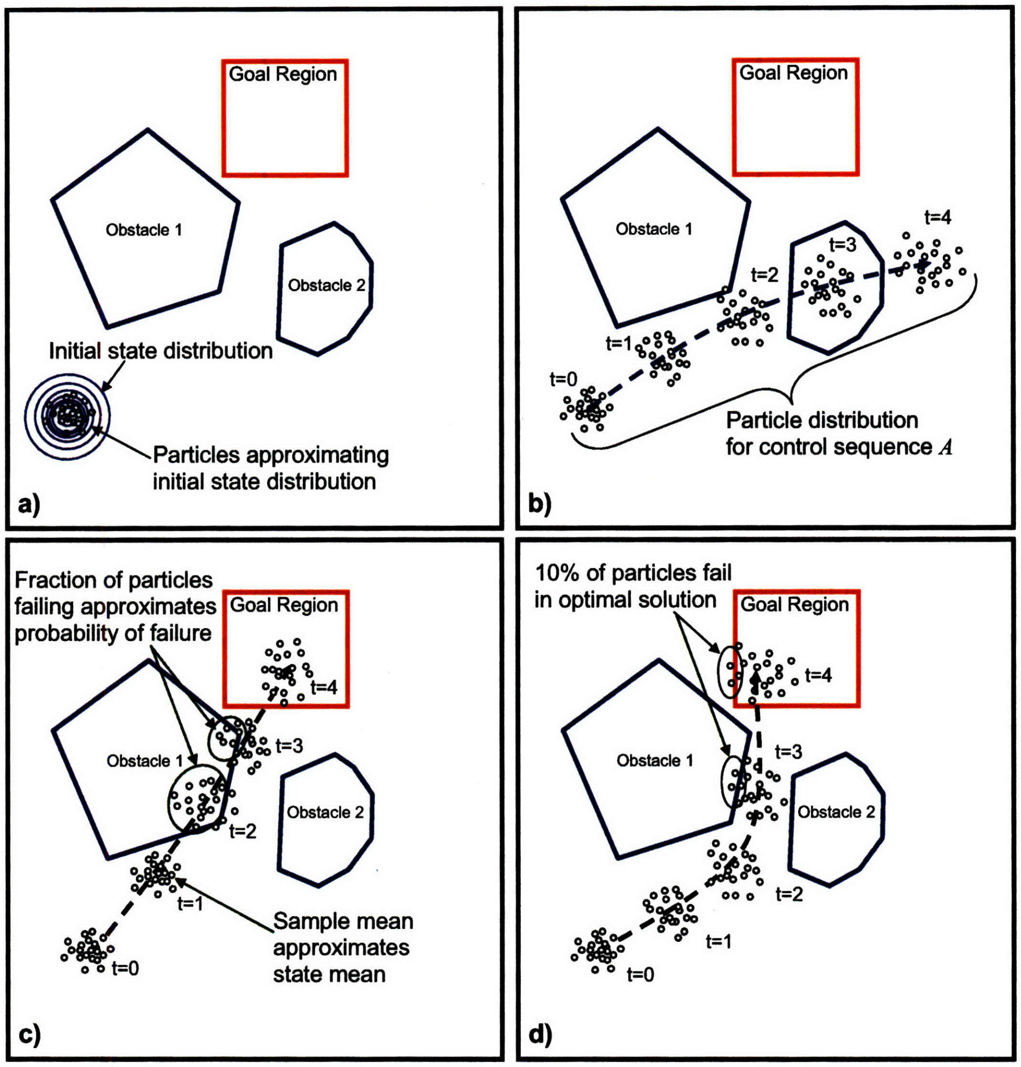
We demonstrate the Particle Control approach in simulation for four systems; the Monterey Bay Aquarium Research Institute (MBARI) Autonomous Underwater Vehicle, which performs autonomous ocean science missions; a Boeing 747 operating in heavy turbulence; an Unmanned Air Vehicle operating in a cluttered environment; and an Unmanned Ground Vehicle that is prone to brake failure. All of these systems can be modeled effectively using JMLS, subject to state constraints. We show that the Particle Control approach generates control inputs that solve the approximated execution problem, and that the approximation error converges in probability to zero as the number of particles used increases. For reasonably-sized particle sets, generation of control sequences for full-size system models can be carried out in seconds. We show, however, that the Particle Control approach performs poorly for systems with low-probability discrete transitions such as component failures. This is because such events are rarely sampled, yet they have a large effect on the optimal plan. To overcome this, we extend the Particle Control approach to use importance-weighted particles[37]. The key idea is to sample mode sequences from a *proposal distribution*, which is chosen so that failures are more likely to be taken into account. By correct selection of the importance weights, we retain the convergence properties of the Particle Control approach. We show in simulation that the resulting PC-WEIGHTED algorithm performs much better than the original Particle Control approach for the same number of particle.

1.2 Active Hybrid Estimation

1.2.1 Active Hybrid Estimation Capability

Our second contribution is Qualitative State Plan constrained active hybrid estimation. In stochastic hybrid systems, the system state is only partially observable and furthermore estimating the hybrid state distribution exactly is, in general, intractable. The discrete state is in most cases not observed at all, and is therefore particularly challenging to estimate. Tractable approximate hybrid estimation methods can lose track of the true discrete mode sequence, in which case a control algorithm using the state estimate will issue commands that are sub-optimal or even unsafe.

Within the state-plan specification, however, the hybrid executive has additional degrees of



Goal Region
Obstacle 1
Obstacle 2
Initial state distribution
Particles approximating initial state distribution
a)
t=0
t=1
t=2
t=3
t=4
Particle distribution for control sequence A
b)
Fraction of particles failing approximates probability of failure
Sample mean approximates state mean
c)
10% of particles fail in optimal solution
d)

Figure 1-4. Illustration of particle control algorithm for chance constrained execution of Qualitative State Plans, applied to simple AUV scenario. In a) the initial state distribution (and all other uncertain distributions) are sampled to give a finite number of probabilistic particles. The trajectory of each particle depends on the sequence of control inputs applied. Applying the same control sequence to each particle, the effect of a given control sequence on the particle distribution can be evaluated, as shown in b). We approximate the chance constrained execution problem in terms of the particles. As shown in c), the probability of failure is approximated using the fraction of particles failing, the state mean is approximated using the sample mean, and the expected cost is approximated using the mean of the cost function evaluated for each particle. These approximations give a deterministic optimization problem, which is solved to give the optimal solution to the approximated problem. The optimal solution for this example is shown in d). As the number of particles tends to infinity, the approximated problem converges to the true problem.

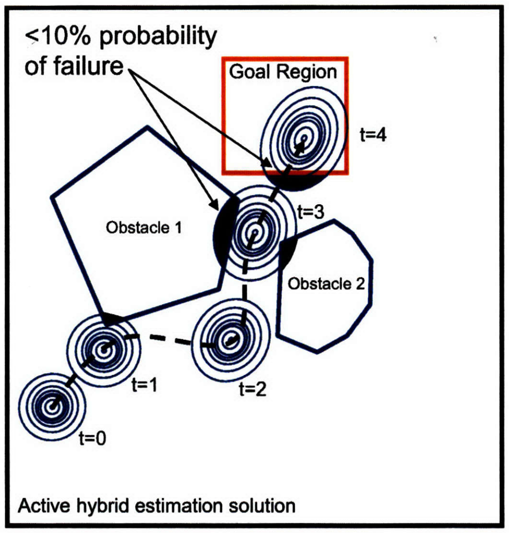

<10% probability of failure
Goal Region
t=4
t=3
Obstacle 1
Obstacle 2
t=1
t=2
t=0
Active hybrid estimation solution

Figure 1-5. Plan generated by AE-QSP algorithm for AUV mission with maximum probability of failure set to 0.01. The algorithm uses the additional degrees of freedom in the Qualitative State Plan to probe the system in order to reduce uncertainty in the hybrid state. In the AUV case, this means peforming a sideways maneuver to determine whether the tailcone actuators have failed. The solution returned satisfies the Qualitative State Plan.

freedom that can be used by the executive to probe the system in order to reduce uncertainty in the hybrid state. In this thesis we develop an *active hybrid estimation* capability that performs this probing, which we refer to as the AE-QSP algorithm. This capability finds a solution that is locally optimal with regard to *estimation* rather than resources (such as fuel). Any solution generated by the active estimation algorithm is guaranteed to satisfy the chance-constrained Qualitative State Plan. The AE-QSP capability is illustrated in Figure 1-5.

1.2.2 Upper Bound Minimization Approach

Peforming active hybrid estimation while ensuring state-plan satisfaction is challenging for two key reasons. First we must define a cost function for active estimation that evaluates the utility of a given control sequence in terms of disambiguating the hybrid system state. Second, we must ensure that the state plan is specified, while optimizing this cost function. We claim that the ideal cost function is the probability of an approximate hybrid estimator losing track of the true discrete mode sequence, since prior work has shown empirically that this leads to divergence of the hybrid state estimate. Unfortunately this cannot be evaluated in closed form, and hence cannot be minimized using standard optimization techniques. To overcome this problem we provide a tractable, analytic upper bound on the probability of the approximate hybrid estimator losing the true mode sequence. Upper bound minimization is intuitively appealing, because, if we can achieve a small upper bound, we guarantee a small value of the original cost function. In deriving the analytic upper bound we extend prior results in Bayesian hypothesis testing to the case of multiple (more than 2) hypotheses.

Having defined the upper bound, the remaining challenge is to perform optimization subject to constraints on the probability of failure (chance constraints). This is challenging, since the probability of failure cannot even be evaluated in closed form. As for the PC-QSP algorithm, the probability of failure can be approximated using particles; however this leads to highly nonconvex constraints. This makes optimization of the nonlinear cost function for active hybrid estimation intractable. The key idea in solving this problem is to start from a resource-optimal solution that satisfies the Qualitative State Plan generated using the PC-QSP algorithm. We then tighten the constraints imposed by the chance-constrained state plan until they are convex. Intuitively, this means fixing the times at which events occur, fixing which particles succeed and fixing which constraints are satisfied by each successful particle at each time step. This is illustrated in Figure 1-6. With a convex feasible region, the cost function for active estimation can be minimized using existing local optimization techniques.

In developing the active hybrid estimation approach we provide three distinct active probing capabilities. First, we develop a method for optimal discrimination between a finite set of linear dynamic models subject to expected state constraints. The resulting AE-MM algorithm designs control inputs to minimize an upper bound on the probability of a Multiple-Model selection algorithm [106] making an error, while ensuring that a control task, defined in terms of the expected state, is achieved. We show, using a simulated aircraft scenario, that this algorithm significantly reduces the probability of error compared to a power-limited auxiliary signal[50] and a fuel-optimal control sequence. Second, we develop the AE-JMLS algorithm for Jump Markov Linear Systems. This active estimation capability minimizes an upper bound on the probability of an approximate hybrid estimator losing the true mode sequence, while ensuring that expected state constraints are satisfied. We demonstrate this algorithm in simulation using an aircraft scenario. Finally, we develop the AE-QSP algorithm for active estimation of Jump Markov Linear Systems subject to chance-constrained Qualitative State Plan satisfaction, already described in this section. We demonstrate using an MBARI AUV scenario that the algorithm can perform active probing of the hybrid system state while ensuring that the mission, defined in terms of a Qualitative State Plan, is satisfied with the required probability.

1.3 Hybrid Model Learning

Our final contribution is a novel hybrid model-learning approach. Prior work in model-based hybrid estimation has shown that specifying accurate hybrid system models is essential for good performance while also being extremely challenging. This is equally true for control of hybrid systems. The hybrid executive must therefore determine hybrid system models from observed data. Since the discrete and continuous dynamics of such models are coupled, partially observable and stochastic, this is a very challenging problem.

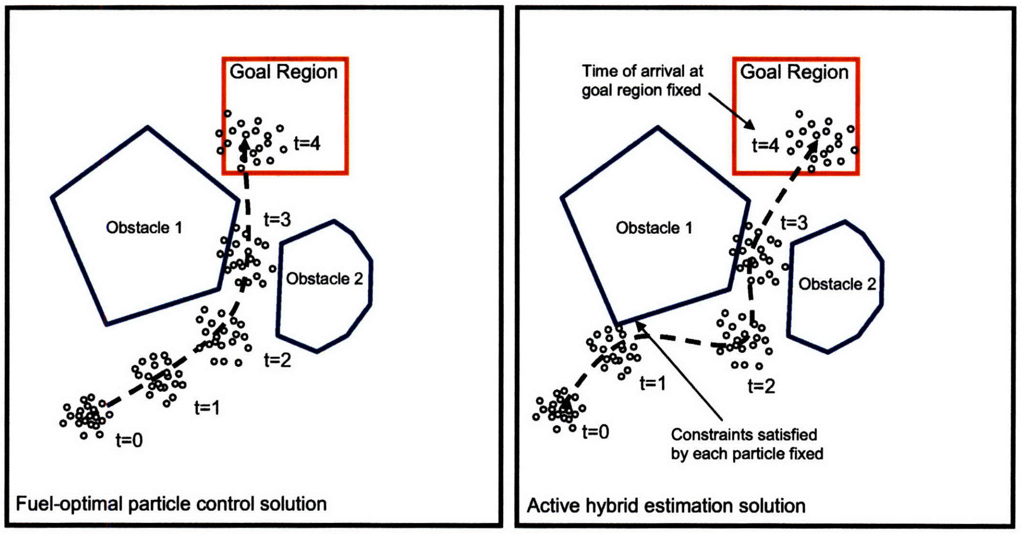

Goal Region
t=4
Obstacle 1
t=3
Obstacle 2
t=2
t=1
t=0
Fuel-optimal particle control solution
Time of arrival at goal region fixed
Goal Region
t=4
Obstacle 1
t=3
Obstacle 2
t=1
t=2
t=0
Constraints satisfied by each particle fixed
Active hybrid estimation solution

Figure 1-6. Left: Initial solution satisfying chance-constrained Qualitative State Plan, generated using the PC-QSP algorithm. Exactly 10% of the particles fail. **Right:** Active hybrid estimation solution, optimized with regard to the probability of estimator error. The approximated chance constraints are satisfied, since at most 10% of the particles fail. The times of events, the particles that succeed, and the constraints satisfied by each successful particle are the same as for the initial solution.

In the case of linear systems with only continuous dynamics, previous work ([47, 27]) developed methods to determine the Maximum-Likelihood (ML) model parameters using an Expectation-Maximization(EM)[34] approach. This approach guarantees convergence to a local maximum of the likelihood function. More recent work extended this approach to Jump Markov Linear Systems(JMLS) [48, 9]. These systems have linear continuous dynamics and Markovian discrete transitions; in this case the discrete dynamics are independent of the continuous state. [48] used an 'approximate' EM approach to learn the parameters of JMLS. Due to the approximation introduced in the Expectation Step (E-Step), this approach does *not* have guaranteed convergence. [9] used an approach inspired by EM to guarantee convergence; this approach iterates between calculating the maximum likelihood discrete model and the maximum likelihood continuous model. This is analogous to 'hard' EM, where instead of using distributions over the unknown variables, only the most likely values are used.

In this thesis we present a new approach to hybrid model learning that extends this work in three ways. First, we consider systems where transitions in the discrete state *do* depend on the continuous state; we call these *autonomous* mode transitions. Second, we extend the work of [48, 9] to learn explicitly the dependence of control inputs on the system dynamics. Finally, we consider 'soft' EM, where a distribution over the hidden variables is used, rather than just the most likely values. The new algorithm is an extension of the approximate EM approach used by [48], which we refer to as the HML algorithm. This approach works by maximizing successive lower bounds to the likelihood function. Intuitively, this means calculating an estimate of the hidden hybrid state sequence given the current guess of the model parameters, then maximizing the likelihood of the observations given the model parameters and the hidden state sequence estimate, as illustrated in Figure 1-7. With sufficient computational resources, this is guaranteed to converge to a local maximum of the likelihood function; we show empirical convergence with limited computational resources using a simulated Mars Rover drivetrain subsystem.

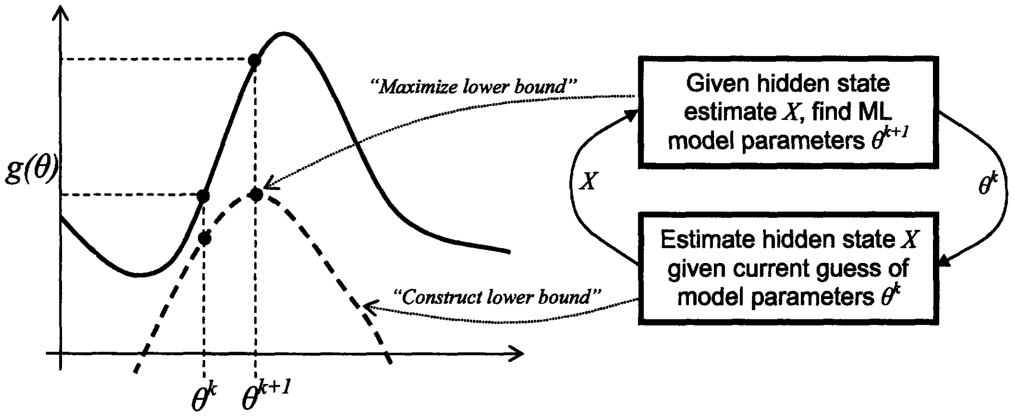

$g(\theta)$
"Maximize lower bound"
"Construct lower bound"
θ^k
θ^{k+1}
Given hidden state estimate X, find ML model parameters θ^{k+1}
Estimate hidden state X given current guess of model parameters θ^k
X
θ^k

Figure 1-7. Approximate Expectation Maximization for hybrid model learning. The objective is to maximize $g(\theta)$, the likelihood of the data given the model. At each iteration the algorithm estimates the distribution of the hidden state sequence given the current guess for the model parameters θ^k; this creates a lower bound on the likelihood function. Then this lower bound is maximized with regard to the model parameters θ to give the next guess for the parameters, denoted θ^{k+1}.

1.4 Summary of Contributions and Conclusions

In this section we give a summary of the contributions of this thesis, and the major conclusions drawn.

1. Chance-Constrained Particle Control

    (a) *Chance-constrained planning problems can be approximated through sampling.* We show that finite-horizon chance-constrained planning problems involving discrete-time dynamics (which may be nonlinear and/or switching) with continuous-valued control inputs, for which all stochastic variables are independent of the control input and system state, can be approximated by sampling all random variables before searching for the optimal plan. Sample-based approximations of event probabilities (over which chance constraints are defined), the expected state and the expected objective function, all converge in probability to their true values as the number of samples tends to infinity.

    (b) *For certain classes of the chance-constrained planning problem, sample-based approximations can be solved using Mixed Integer Linear Programming.* We show that if we restrict our attention to the case of 1) control of Jump Markov Linear Systems, 2) piecewise-linear objective functions, and 3) polytopic feasible regions, the chance-constrained planning problem, approximated using sampling, can be encoded as a Mixed Integer Linear Program. This enables the approximated problem to be solved to global optimality in finite time.

(c) *Chance-constrained planning problems can be approximated through sampling from proposal distributions.* We show that instead of sampling from the true distribution of stochastic variables, the chance-constrained planning problem can be approximated by sampling from an alternative, *proposal* distribution and calculating an analytic *importance weight* for each of the particles. Approximations of event probabilities, expected state and the expected objective function, in terms of importance-weighted samples, all converge in probability to their true values as the number of samples tends to infinity.

(d) *For certain classes of the chance-constrained planning problem, approximations made using importance-weighted samples can be solved using Mixed Integer Linear Programming.* We show that if we restrict our attention to the case of 1) control of Jump Markov Linear Systems, 2) piecewise-linear objective functions, and 3) polytopic feasible regions, the chance-constrained planning problem approximated using *importance-weighted samples* can be encoded as a Mixed Integer Linear Program. This enables the approximated problem to be solved to global optimality in finite time.

(e) *Solution times.* We show empirically that the sample-based approximation of control within a convex region can be solved with full-size linearized models of air and sea vehicles in under a minute with over 100 samples. Non-convex feasible regions greatly increase the complexity of the planning problem and take many minutes to solve similar-sized problems.

(f) *Approximation Accuracy.* We show empirically that for representative vehicle control scenarios, with 100 particles or above, the true probability of mission failure is, on average, within 5% of the specified value, however the standard deviation is relatively high, at around 25% of the specified value.

(g) *Effect of Importance Weighting.* We show empirically with a brake-failure scenario that the use of importance weighting along with a proposal distribution designed for low-probability failure events enables an orders-of-magnitude improvement in approximation error, measured in terms of the difference between the true probability of failure and the specified probability of failure.

2. Active Hybrid Estimation.

The following results assume that all random distributions are Gaussian.

(a) *A closed-form upper bound exists on the probability of error, when performing Bayes-optimal selection between multiple dynamic models.* This result assumes that the dynamic models are linear. The bound requires evaluating a number of terms quadratic in the number of models.

(b) *A finite sequence of control inputs can be found that optimizes the bound on model selection error probability subject to expected state constraints.* The resulting planning problem can be solved using Sequential Quadratic Programming to local optimality.

(c) *Reduction in probability of model selection error.* We show, using a simulated aircraft scenario, that minimization of the bound on model selection error reduces the true model selection error by several orders of magnitude when compared to a typical manually-generated sequence and a power-limited auxiliary signal.

(d) *A closed-form upper bound exists on the probability of K-best hybrid estimation losing the true mode sequence.* This result assumes that we are estimating the state of a Jump Markov Linear System.

(e) *A finite sequence of control inputs can be found that optimizes an upper bound on the probability of losing the true mode sequence, subject to expected state constraints.* We show that we can avoid evaluating an exponential number of terms while still minimizing a (looser) upper bound on the probability of losing the true mode sequence.

(f) *Reduction in probability of losing true mode sequence.* We show, using a simulated aircraft scenario, that minimization of the bound on the probability of losing the true mode sequence reduces the true probability of loss by a factor of 2 when compared to a typical fuel-optimal control sequence.

(g) *A finite sequence of control inputs can be found that optimizes an upper bound on the probability of losing the true mode sequence, while ensuring that an approximated chance-constrained temporally-flexible state plan is satisfied.* We use a sample-based approximation to the chance-constrained state plan, and show that the approximation becomes exact as the number of particles converges to infinity.

3. Hybrid Model Learning

(a) *The Maximum-Likelihood parameters of a class of hybrid systems with autonomous mode transitions can be found using Expectation Maximization.* Specifically this refers to Linear Probabilistic Hybrid Automata (LPHA), which have linear continuous dynamics, and piecewise constant dependence of mode transition probabilities on the continuous state. The EM approach guarantees convergence to a local maximum of the likelihood function, but is intractable in practice as it is exponential in the length of the observation sequence.

(b) *Approximate Expectation Maximization can be performed to learn LPHA parameters.* This approach calculates an approximate distribution over the hidden hybrid state, and is tractable while not guaranteeing convergence. We show convergence empirically on a Mars rover example.

(c) *The number of tracked sequences does not affect learning accuracy.* We show empirically that the number of tracked sequences does not affect the accuracy of the learned hybrid model, measured in terms of the maximum a posteriori mode estimation error.

Chapter 2

Problem Statement

In this thesis we are concerned with the problem of robust execution of Qualitative State Plans using a hybrid stochastic plant model. In Section 2.1 we define a stochastic hybrid plant model and in Section 2.2 we define a Qualitative State Plan. In Section 2.3 we define the problem of robust execution for Qualitative State Plans.

2.1 Stochastic Hybrid Plant Model Definition

A probabilistic plant model represents the behavior of a dynamic system, or 'plant', in terms of stochastic relationships. The model defines the evolution of the system state using probability distributions. These probability distributions are convenient for modeling dynamic systems that are uncertain. In the case of an AUV, stochastic processes are realistic models for wind disturbances. The probabilistic plant model for the AUV defines the evolution of the AUV state incorporating these stochastic disturbances.

Probabilistic *hybrid* models are models whose system state includes both continuous (real-valued) and discrete (set-valued) components. Probabilistic hybrid models date back to the 1970s [3] and are useful in many applications, including visual tracking [104] and fault diagnosis [61, 94]. In this section, we review a formalism for modeling probabilistic hybrid systems, known as Concurrent Probabilistic Hybrid Automata.

Prior authors developed Concurrent Probabilistic Hybrid Automata (CPHA) [62], a formalism for modeling engineered systems that consist of a large number of concurrently operating components with non-linear dynamics. 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.

Definition 1. *We use subscript notation to denote the value of a variable at a given time, e.g.* $\mathbf{x}_{c,\tau}$ *is the value of* $\mathbf{x}_c$ *at time step* τ*, where* τ *is an integer. We denote the time at which time step* τ *occurs as* t_τ*. We assume a fixed time interval* Δt*, hence* $t_\tau = t_0 + \tau \cdot \Delta t$*. We use* $\mathbf{x}_{c,0:N}$ *to denote the finite sequence* $\langle \mathbf{x}_{c,0}, \ldots, \mathbf{x}_{c,N} \rangle$*. The infinite sequence* $\langle \mathbf{x}_{c,0}, \mathbf{x}_{c,1} \ldots \rangle$ *is denoted* $\mathbf{x}_{c,0:\infty}$*. The realization of a random variable* $\mathbf{x}$ *is denoted using the upper case* X*. A sample drawn from a random variable* $\mathbf{x}$ *is denoted* $\mathbf{x}^{(i)}$ *where* i *is the sample number.*

We now formally define a Probabilistic Hybrid Automaton.

Definition 2. *A* ***Probabilistic Hybrid Automaton*** *is a tuple* $\langle \mathbf{x}, \mathbf{w}, \nu, \omega, \mathcal{D}, V, \mathcal{X}_0, \mathcal{X}_d, \mathcal{X}_c, \mathcal{U}_d, \mathcal{U}_c, \delta \rangle$*:*

- $\mathbf{x}$ denotes the *hybrid state* of the automaton. $\mathbf{x} \triangleq \mathbf{x}_d \cup \mathbf{x}_c$ where $\mathbf{x}_d$ denotes the discrete state variables $\mathbf{x}_d \in \mathcal{X}_d$ and $\mathbf{x}_c$ denotes the continuous state variables $\mathbf{x}_c \in \mathcal{X}_c \subseteq \mathbb{R}^{n_x}$. We use $|\mathcal{X}_d|$ to denote the number of possible discrete states and n_x to denote the size of the continuous state vector. Discrete state are also called *modes.*

- The set of *I/O variables* $\mathbf{w} = \mathbf{u}_d \cup \mathbf{u}_c \cup \mathbf{y}$ of the automaton is composed of disjoint sets of discrete input variables $\mathbf{u}_d \in \mathcal{U}_d$, continuous input variables $\mathbf{u}_c \in \mathcal{U}_c \subseteq \mathbb{R}^{n_u}$, and continuous output variables $\mathbf{y} \in \mathcal{Y}_c \subseteq \mathbb{R}^{n_y}$. In this thesis we consider only continuous input variables. For notational simplicity we define $\mathbf{u} \triangleq \mathbf{u}_c$ for the rest of the thesis.

- $\nu \in \mathbb{R}^{n_x}$ and $\omega \in \mathbb{R}^{n_y}$ are continuous-valued stochastic disturbance and observation noise processes, respectively. These processes have known, but arbitrary probability distributions.

- $\mathcal{D} : \mathcal{X}_d \to F_{DE}$ specifies the *continuous evolution* of the automaton for each discrete mode, in terms of the set of discrete-time difference equations F_{DE} over the variables $\mathbf{x}_c$, $\mathbf{u}$, $\mathbf{y}$, ν and ω. In the general case, these difference equations can take the nonlinear, time-varying form:

$$\mathcal{D}(\mathbf{x}_d) \triangleq \left\langle \begin{array}{c} \mathbf{x}_{c,\tau+1} = f(\tau, \mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}, \mathbf{u}_\tau, \nu_\tau, \omega_\tau) \\ \mathbf{y}_{\tau+1} = g(\tau, \mathbf{x}_{c,\tau+1}, \mathbf{x}_{d,\tau}, \mathbf{u}_\tau, \nu_\tau, \omega_\tau) \end{array} \right\rangle. \tag{2.1}$$

  In Chapters 3 through 5 we restrict our attention to special cases of these dynamics, for example those where the equations $f(\cdot)$ and $g(\cdot)$ are linear.

- V specifies the probabilistic discrete evolution of the automaton. V is a finite set of *transition specifications* $\mathcal{C}_i \triangleq \langle g_i, \mathcal{T}_i \rangle$. Each *transition specification* $\mathcal{C}_i$ has an associated Boolean *guard* $g_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathcal{U}_d \to \{true, false\}$ and a *transition probability matrix* $\mathcal{T}_i \in \mathbb{R}^{|\mathcal{X}_d| \times |\mathcal{X}_d|}$. We assume that the guards are disjoint and collectively exhaustive. Each matrix $\mathcal{T}_i$ defines the probability of a transition from any mode to any other mode given that guard condition i is satisfied, such that $\mathcal{T}_i(m,n) = p(\mathbf{x}_{d,t+1} = m | \mathbf{x}_{d,t} = n, g_i = \text{true})$. Here $\mathcal{T}_i(m,n)$ denotes the (m,n)'th element of the matrix $\mathcal{T}_i$. The elements of each column in $\mathcal{T}_i$ must sum to 1.

- The initial state distribution of the automaton is specified as $\mathcal{X}_0 \triangleq p(\mathbf{x}_0)$.

Remark 2.1. *Note that for a PHA, in general, the set of transition specifications V collectively specify the distribution $p(\mathbf{x}_{d,\tau+1}|\mathbf{x}_{d,\tau}, \mathbf{x}_{c,\tau}, \mathbf{u}_{d,\tau}, \mathbf{u}_\tau)$. Each transition tuple defines this distribution as having a constant value in the regions satisfied by the guard c. The transition specifications can thus specify conditional distributions $p(\mathbf{x}_{d,\tau+1}|\mathbf{x}_{d,\tau}, \mathbf{x}_{c,\tau}, \mathbf{u}_{d,\tau}, \mathbf{u}_\tau)$ that are piecewise constant in the continuous state $\mathbf{x}_{c,\tau}$ and the continuous control input $\mathbf{u}_\tau$. This is illustrated in Figure 2-3.*

The conditional dependencies between the variables of a PHA are shown in Bayes Net form in Figure 2-1. An example of a stochastic hybrid plant and the corresponding probabilistic hybrid model is shown in Figure 2-2. The plant is the actuator subsystem of an AUV tailcone. The angle of the tailcone is controlled in order to control the motion of the AUV. Due to an intermittent fault the actuator is prone to 'stick' at random. In the probabilistic hybrid model, the discrete state $\mathbf{x}_d$ consists of a single variable that indicates whether the actuator is operating normally, or is stuck. In other words, $\mathbf{x}_d \in \mathcal{X}_d \triangleq \langle \texttt{nominal}, \texttt{stuck} \rangle$. The continuous state $\mathbf{x}_c$ consists of the motor current i and the actuator angular velocity $\dot{\theta}$. The continuous input variables $\mathbf{u}$ consist of the voltage V applied to the motor drive system. There are no discrete input variables. The continuous output variables $\mathbf{y}$ consist of noisy observations of the angular velocity of the actuator made through the encoder, which we denote $\dot{\theta}_{obs}$. The observation noise at the encoder is modeled using the noise process ν. The process noise ω models local variations in the slope and roughness of the terrain. The continuous evolution of the system state is defined using linear discrete-time difference equations, such that:

$$\begin{aligned}\mathbf{x}_{c,\tau+1} &= A(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau} + B(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \omega_\tau \\ \mathbf{y}_{\tau+1} &= C(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau+1} + D(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \nu_\tau.\end{aligned} \tag{2.2}$$

These equations are derived through time-discretization of the differential equations governing the electrical and mechanical dynamics of the system. The matrices $\langle A(\mathbf{x}_{d,\tau}), B(\mathbf{x}_{d,\tau}), C(\mathbf{x}_{d,\tau}), D(\mathbf{x}_{d,\tau})\rangle$ are defined so that for $\mathbf{x}_{d,\tau} = \texttt{stuck}$, the mechanical damping of the actuator is significantly greater than for $\mathbf{x}_{d,\tau} = \texttt{nominal}$. The numerical values of the matrices in (2.2) are given in Chapter 5.

The actuator has a different probability of sticking depending on which direction it is driving. This behavior is modeled using two transition specifications, with guards defined over the continuous velocity $\dot{\theta}$ as follows:

$$g_1 = \begin{cases} \text{true} & \dot{\theta} \in [-\infty, 0] \\ \text{false} & \dot{\theta} \in [0, \infty] \end{cases} \quad g_2 = \begin{cases} \text{true} & \dot{\theta} \in [0, \infty] \\ \text{false} & \dot{\theta} \in [-\infty, 0] \end{cases}. \tag{2.3}$$

The transition probability matrices are defined so that $\mathcal{T}_2$ gives a greater probability of transition to

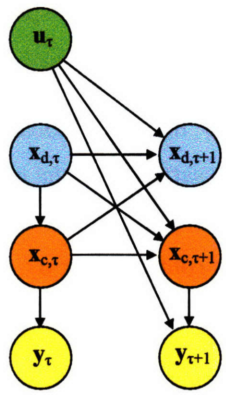
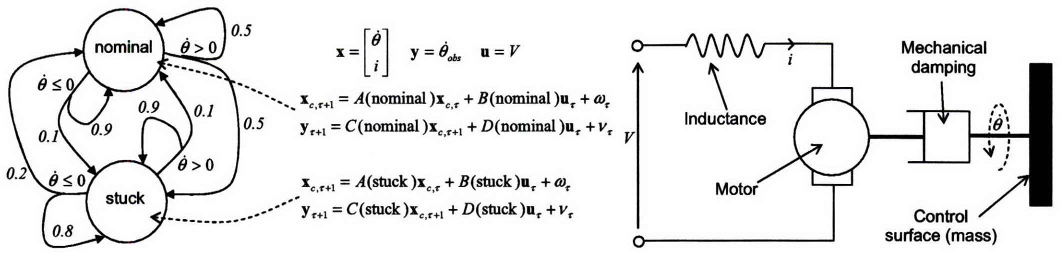


Figure 2-1. Bayes Network showing conditional dependencies between variables at time τ and $\tau+1$ in a Probabilistic Hybrid Automaton. The noise variables ω and ν are not shown.



nominal
stuck
$\dot\theta > 0$
$\dot\theta \le 0$
$\dot\theta \le 0$
$\dot\theta > 0$
0.5
0.5
0.9
0.1
0.1
0.9
0.2
0.8
$\mathbf{x} = \begin{bmatrix} \dot\theta \\ i \end{bmatrix}$ $\mathbf{y} = \dot\theta_{obs}$ $\mathbf{u} = V$
$\mathbf{x}_{c,\tau+1} = A(\text{nominal})\mathbf{x}_{c,\tau} + B(\text{nominal})\mathbf{u}_\tau + \omega_\tau$
$\mathbf{y}_{\tau+1} = C(\text{nominal})\mathbf{x}_{c,\tau+1} + D(\text{nominal})\mathbf{u}_\tau + \nu_\tau$
$\mathbf{x}_{c,\tau+1} = A(\text{stuck})\mathbf{x}_{c,\tau} + B(\text{stuck})\mathbf{u}_\tau + \omega_\tau$
$\mathbf{y}_{\tau+1} = C(\text{stuck})\mathbf{x}_{c,\tau+1} + D(\text{stuck})\mathbf{u}_\tau + \nu_\tau$
V
i
Inductance
Motor
Mechanical damping
$\dot\theta$
Control surface (mass)

Figure 2-2. Left: Probabilistic Hybrid Automaton used to model the motor drive subsystem. **Right:** Schematic of motor drive subsystem.

and from the stuck mode than $\mathcal{T}_2$. For numerical values, refer to Chapter 5. Initially, we assume that the actuator is approximately at rest and is not stuck. This is modeled by specifying the initial state distribution $\mathcal{X}_0$ such that $p(\mathbf{x}_{d,0} = \texttt{nominal}) = 1$ and $p(\mathbf{x}_{d,0} = \texttt{stuck}) = 0$. The continuous distribution $p(\mathbf{x}_{c,0}|p(\mathbf{x}_{d,0}))$ is independent of $\mathbf{x}_{d,0}$ in this case, and is specified to be a Gaussian with mean $[0\ 0]'$ and small covariance.

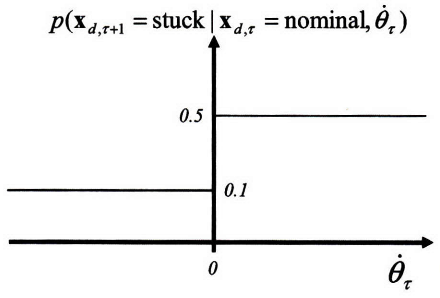


Figure 2-3. The transition specifications V for a PHA collectively specify the distribution $p(\mathbf{x}_{d,\tau+1}|\mathbf{x}_{d,\tau}, \mathbf{x}_{c,\tau}, \mathbf{u}_{d,\tau}, \mathbf{u}_{c,\tau})$. Shown here is the distribution $p(\mathbf{x}_{d,\tau+1} = \text{stuck}|\mathbf{x}_{d,\tau} = \text{nominal}, \mathbf{x}_{c,\tau}, \mathbf{u}_{d,\tau}, \mathbf{u}_{c,\tau})$ specified by the actuator Probabilistic Hybrid Automaton in Figure 2-2. Note that this distribution is piecewise constant in the continuous state $\mathbf{x}_{c,\tau}$.

2.2 Qualitative State Plan Definition

In this section we review Qualitative State Plans, first described in [76] and later in [63] A Qualitative State Plan is a temporally flexible plan [4], with *state activities* (also called *episodes* [57]) that specify qualitative constraints on the state trajectory of the plant. Consistent with previous approaches to model-based execution[133], a human operator or higher level planner uses a Qualitative State Plan to specify the desired states the plant should be in. This elevates the level of interaction between the operator and the plant, allowing the operator to focus on the goals to be achieved, rather than how to achieve those goals. In addition, the flexibility in the plan in terms of time and state constraints give extra latitude to the autonomous controller, which we use to add robustness and to aid state estimation. We provide a definition of Qualitative State Plans in Section 2.2.1 along with an example.

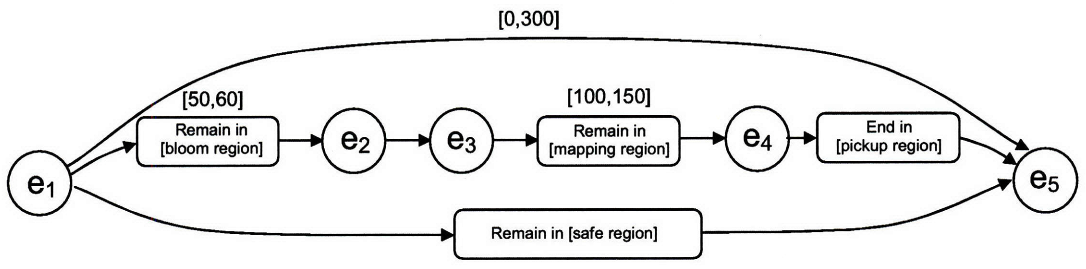
In the general case, episodes in a Qualitative State Plan are flexible in both time and space. This is a generalization of a *configuration goal sequence* (CGS)[134]; each configuration goal in a CGS specifies constraints on the system state, but is fixed in time. In this thesis we first develop algorithms for execution of Configuration Goal Sequences, and then extend these approaches to execution of general Qualitative State Plans.

To the definition of Qualitative State Plans given by [76], we add *chance constraints* and *expected state constraints.* Chance constraints impose constraints on the required probability of success of sets of episodes within the plan. An operator uses these constraints to specify required levels of reliability in achieving success. These constraints are a powerful way of allowing the operator to deal with uncertainty for two principal reasons; first, by adjusting the specified probabilities, conservatism can be traded against performance; a plan with very low failure probabilities will typically be expensive in terms of fuel or time; second, the operator can express the relative importance of achieving different portions of the state plan in a meaningful manner. Expected state constraints ensure that sets of episodes succeed for the expected system state.[1] These are useful when the operator needs to specify the average behavior of the system; we elaborate on this in Section 2.2.6.

2.2.1 Overall Definition of a *Qualitative State Plan*

Definition 3. *A* ***Qualitative State Plan*** $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$ *specifies a desired evolution of the plant state over time, and is defined by a set* $\mathcal{E}$ *of discrete events, a set* $\mathcal{A}$ *of* ***episodes*** *, imposing constraints on the plant state evolution, a set* $\mathcal{C}$ *of* ***temporal constraints*** *(Def. 8) between events, a set* $\mathcal{G}_c$ *of* ***chance constraints*** *(Def. 11) that specify reliability constraints on the success of sets of episodes in the plan, a set* $\mathcal{G}_m$ *of* ***expected state constraints*** *(Def. 13) that guarantee the success of episodes for the most likely system state, and an* ***objective function*** F *(Def. 15), which must*

[1]For unimodel, symmetric distributions, such as Gaussians, the expected system state is also the most likely state.



$p($ End in [goal region] $fails$ OR Remain in [safe region] $fails$ $) < 0.01$

$p($ Remain in [bloom region] $fails$ OR Remain in [mapping region] $fails$ $) < 0.1$

Figure 2-4. Qualitative State Plan for AUV science mission. The mission involves collecting data from an algal bloom and mapping the sea floor before being picked up by a surface vessel.

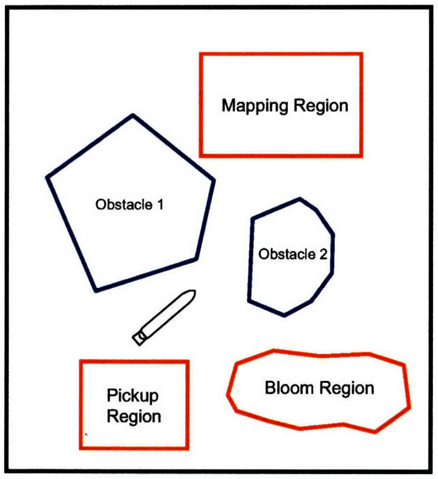


Figure 2-5. Illustration of AUV science mission described in Figure 2-4

be minimized.

We illustrate a Qualitative State Plan diagrammatically by an acyclic directed graph in which the events in $\mathcal{E}$ are represented by nodes, drawn as circles, and the episodes as arcs with ovals. A simple example of a Qualitative State Plan is shown in Figure 2-4. This plan describes an AUV science mission, depicted in Figure 2-5 , which can be stated informally as:

> Reach the algal bloom region and remain there for between 50 and 60 time units. After that, reach the mapping region and remain there for between 100 and 150 time units. End the mission in the pickup region in order to rendezvous with the surface vessel. At all times remain in the safe region by avoiding the obstacles. The entire mission must take at most 300 time units.

2.2.2 Definition of an *Episode*

Def. 3 defined a qualitative state plan P as a tuple $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$, where $\mathcal{A}$ is a set of *episodes* that describe the desired evolution of the plant state throughout the execution of the state plan. In this section, we formally define an episode in terms of a time interval over which the episode imposes a given *state constraint* on the plant.

Definition 4. *An **event** e is a real-valued variable that stands for either the start (denoted e_S) or end (denoted e_E) time of an episode a.*

Definition 5. *An **episode** $a = \langle e_S, e_E, c_S \rangle$ has an associated start event e_S and an end event e_E. c_S is called a **state constraint** on the hidden state $\mathbf{x}$ and the I/O variables w. In the general case, a state constraint is a constraint involving both state and time. A state constraint can be thus described by a time-varying relation, such as a* flow-tube*[63]. In this thesis, however, we restrict our attention to state constraints that take on one of the following four forms, where R_S, R_E, $R_\forall$ and $R_\exists$ are regions in the joint state and I/O space $\mathcal{X}_c \cup \mathcal{X}_d \cup \mathcal{U}_d \cup \mathcal{U}_c \cup \mathcal{Y}_c$, and T is a schedule (Def. 9) for P:*

1. Start in state region R_S: $\langle \mathbf{x}_\tau, \mathbf{w}_\tau \rangle \in R_S \;\; t_\tau = T(e_S)$;
2. End in state region R_E: $\langle \mathbf{x}_\tau, \mathbf{w}_\tau \rangle \in R_E \;\; t_\tau = T(e_E)$;
3. Remain in state region $R_\forall$: $\langle \mathbf{x}_\tau, \mathbf{w}_\tau \rangle \in R_\forall \;\; \forall t_\tau \in [T(e_S), T(e_E)]$;
4. Go through state region $R_\exists$: $\langle \mathbf{x}_\tau, \mathbf{w}_\tau \rangle \in R_\exists \;\; \exists t_\tau \in [T(e_S), T(e_E)]$.

Since we use a discrete-time system model, we do not constrain the system state between time intervals t_τ and $t_{\tau+1}$.

An example of an episode is the *Remain in* `[bloom region]` episode shown in Figure 2-5. This episode has event e_1 as its start event and e_2 as its end event. The state constraint for the episode requires the AUV position to remain in the algal bloom region for all time steps between $T(e_1)$ and $T(e_2)$.

As can be seen in Def. 5, there are two main types of state constraints in a qualitative state plan: *durative* state constraints (*remain in*) and *instantaneous* state constraints (*start in, end in* and *go through*). As shown in [76], all instantaneous state constraints can be expressed using an *end in* episode such that we can consider only *remain in* and *end in* episodes without loss of expressivity.

Definition 6. *An episode $a = \langle e_S, e_E, c_S \rangle$ **succeeds** for system state sequence $X_{0:f}$ and schedule T containing times $T(e_S)$ and $T(e_E)$ if the associated state constraint c_S is satisfied. Otherwise the episode **fails**. A set of episodes $\mathcal{A}$ succeeds if all episodes $a \in \mathcal{A}$ succeed. A set of episodes $\mathcal{A}$ fails if any episode $a \in \mathcal{A}$ fails.*

Definition 7. *We define the* ***success indicator function*** *s for a set of episodes $\mathcal{A}$ as:*

$$s(\mathcal{A}, X_{0:f}, T) = \begin{cases} 0 & \textit{All episodes } a \in \mathcal{A} \textit{ succeed for state sequence } X_{0:f} \textit{ and schedule } T \\ 1 & \textit{There exists an episode } a \in \mathcal{A} \textit{ that fails for state sequence } X_{0:f} \textit{ and schedule } T. \end{cases} \tag{2.4}$$

2.2.3 Definition of a *Simple Temporal Constraint*

In a qualitative state plan, episodes are composed together in order to describe valid plant state trajectories over time. This composition is achieved through the use of *simple temporal constraints* (Def. 8). As with the state constraints, these temporal constraints are qualitative, in that they partially constrain the time of each event, rather than specifying a fixed value on the times at which each event must be scheduled. Simple temporal constraints specify lower and upper bounds on the duration between pairs of events in the qualitative state plan; this is a special case of the more general *temporal constraint* defined by [33, 32].

Definition 8. *A* ***simple temporal constraint*** *$c = \langle e_A, e_B, \Delta T^{min}_{e_A \to e_B}, \Delta T^{max}_{e_A \to e_B} \rangle$ specifies that the duration from an event e_A to an event e_B be in the real-valued interval $[\Delta T^{min}_{e_A \to e_B}, \Delta T^{max}_{e_A \to e_B}] \subseteq [0, +\infty]$.*

Temporal constraints are represented diagrammatically by arcs between nodes, labeled with the time bounds $[\Delta T^{min}_{e_A \to e_B}, \Delta T^{max}_{e_A \to e_B}]$. Note that temporal constraints can be between any two nodes. In the case that these nodes are the start and end event of a single episode the temporal constraint places bounds on the duration of the episode . An example of this in the AUV scenario is the constraint between event e_1 and e_2, which constrains the *Remain in* `[bloom region]` episode to last between 50 and 60 time units. An example where the temporal constraint is not associated with a single episode is the constraint between e_1 and e_5; this constrains the entire plan to last between 0 and 300 time units.

2.2.4 Definition of a *Schedule*

A fundamental feature of Qualitative State Plans is that the times at which each event in the plan must be executed are specified in a flexible manner, rather than strictly scheduled beforehand. This motivates the definition of a *schedule* for a qualitative state plan P.

Definition 9. *A* ***schedule*** *T for a Qualitative State Plan P is an assignment $T : \mathcal{E} \mapsto \mathbb{R}$ of execution times to all the events in P.*

This is the same definition as that of a schedule in temporal planning[33]. Note that although in Def. 2 we describe the dynamics of the Probabilistic Hybrid Automaton in terms of discrete-time stochastic difference equations, here a schedule is not restricted to take on values from this time discretization; it can take on any real values. This is because the temporal constraints are defined in terms of real-valued constraints. Note also that a schedule is a *deterministic* assignment of values rather than a stochastic distribution.

Definition 10. *Given a schedule T and a set of temporal constraints C, T is* ***temporally consistent*** *if it satisfies all of the constraints $c \in C$.*

An example of a temporally consistent schedule for the AUV scenario is as follows:

$$T(e_1) = 0 \quad T(e_2) = 50 \quad T(e_3) = 50 \quad T(e_4) = 150 \quad T(e_5) = 150. \tag{2.5}$$

Although this schedule is temporally consistent, it is a poor choice for the AUV mission since it allows no time for the AUV to travel between different regions. For example, according to this schedule immediately before time $t = 150$ the AUV must be in the mapping region, but at time $t = 150$ it must be in the pickup region. Instead we need a temporally consistent schedule that can be achieved by the AUV. We call this a schedule that *robustly satisfies* a Qualitative State Plan, and in Section 2.3 we define formally robust satisfaction of a Qualitative State Plan.

2.2.5 Definition of a *Chance Constraint*

For a stochastic system, the state $\mathbf{x}$ is a random variable. The success of a set of episodes $\mathcal{A}$ is hence a random event. We use *chance constraints* to specify a maximum allowed probability of failure for a particular set of episodes .

Definition 11. *Each* ***chance constraint*** *$c_c \in \mathcal{G}_c$ takes on one of the following two forms:*

1. *Success constraint: The set of episodes $\mathcal{A}(c_c)$ must succeed (Def. 6) with probability at least $\gamma(c_c)$:*
$$p\Big(s(\mathcal{A}(c_c), \mathbf{x}_{0:f}, T) = 0\Big) \geq \gamma(c_c). \tag{2.6}$$

2. *Failure constraint: The set of episodes $\mathcal{A}(c_c)$ must fail (Def. 6) with probability at most $\delta(c_c)$:*
$$p\Big(s(\mathcal{A}(c_c), \mathbf{x}_{0:f}, T) = 1\Big) \leq \delta(c_c). \tag{2.7}$$

The set $\mathcal{A}(c_c)$ can be defined to include a single episode , or every episode in the state plan. In the latter case, c_c describes a required maximum probability of failure of the entire state plan.

The AUV example in Figure 2-5 has two chance constraints:

1. The probability of both of the episodes *Remain in* `[bloom region]` and *Remain in* `[mapping region]` succeeding is at least 90%.

2. The probability of either of the episodes *Remain in* `[safe region]` and *End in* `[pickup region]` failing is at most 0.1%.

Intuitively, these chance constraints ensure that the probability of occurrences that would jeopardize the safe return of the AUV is very low, while the probability of the science mission succeeding is relatively high.

Remark 2.2. *A success constraint can be trivially rewritten as an equivalent failure constraint; we will therefore only consider failure constraints in the rest of the development.*

Definition 12. *A failure chance constraint c_c is* **satisfied** *if and only if:*

$$p\Big(s(\mathcal{A}(c_c), \mathbf{x}_{0:f}, T) = 1\Big) \leq \delta(c_c). \tag{2.8}$$

Remark 2.3. *In typical scenarios, the success of individual episodes are* not *independent events, nor are they mutually exclusive events. Hence the probability of all episodes succeeding is* not *a simple product or sum of the individual episode success probabilities.*

For example, consider two episodes a_A and a_B scheduled one after the other, which require the AUV to be in locations A and B, respectively, where A and B are close to each other (and may overlap). Intuitively, a large disturbance that prevents the AUV from reaching location A on schedule is also likely to prevent the AUV from reaching location B on schedule. Hence failure of a_A and a_B are not independent events.

2.2.6 Definition of an *Expected State Constraint*

A Qualitative State Plan specifies constraints on the expected system state. The expected system state sequence $E[\mathbf{x}_{0:f}]$ is not a random variable, hence the Qualitative State Plan allows the user (or a higher level planner) to *require* the success of a set of episodes for the expected state.

Definition 13. *An* **expected state constraint**, *denoted c_m, requires the set of episodes $\mathcal{A}(c_m)$ to succeed for the expected system state sequence $E[\mathbf{x}_{0:f}]$ and schedule T, in other words:*

$$s(\mathcal{A}(c_m), E[\mathbf{x}_{0:f}], T) = 0 \tag{2.9}$$

Definition 14. *An expected state constraint c_m is* **satisfied** *by the state trajectory distribution $\mathbf{x}_{0:f}$ and schedule T if and only if:*

$$s(\mathcal{A}(c_m), E[\mathbf{x}_{0:f}], T) = 0 \tag{2.10}$$

Expected state constraints are useful when the state constraint c_S has a feasible region that is a point. For example, we might want the AUV to end its mission with zero velocity. The probability of a continuous value taking a single value is zero, hence it is not meaningful to constrain the probability of achieving zero velocity with a chance constraint. One approach would be to define an arbitrary region around zero and constrain the probability of the velocity falling within that region. This is not practical, since an unfortunate choice of the region leads to unintended consequences. If the region is large compared to the state uncertainty, the optimal control strategy will exploit this and leave the system state far from the intended zero location. If the region is small compared to the state uncertainty, no control strategy will exist to satisfy the chance constraint. Instead of this chance constrained approach, we use an expected state constraint to ensure that the system state is centered on the desired value.

2.2.7 Definition of an *Objective Function*

As introduced in Def. 3, a Qualitative State Plan describes the desired behavior of the plant in time, both in terms of state and temporal constraints that must be satisfied, and in terms of an *objective function* F whose expected value is minimized. In this section, we formally define the notion of objective function (Def. 15).

Definition 15. *Given sequences of input variables* $\mathbf{u}_{0:f}$, *a sequence of state variables* $X_{0:f}$ *and given a schedule* T *for a set of events* $\mathcal{E}$, *an* ***objective function*** F *is a real-valued function* $F : \mathbb{R}^{n_x \times (f+1)} \times \mathbb{R}^{n_u \times (f+1)} \times \mathbb{R}^{|\mathcal{E}|} \rightarrow \mathbb{R}$.

Since the state sequence $\mathbf{x}_{0:f}$ is a random variable, we minimize the expectation of F over the distribution of $\mathbf{x}_{0:f}$, denoted $E[F]$:

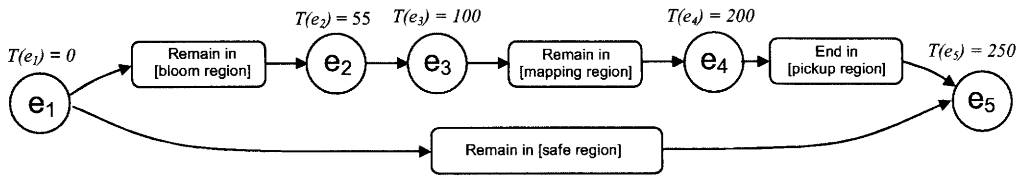
$$E[F] = \int F(\mathbf{u}_{0:f}, \mathbf{x}_{0:f}, T) p(\mathbf{x}_{0:f}) d\mathbf{x}_{0:f}. \tag{2.11}$$

Objective functions that involve the system state include the deviation of the system state from an optimal location. Alternatively, F can represent the amount of fuel required by the control sequence $\mathbf{u}_{0:f}$. In the AUV example we set $F(\mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T) = T(e_5)$. In this case F represents the total plan execution time.

Remark 2.4. *In the case that F does not depend on the system state, the expectation (2.11) need not be calculated, since $E[F] = F$.*

2.2.8 Overall Definition of a *Configuration Goal Sequence*

A Qualitative State Plan for which the schedule of every event is fixed, is known as a Configuration Goal Sequence [134]. Each episode specifies constraints on the system state, but the time points at



$p($End in [goal region]$fails$ OR Remain in [safe region]$fails$ $) < 0.01$

$p($Remain in [bloom region]$fails$ OR Remain in [mapping region]$fails$ $) < 0.1$

Figure 2-6. Configuration Goal Sequence for a simple AUV science mission.

which these constraints apply are fixed.

Definition 16. *A **Configuration Goal Sequence** $P = \langle \mathcal{E}, T, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$ is defined by a set $\mathcal{E}$ of discrete events, a set $\mathcal{A}$ of **episodes** , imposing constraints on the plant state evolution, a schedule T (Def. 9), a set $\mathcal{G}_c$ of chance constraints (Def. 11) that specify reliability constraints on the success of episodes in the plan, a set $\mathcal{G}_m$ of expected state constraints (Def. 13) that guarantee the success of episodes for the expected system state, and an objective function F (Def. 15), which must be minimized.*

Def. 16 is identical to the definition of a Qualitative State Plan (Def. 3) except that, instead of a set of temporal constraints $\mathcal{C}$, the Configuration Goal Sequence has a fixed schedule T. An example of a Configuration Goal Sequence is shown in Figure 2-6. This example was created by taking the AUV Qualitative State Plan in Figure 2-4 and specifying a fixed schedule for each event. Note that in doing so we remove a great deal of flexibility in the plan. For example, we have uniquely specified the time that the AUV must take to travel from the bloom region to the mapping region. This may make the plan take more time or fuel than necessary, or may make the plan more brittle to disturbances than a temporally flexible one.

2.3 Definition of Robust Qualitative State Plan Execution

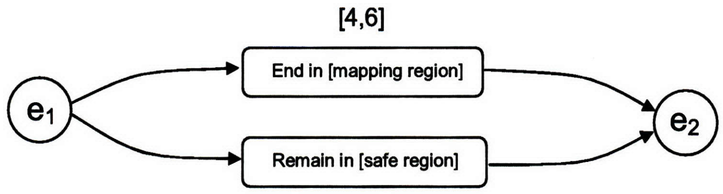
We now define what is meant by *robust satisfaction* of a Qualitative State Plan and define the robust Qualitative State Plan execution problem.

Definition 17. *A Qualitative State Plan $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$ is **robustly satisfied** by a state sequence $\mathbf{x}_{0:f}$ and a schedule T if and only if the following three propositions hold:*

1. *The schedule T is temporally consistent, that is T satisfies all of the constraints in $\mathcal{C}$ (Def. 9);*

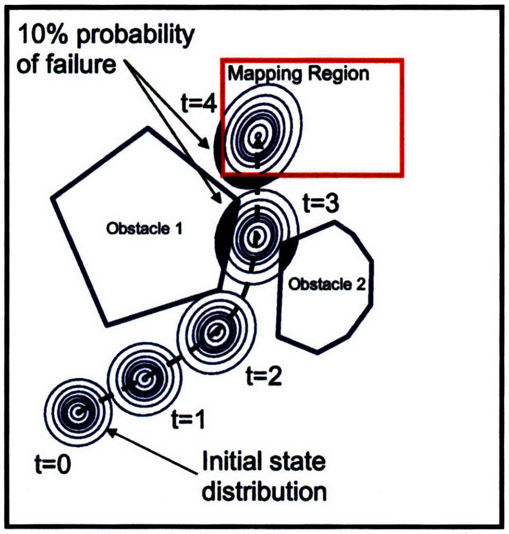
2. *All of the chance constraints in $\mathcal{G}_c$ are satisfied (Def. 12);*

3. *All of the expected state constraints in $\mathcal{G}_m$ are satisfied (Def. 14).*



$p($End in [mapping region]fails OR Remain in [safe region]fails $) < 0.1$

Figure 2-7. Simplified Qualitative State Plan for AUV science mission.



$T(e_1) = 0 \qquad T(e_2) = 4$

Figure 2-8. Solution to simplified Qualitative State Plan (Figure 2-7). This is the optimal robust feasible solution (Def. 23) when the time of event 2 is the objective function to be minimized.

For the sake of illustration, consider the simplified Qualitative State Plan shown in Figure 2-7. Figure 2-8 shows a state sequence (where $\Delta t = 1$) and schedule that robustly satisfies the simplified Qualitative State Plan; the schedule is temporally consistent, since e_2 occurs between 4 and 6 time units after e_1; the single chance constraint is satisfied, since the probability of either activity failing is at most 0.1; and there are no expected state constraints.

Definition 18. *A random state sequence* $\mathbf{x}_{0:f}$ *is* ***consistent*** *with observations* $\mathbf{y}_{0:g}$*, input sequence* $\mathbf{u}_{0:f}$ *and the Probabilistic Hybrid Automaton* $\mathcal{M}$*, where* $g \leq f$*, if the distribution of* $\mathbf{x}_{0:f}$ *is given by* $p(\mathbf{x}_{0:f}|\mathbf{y}_{0:g}, \mathbf{u}_{0:f}, \mathcal{M})$*. The* ***Hybrid Estimation Problem*** *consists of determining the consistent random sequence* $\mathbf{x}_{0:f}$*, that is, determining the distribution* $p(\mathbf{x}_{0:f}|\mathbf{y}_{0:f}, \mathbf{u}_{0:f}, \mathcal{M})$*.*

Note that we allow the sequence of observations to be shorter than that of the control inputs. In this case, the hybrid estimation problem involves an inference portion, where observations are available, and a prediction portion, where observations are unavailable. In the AUV example, at time step τ we have noisy observations of the AUV position from time step 0 to τ. In this case the discrete state includes the mode of the tailcone actuator (`nominal` or `stuck`) and the continuous state includes the position and velocity of the AUV. We are interested in the distribution of the system state from time 0 to time τ, since this tells us about the current location of the aircraft, and whether or not the actuator is stuck. We are also interested in the predicted system state from $\tau + 1$ to $tau + N$. The hybrid estimation problem consists of finding the consistent system state $\mathbf{x}_{0:\tau+N}$; this means both inferring the system state $\mathbf{x}_{0:\tau}$ from the observations $\mathbf{y}_{0:\tau}$, and predicting the system state $\mathbf{x}_{\tau+1:\tau+N}$ without observations.

In this thesis we do not tackle the hybrid estimation problem; prior work has shown that the problem is intractable, in general, but has suggested tractable approximate approaches. These approaches are summarized in Chapter 4. We do show that robust Qualitative State plan Execution can be used to minimize the probability of an approximate hybrid estimator losing track of the hybrid state; this is described in detail in Chapter 4.

Definition 19. *A* ***PHA*** *$\mathcal{M} = \langle \mathbf{x}, \mathbf{w}, \nu, \omega, \mathcal{D}, V, \mathcal{X}_0, \mathcal{X}_d, \mathcal{X}_c, \mathcal{U}_d, \mathcal{U}_c, \delta \rangle$* ***is consistent*** *with the observation sequence $\mathbf{y}_{0:f}$ and the input sequence $\mathbf{u}_{0:f}$ if $\mathcal{M}$ is the Maximum Likelihood (ML) model given $\mathbf{y}_{0:f}$ and $\mathbf{u}_{0:f}$. The Maximum Likelihood model is the one that maximizes $p(\mathbf{y}_{0:f}|\mathbf{u}_{0:f}, \mathcal{M})$. The* ***Hybrid Model Learning Problem*** *consists of determining the consistent PHA $\mathcal{M}$ given $\mathbf{y}_{0:f}$ and $\mathbf{u}_{0:f}$.*

In the AUV example, the dynamics of the AUV may not be *a priori* available, or may be uncertain. In particular, the behavior of the AUV when the aileron actuator is stuck may not be known. Intuitively, the hybrid model learning problem is to determine the hybrid discrete-continuous dynamics of the AUV from the observed data given knowledge the inputs applied to the system.

Definition 20. *A control sequence $\mathbf{u}_{0:f}$* ***satisfies a Probabilistic Hybrid Automaton*** *$\mathcal{M} = \langle \mathbf{x}, \mathbf{w}, \nu, \omega, \mathcal{D}, V, \mathcal{X}_0, \mathcal{X}_d, \mathcal{X}_c, \mathcal{U}_d, \mathcal{U}_c, \delta \rangle$ if and only if all control inputs are within the input space, that is:*

$$\forall \tau = 0, \ldots, f \quad \mathbf{u}_\tau \in \mathcal{U}_c. \tag{2.12}$$

The AUV has hard actuator limits; for example the aileron angle is limited to be in the range $[-10°, 10°]$. In order to satisfy the PHA modeling the AUV, the control inputs specifying desired aileron angle must remain within this range.

Definition 21. *A* ***solution*** *is defined as $\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$. A solution is* ***complete*** *if and only if the span of the state sequence $\mathbf{x}_{0:f}$ and the span of the input sequence $\mathbf{u}_{0:f}$ cover the span of the schedule T. The span of a sequence $\mathbf{x}_{0:f}$ is defined as the time interval $[t_0, t_f]$. The span of a schedule T is defined as the time interval $[T(e_{first}), T(e_{last})]$ where e_{first} is the earliest scheduled event in T and e_{last} is the last scheduled event in T. A span $[a, b]$ covers another span $[c, d]$ if and only if $a \leq c$ and $b \geq d$. An incomplete solution $\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$ is said to be* ***partial***.

Intuitively, a solution is complete if and only if both the state sequence and input sequence start at or before the time at which the start event of the qualitative state plan is scheduled, and end at or after the time at which the end event is scheduled. This way, the two sequences cover the complete Qualitative State Plan execution. The solution sequence shown in Figure 2-8 (the corresponding control sequence $\mathbf{u}_{0:4}$ is not shown) is complete; the span of the sequences $\mathbf{x}_{0:4}$ and $\mathbf{u}_{0:4}$ are both $[0, 4]$, hence covering the span of the schedule, which is $[0, 4]$.

Definition 22. *Given a probabilistic hybrid automaton $\mathcal{M}$, a sequence of observations $\mathbf{y}_{0:\tau}$ and a Qualitative State Plan $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$, a* **robust feasible solution** *$\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$ is one for which the following propositions hold:*

1. *The state sequence distribution $p(\mathbf{x}_{0:f})$ is consistent with the sequence of observations $\mathbf{y}_{0:f}$, the input sequence $\mathbf{u}_{0:f}$ and the hybrid automaton $\mathcal{M}$ (Def. 18).*

2. *$\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$ robustly satisfies the qualitative state plan P (Def. 17).*

3. *$\mathbf{u}_{0:f}$ satisfies the hybrid automaton $\mathcal{M}$ (Def. 20).*

4. *$\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$ is complete (Def. 21).*

Definition 23. *Given a QSP P with objective function F, the* **optimal robust solution** *is the robust feasible solution (Def. 22) that minimizes the expectation of the objective function, $E[F(\mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T)]$. This is a constrained minimum, in that $E[F(\cdot)]$ is the minimum value for which Propositions 1 through 4 of Def. 22 hold.*

A powerful aspect of the Qualitative State Plan formulation is that the set of robust solutions to choose from can be large. This means that the optimal robust solution depends heavily on the objective function chosen. Intuitively, the larger the feasible set, the more latitude optimization has to reduce the objective function. In the AUV example, the optimal robust solution is the one that minimizes the time of the event e_2, shown in Figure 2-8. In a situation where there is a possibility that an actuator has failed, we can change the optimality criterion to the *active estimation* criterion. This causes the executive to probe the system to disambiguate the hybrid state of the system. In the AUV example, this corresponds to performing a lateral maneuver, as shown in Figure 2-9. Note that this solution also robustly satisfies the Qualitative State Plan. We elaborate on the active estimation criterion in Chapter 4.

Definition 24. *Given an initial probabilistic hybrid automaton model $\mathcal{M}_0$, a sequence of observations $\mathbf{y}_{0:\tau}$ and a Qualitative State Plan $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$, the* **Robust Execution Problem** *consists of performing the following at every time step $\tau = 0, \ldots, f$:*

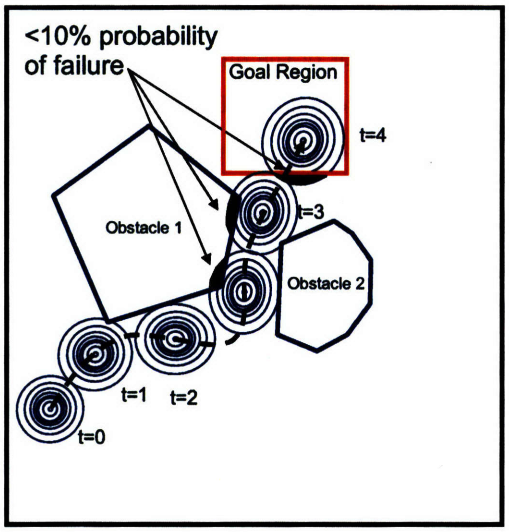


Figure 2-9. Optimal robust solution (Def. 23) to simplified Qualitative State Plan (Figure 2-7) when an active estimation criterion is used as the objective function.

1. *Generate a PHA model $\mathcal{M}_\tau$ that is consistent with the observation sequence $\mathbf{y}_{0:\tau}$ and the input sequence $\mathbf{u}_{0:\tau-1}$. As in (Def. 19) we refer to this subproblem as the* ***Model Learning Problem.***

2. *Generate a state sequence $\mathbf{x}_{0:\tau}$ that is consistent with the current guess for the PHA model $\mathcal{M}_\tau$, observation sequence $\mathbf{y}_{0:\tau}$, and the input sequence $\mathbf{u}_{0:\tau-1}$. As in (Def. 18) we refer to this subproblem as the* ***Hybrid Estimation Problem.***

3. *Generate the next control input $\mathbf{u}_\tau$ and a schedule T such that the solution for $\tau = 0, \ldots, f$, denoted $\langle \mathbf{x}_{0:f}, \mathbf{u}_{0:f}, T \rangle$, is the optimal robust solution (Def. 23) given $\mathcal{M}_\tau$, P and $\mathbf{y}_{0:\tau}$. We refer to this subproblem as the* ***Robust Control Problem.***

In the AUV mission, the control inputs $\mathbf{u}$ consist of waypoints provided to the AUV. Figure 2-10 shows one step of robust execution at $\tau = 1$. Note that since $\Delta t = 1$, $t_\tau = \tau$. A waypoint has already been generated at $\tau = 0$. Noisy observations of the AUV position are available for $\tau = 0$ and $\tau = 1$. Given these waypoints and the observations, robust execution must generate the most likely model of the AUV. This model is then used to determine the state distributions at time $\tau = 0$ and $\tau = 1$, given the observations and the waypoints. Note that the expected state deviates from the planned path due to disturbances acting on the AUV. Using the AUV model and the distribution of the AUV state, robust execution generates a control input for $\tau = 1$ and a full schedule T. Repeating this process until the schedule is complete gives the sequence of waypoints shown in Figure 2-10; this sequence is the optimal robust solution to the execution problem.

To summarize, the robust hybrid execution problem is as follows; reason explicitly about a stochastic hybrid model such a that a Qualitative State Plan is executed robustly, subject to chance

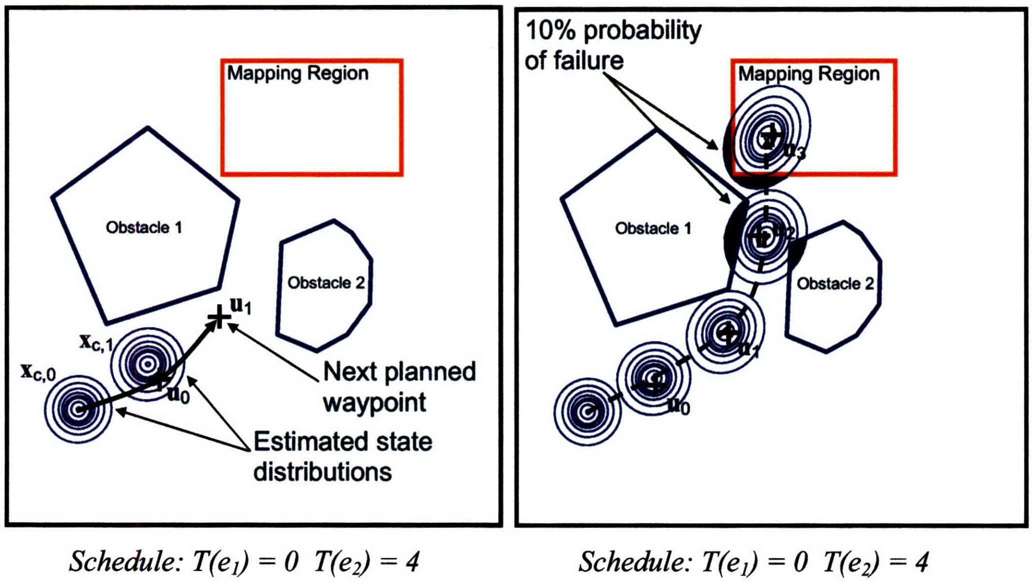


Figure 2-10. Illustration of robust execution problem. **Left:** One step of robust execution at $\tau = 1$. Noisy observations of the AUV position are available for $\tau = 0$ and $\tau = 1$. Using the waypoint already generated at $\tau = 0$, the observations of the AUV position at $\tau = 0$ and $\tau = 1$, robust execution must generate the most likely model of the AUV, determine the state distributions at time $\tau = 0$ and $\tau = 1$, and generate both an optimal control input for $\tau = 1$ and a full schedule T. **Right:** Repeating this process until the schedule is complete gives the optimal, robust sequence of waypoints and schedule shown.

constraints on episode success, and subject to constraints on the expected state. In the following chapters we provide contributions to solving each part of the robust hybrid execution problem; in Chapter 3 we tackle the Robust Control Problem; in Chapter 4 we review existing solutions to the Hybrid Estimation Problem and provide an 'active hybrid estimation' approach that uses control inputs to improve performance; and in Chapter 5 we tackle the Model Learning Problem.

Chapter 3

Robust, Optimal Control using Particles

In this chapter we introduce a novel approach to optimal, robust state plan execution for stochastic hybrid discrete-continuous plants. We refer to this approach as *Particle Control*. This approach tackles the problem of generating control inputs that maximize expected utility, while ensuring that the hard constraints of the state plan are violated with some maximum probability.

Our Particle Control approach draws from prior work in estimation, which uses samples or 'particles' to approximate the system state distribution. Particles are particularly useful as they can approximate arbitrary probability distributions, even the multimodal state distributions that occur in hybrid discrete-continuous models. The particle approximation also gives approximations to important values, namely the probability of events such as plan failure, and expected values such as the expected plan cost. Furthermore as the number of particles tends to infinity, the approximations become exact. The Particle Control algorithm works by sampling all uncertain variables to get a finite set of particles, whose trajectory depends explicitly on the control input sequence. The algorithm then approximates the chance-constrained execution problem in terms of these particles, thereby approximating an intractable stochastic optimization problem as a tractable deterministic one. For an important class of hybrid discrete-continuous system known the Jump Markov Linear System (JMLS), the resulting deterministic optimization problem can be posed as a Mixed Integer Linear Program (MILP) and solved to global optimality using efficient commercially-available solvers. Furthermore, as the number of particles tends to infinity, the approximate deterministic optimization problem converges to the true stochastic optimization problem.

This chapter is organized as follows. In Section 3.1 we describe a receding horizon approximation of the general robust execution problem (Def 24). In Section 3.2 we review prior work related to the solution of this problem. In Section 3.3 we review some results related to sampling from random variables, before outlining the new Particle Control approach for robust state plan execution in Section 3.4. We then describe the approach in detail. First we focus on execution of Configuration Goal Sequences where the execution schedule is fixed, introducing the PC-LIN algorithm in Section 3.5 for execution of continuous, linear systems and introducing the PC-JMLS algorithm in Section 3.7 for execution of Jump Markov Linear Systems. In Section 3.8 we then generalize the approach to execute Qualitative State Plans, where execution times are specified in a temporally flexible manner. We refer to this as the PC-QSP algorithm. In Section 3.10 we introduce a weighted particle control extension, which we call PC-WEIGHTED , that is more effective in the case of low probability events, such as failures. We demonstrate the new algorithms in simulation using two robotic scenarios and show that the method is effective in solving the approximated robust execution problem. Furthermore, for a large number of particles the approximation error becomes small. In addition, we demonstrate that by using state of the art Mixed Integer Linear Programming techniques, the Particle Control approach is many orders of magnitude faster than existing stochastic control approaches based on Monte Carlo Markov Chain (MCMC) methods[78].

3.1 Problem Statement

We use a receding horizon approach to solving the robust execution problem given in Definition 24. In this section we define the receding horizon execution problem.

Definition 25. *Given an initial probabilistic hybrid automaton model $\mathcal{M}_0$, a sequence of observations $\mathbf{y}_{0:\tau}$ and a Qualitative State Plan $P = \langle \mathcal{E}, \mathcal{C}, \mathcal{A}, \mathcal{G}_c, \mathcal{G}_m, F \rangle$, the* ***Receding Horizon Robust Execution Problem*** *consists of performing the following at every time step $\tau = 0, \ldots, f$:*

1. *Generate a PHA model $\mathcal{M}_\tau$ that is consistent with the observation sequence $\mathbf{y}_{0:\tau}$ and the input sequence $\mathbf{u}_{0:\tau-1}$.*

2. *Generate a state sequence $\mathbf{x}_{0:\tau}$ that is consistent with the current guess for the PHA model $\mathcal{M}_\tau$, observation sequence $\mathbf{y}_{0:\tau}$, and the input sequence $\mathbf{u}_{0:\tau-1}$.*

3. *Given the PHA model $\mathcal{M}_\tau$ and the state sequence $\mathbf{x}_{0:\tau}$, solve the* ***single-stage, limited horizon robust execution problem*** *(Def. 26) at time step τ with a planning horizon N_p to yield a solution $\langle \mathbf{x}_{\tau:\tau+N_p}, \mathbf{u}_{\tau:\tau+N_p-1}, T \rangle$. Apply the first control input $\mathbf{u}_\tau$ to the plant. Set $\mathbf{u}_\tau{}^* = \mathbf{u}_\tau$.*

until the solution $\langle \mathbf{u}_{0:f}{}^*, \mathbf{x}_{0:f}, T \rangle$ is complete (Def. 21).

Definition 26. *Let $N_p \in \mathbb{Z}$ be the* ***planning horizon****. We use the term* ***single-stage, limited horizon robust execution at time step*** *τ to refer to the problem of finding the optimal robust solution $\langle \mathbf{x}_{\tau:\tau+N_p}, \mathbf{u}_{\tau:\tau+N_p-1}, T \rangle$, given:*

1. *A probabilistic hybrid automaton $\mathcal{M}_\tau$;*

2. *A distribution of the system state at time step τ, denoted $\mathbf{x}_\tau$;*

3. *A qualitative state plan P, and a schedule $\{\langle e_i, T(e_i)\rangle \mid T(e_i) < t_\tau\}$ of events in the qualitative state plan that are scheduled before t_τ;*

Remark 3.1. *In the case that the Qualitative State Plan P is a Configuration Goal Sequence, the schedule T is specified in the plan. This reduces the complexity of the execution problems (Def. 24 and Def. 25).*

Remark 3.2. *In order to simplify notation slightly, we assume for the rest of this chapter that we are solving the single-stage, limited horizon robust execution problem (Def. 26) at time $\tau = 0$. That is, we would like to find the optimal, robust solution $\langle \mathbf{x}_{0:N_p}, \mathbf{u}_{0:N_p-1}, T\rangle$.*

Intuitively, the single-stage, limited horizon robust execution problem is the restriction of the general robust control problem (Def. 24) over a limited planning window. One important difference with general robust execution is that, since the spans of the state and input sequences are now limited, the sequences are no longer required to be complete (Item 4 in Def. 24), but rather are allowed to be only partial. We also require that events that have previously been scheduled before time step τ remain scheduled at the same time (Item 3 in Def. 26).

In this section we tackle the single-stage, limited horizon robust execution problem (Def. 26). In later sections we consider the remaining parts of the receding horizon robust execution problem (Def. 25); in Chapter 4 we tackle the problem of generating a consistent state sequence, while in Chapter 5 we tackle the problem of generating a consistent PHA model.

3.2 Related Work

For systems with continuous dynamics, the robust plan execution problem is one of control under stochastic uncertainty. Control under stochastic uncertainty has been researched extensively, for example[8, 16, 105, 107]. Early approaches, such as Linear Quadratic Gaussian control[16], used the certainty equivalence principle[6]; this enables uncertain variables to be replaced with their expectations, and control laws to be designed in terms of these expectations. For unconstrained problems with linear systems and Gaussian noise, controllers can be designed that are optimal with regard to the original stochastic control problem.

Alternative approaches express certain classes of stochastic control problems as Markov Decision Processes (MDPs)[109]. For discrete decision and state spaces, value iteration can be used to find the control policy that maximizes expected reward. Continuous spaces can be handled by this approach using discretization; however, the problem quickly becomes intractable as the number of discrete states grows[69]; hence the application of this approach to dynamic systems with large continuous state spaces has been limited. In addition, in this thesis we are concerned with chance-constrained problems, which are not readily expressed in an MDP framework.

Robust planning has received a great deal of attention in the Artificial Intelligence community. *Conformant planning* aims to find a strategy that ensures that the final system state is the same no matter what the state of the world is. The Conformant Graphplan approach[124] builds on the Graphplan algorithm[21] by developing separate plan graphs for each possible state of the world, and linking the control actions across the different graphs. Search is performed to find one set of control actions that achieves the goal for every graph. This is analogous to the Particle Control approach presented in this thesis, in that particles represent different possible outcomes of the world, and that the control inputs are the same for all particles. Later work by [35] avoided the need to store sets of possible worlds by, instead, reasoning about the effects of action sequences. Important differences between this work and the Particle Control approach are first, that Particle Control deals with continuous-valued control inputs and uncertainty; second, that we consider temporally-flexible plans; and third that we are concerned with finding optimal control sequences that ensure a minimum probability of success, rather than guaranteeing success for all possible outcomes. In this spirit, *probabilistic planning* looks for plans such that success is above a threshold[75, 85, 84]. [127] introduced a *scenario-based* approach to stochastic constraint programming. By considering each scenario as a possible outcome of a stochastic variable, this approach converts a chance-constrained stochastic program into a deterministic one that can be solved using existing constraint solvers. Again, this is analogous to the Particle Control approach, but is restricted to discrete decision and uncertainty spaces.

Robust Model Predictive Control under set-bounded uncertainty has been a topic of research for many years, see [15] and [67] for surveys. This work assumes that all uncertainty (in terms of disturbances, model uncertainty and initial state) is guaranteed to be within a set, but does not place any restrictions beyond this. [110], for example, uses a *constraint-tightening* approach to ensure that constraints are satisfied, and that the optimization problem is feasible at each planning horizon. In this thesis, by contrast, we are concerned with the case where uncertainty is characterized as a random variable with a known probability distribution, and seek to guarantee constraint satisfaction with a certain probability. Furthermore, we argue that in many cases a probabilistic representation is more appropriate than a set-bounded one; for example in the case of wind disturbances it is natural to describe the mean and variance of the wind velocity, but it is difficult to place absolute bounds on the velocity. Prior authors, for example [82, 119], have nevertheless employed set-bounded techniques to solve problems with probabilistic uncertainty, by converting a probabilistic representation into a set-bounded one. For chance-constrained problems, this is performed by finding a set such that there is a small probability δ of any uncertain variable falling outside of the set. By ensuring that the plan succeeds for all values within the set, we in turn ensure that the probability of failure is at most δ. This approach has a number of drawbacks, since, even in very simple cases, there are an uncountable number of suitable sets from which to choose from. First, choosing which of these sets *before* the planning process starts introduces conservatism. Second, finding the suitable set may be computationally complex in the general case. For the case of purely continuous-valued systems, typical approaches, such as those proposed by [82, 119], constrain the form of the set to ensure that this computation is tractable; however this introduces a relatively high level of conservatism and relies on the assumptions that all uncertainty is Gaussian. In this thesis we are concerned not only with non-Gaussian uncertainty distributions, but also hybrid discrete-continuous systems. Such systems lead to multi-modal state distributions for which calculating equivalent bounding sets is extremely challenging. Nonetheless, comparison of the approximation introduced by the Particle Control approach with the conservatism introduced by set-bounded techniques is worthwhile. We defer this comparison to future work.

Control of Jump Markov Linear Systems has been studied since their introduction in the 1960's. Much work has been done in the area of feedback control for JMLS, see [28] for a survey. [22] considered the robust feedback control problem, where robustness is with regard to *set-bounded* uncertainty. Recent work developed a Model Predictive Control approach for JMLS that imposes constraints on the mean and covariance of the system state[131]. These are not the same as chance constraints, even when all forms of uncertainty are Gaussian, since the state distribution in a JMLS is multimodal.

For robust task plan execution, we are concerned with *chance-constrained* stochastic control. In the Model Predictive Control community, recent work has considered chance-constrained finite horizon control of linear systems when all forms of uncertainty are Gaussian. A number of authors suggested approaches that, given these assumptions, pose the stochastic control problem as an equivalent deterministic optimization problem. Chance constraints on individual variables where considered by [12, 82, 83, 119], while [60] proposed a conservative approach to deal with chance constraints on the joint distribution of multiple variables. The particle-based approach that we introduce in this thesis extends this work in several ways. First, we do not rely on the uncertainty being Gaussian; second, we can handle more general forms of uncertainty, such as uncertain constraints, model uncertainty and multiplicative uncertainty. Third, we deal with hybrid discrete-continuous systems, and finally, we handle temporally flexible constraints.

Sampling-based methods have been used extensively for estimation [19, 39, 51, 90, 92]. These particle filtering methods are superior to traditional Kalman Filtering methods for non-Gaussian probability distributions; first, they approach the Bayes-optimal estimate with sufficient samples[36], which is not the case for Kalman Filters; second, they have been shown empirically to perform better with regard to average estimation error in many applications, for example [72, 2].

In stochastic control and decision making, a number of authors have proposed the approximation of a stochastic control problem using sampling. Samples are variously referred to as 'particles'[36, 53], 'simulations'[123] and 'scenarios'[96, 127, 137]. In the Pegasus system, the authors propose the conversion of an MDP with stochastic transitions into an equivalent one where all randomness is represented in the initial state[96]. The method draws a finite number of samples, or scenarios, from the initial state, and finds the policy that optimizes the sample mean of the reward for the scenarios. In a similar manner, [53] propose a method for finite horizon control that approximates the expected cost of a given control sequence using a finite number of samples. [5] uses particles to approximate the value function and its gradient in an optimal stochastic control problem, and uses a gradient search method to find a locally optimal solution. In the case of optimal sensor scheduling, [123] employs a similar approximation method, but in this case uses an optimization approach known as Stochastic Approximation[114] to minimize the cost function defined in terms of stochastic variables. This approach is guaranteed to converge to a local optimum under certain assumptions.

In this thesis we extend this work in four ways. First, we incorporate constraints on the probability of failure, enabling *robust* stochastic control. Control with constraints on the probability of events such as constraint violation, or mission failure, is a powerful capability that is also a much more challenging problem than control with only constraints on expected values, for example. Second, we show that in the case of stochastic linear dynamic systems and Jump Markov Linear Systems, the resulting robust control problem can be solved to *global* optimality using Mixed Integer Linear Programming in an efficient manner. Third, we introduce an *importance weighting* extension, and show empirically that this dramatically reduces the number of particles needed for robustness to low-probability mode transitions such as failures. Finally, we generalize the approach to execution of temporally flexible plans.

Recently there have been a number of advances in Monte Carlo Markov Chain(MCMC) methods [115]. These are sampling-based approaches that are typically used to solve problems in state estimation[101]. The key idea is to set up a Markov Chain with the property that when transitions are sampled iteratively, the empirical distribution over states converges to the true state distribution. MCMC can also be used to solve optimal control problems, by defining the Markov Chain so that the state distribution clusters around the optimal control value. Approaches based on MCMC can, in principle, deal with control of systems with both discrete and continuous state, and have been applied to chance-constrained problems[77].

The chance-constrained particle control approach that we present in this thesis has two key advantages over MCMC approaches. First, by converting the stochastic problem to a deterministic one we are able to use deterministic optimization methods that can deal with much larger problems than Monte Carlo optimization approaches and that can deal explicitly with constraints on the distribution, rather than penalizing constraint violation in the cost function; the latter leads to suboptimality in the solution. Second, by exploiting the structure of linear systems and JMLS, we are able to employ Mixed Integer Linear Programming; this enables fast solution of large problems to global optimality. We demonstrate empirically that the particle control approach gives a dramatic improvement in performance over MCMC approaches.

3.3 Sampling from Random Variables

Previous work has shown that approximating the probability distribution of a random variable using samples drawn from that distribution, or particles, can lead to tractable algorithms for estimation and control[36]. Here we review some properties of samples drawn from random variables.

Suppose that we have a multivariate random variable X that has a probability distribution $p(\mathbf{x})$. We draw N independent, identically distributed random samples $\langle \mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)} \rangle$ from this distribution. Often, we would like to calculate an expectation involving this random variable:

$$E_X[f(X)] = \int_X f(\mathbf{x})p(\mathbf{x})d\mathbf{x}. \tag{3.1}$$

In many cases this integral cannot be evaluated in closed form. Instead it can be approximated using the sample mean:

$$\hat{E}_X[f(X)] = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^{(i)}). \tag{3.2}$$

From the strong law of large numbers, the sample mean converges to the true expectation as N tends to infinity.

$$\hat{E}_X[f(X)] \longrightarrow E_X[f(X)]. \tag{3.3}$$

Hence the expectation, which could not be evaluated exactly in closed form, can be approximated as a summation over a number of particles. This can be used to approximate the probability of an event, such as the event that the value $f(\mathbf{x})$ does not fall in the set A, where $f(\cdot)$ is an arbitrary function. This probability is given exactly by:

$$P_A = \int_{f(\mathbf{x})\notin A} p(\mathbf{x})d\mathbf{x}. \tag{3.4}$$

This expression is equivalent to the expectation:

$$P_A = E_X[g(\mathbf{x})], \tag{3.5}$$

where $g(\cdot)$ is an indicator function given by:

$$g(\mathbf{x}) = \begin{cases} 0 & f(\mathbf{x}) \in A \\ 1 & f(\mathbf{x}) \notin A. \end{cases} \tag{3.6}$$

We can therefore approximate P_A as:

$$\hat{P}_A = \frac{1}{N} \sum_{i=1}^{N} g(\mathbf{x}^{(i)}). \tag{3.7}$$

Note that $\sum_{i=1}^{N} g(\mathbf{x}^{(i)})$ is simply the number of particles for which $f(\mathbf{x}^{(i)}) \in A$. Assuming that evaluating $f(\cdot)$, and checking whether a given value is in A, are both straightforward, calculating $\hat{P}_A$ is also; we simply need to count how many of the propagated particles, $f(\mathbf{x}^{(i)})$ fall within A. By contrast, evaluating P_A as in (3.4) requires a finite integral over an arbitrary probability distribution, where even calculating the bounds on the integral may be intractable. Hence, the particle-based approximation is extremely useful, especially given the convergence property:

$$\hat{P}_A \longrightarrow P_A, \tag{3.8}$$

as N tends to infinity. In Section 3.4 we use this property to approximate the stochastic control problem defined in Section 3.1.

3.3.1 Importance Weighting

In certain situations, drawing samples from the distribution $p(\mathbf{x})$ may be intractable, or undesirable. In such cases, previous work suggested sampling from an alternative *proposal* distribution and using *importance sampling* to correct for the discrepancy between the desired distribution and the proposal distribution[36]. We review relevant results here.

The proposal distribution $q(\mathbf{x})$ is chosen so that $p(\mathbf{x}) > 0$ implies $q(\mathbf{x}) > 0$. Intuitively, the proposal distribution $q(\mathbf{x})$ must 'cover' the true distribution $p(\mathbf{x})$, so that any value of $\mathbf{x}$ that has a non-zero probability of occurring according to the true distribution must also have a non-zero probability in the proposal distribution. We draw N independent, identically distributed random samples $\langle \mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)} \rangle$ from $q(\mathbf{x})$. To each sample we assign an importance weight w_i, where:

$$w_i = \frac{p(\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)})}. \tag{3.9}$$

Intuitively, the importance weight corrects for the discrepancy between the proposal distribution and the true distribution. For example, fewer samples than desired will fall where $q(\mathbf{x})$ is small compared to $p(\mathbf{x})$, however these samples will be weighted more highly than where $q(\mathbf{x}) \approx p(\mathbf{x})$.

In order to approximate the expectation of the function $f(\cdot)$ we now use the *weighted* sample mean:

$$\hat{E}_X[f(X)] = \frac{1}{N}\sum_{i=1}^{N} w_i f(\mathbf{x}^{(i)}), \tag{3.10}$$

where the samples $\mathbf{x}^{(i)}$ are drawn from the proposal distribution $q(\mathbf{x})$. From the strong law of large numbers, we have the convergence property as N tends to infinity:

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N} w_i f(\mathbf{x}^{(i)}) &\longrightarrow \int w_i f(\mathbf{x})q(\mathbf{x})d\mathbf{x}\\ &= \int \frac{p(\mathbf{x})}{q(\mathbf{x})} f(\mathbf{x})q(\mathbf{x})d\mathbf{x} = E_X[f(X)].\end{aligned} \tag{3.11}$$

Hence the approximated expectation converges to the true expectation as the number of samples tends to infinity:

$$\hat{E}_X[f(X)] \longrightarrow E_X[f(X)]. \tag{3.12}$$

In order to approximate the probability of the event $f(\mathbf{x}) \notin A$ we use the *weighted* number of propagated particles that do not fall within A:

$$\hat{P}_A = \frac{1}{N}\sum_{i=1}^{N} w_i g(\mathbf{x}^{(i)}), \tag{3.13}$$

where $g(\cdot)$ is as defined in (3.6). As in (3.7) we have the convergence property $\hat{P}_A \longrightarrow P_A$ as $N \rightarrow \infty$.

3.4 Outline of Chance-Constrained Particle Control Method

In this section we describe a new method for solving the robust stochastic control problem described in Section 3.1, which we call the Particle Control approach. In Section 3.5 we show that with linear dynamic systems and polygonal constraints, in the fixed-schedule case, the resulting optimization can be solved efficiently using MILP; this gives us the PC-LIN algorithm. In Section 3.7 we show that this is also the case for Jump Markov Linear Systems, and we present the PC-JMLS algorithm. In Section 3.8 we desribe the PC-QSP algorithm, which uses the Particle Control approach for robust execution of temporally-flexible Qualitative State Plans. Finally, in Section 3.10 we extend the PC-JMLS approach to deal with low probability mode transitions and present the PC-WEIGHTED algorithm.

The key observation behind the new method is that, by approximating all probabilistic distributions using particles, an intractable stochastic optimization problem can be approximated as a tractable deterministic optimization problem. By solving this deterministic problem we obtain an approximate solution to the original stochastic problem, with the additional property that as the number of particles used tends to infinity, the approximation becomes exact.

In outlining the method, we consider a general discrete-time dynamic system where the future states $\mathbf{x}_{1:N_p}$ are functions of the control inputs $\mathbf{u}_{0:N_p-1}$, the initial state $\mathbf{x}_0$, and disturbances $\nu_{0:N_p-1}$:

$$\begin{aligned}
\mathbf{x}_1 &= f_1(\mathbf{x}_0, \mathbf{u}_0, \nu_0) \\
\mathbf{x}_2 &= f_2(\mathbf{x}_0, \mathbf{u}_0, \mathbf{u}_1, \nu_0, \nu_1) \\
&\vdots \\
\mathbf{x}_{N_p} &= f_{N_p}(\mathbf{x}_0, \mathbf{u}_{0:N_p-1}, \nu_{0:N_p-1}).
\end{aligned} \tag{3.14}$$

The initial state and disturbances are uncertain, but are modeled as random variables with known distributions. Hence the future states are also random variables, whose distributions depend on the control inputs. We assume that the initial state and disturbances are independent.Note that a Probabilistic Hybrid Automaton is a special case of this general discrete-time system.

The new chance constrained particle control method is given in Table 3.1. The method is illustrated in Figure 3-1. We now describe each step of the algorithm in more detail.

Sample all uncertain variables. First, N samples are drawn from all uncertain variables. These variables can include disturbances, initial state, system model parameters, mode transitions, obstacle locations and more; however for notational simplicity we consider only disturbances and initial state uncertainty. This sampling process gives a set $\{\mathbf{x}_0^{(i)}, \nu_{0:N_p-1}^{(i)}\}$ for each $i = 1, \dots, N$. In Figure 3-1a) we illustrate the generation of samples for the initial state distribution. The particles may be generated using a pseudo-random number generator[1], or may be generated using simulations of the physical processes underlying the plant model. While these simulations may be computationally intensive, this computation can be done off-line; since we assume independence of the random variables and the system state, we can generate and store a large number of samples before execution begins, and perform a random lookup of samples when needed.

Express particles as analytic function of control inputs. Each particle $\mathbf{x}_{1:N_p}^{(i)}$ is an analytic function that corresponds to the state trajectory given a particular set of samples $\{\mathbf{x}_0^{(i)}, \nu_{0:N_p-1}^{(i)}\}$. The analytic function *depends explicitly on the control inputs* $\mathbf{u}_{0:N_p-1}$, which are yet to be generated:

$$\mathbf{x}_{1:N_p}^{(i)} \triangleq \langle \mathbf{x}_1^{(i)} \mathbf{x}_2^{(i)} \dots \mathbf{x}_{N_p}^{(i)} \rangle \quad \mathbf{x}_\tau^{(i)} = f_\tau(\mathbf{x}_0^{(i)}, \mathbf{u}_{0:\tau-1}, \nu_{0:\tau-1}^{(i)}),$$

[1]Standard techniques exist for generating random samples from an arbitrary distribution, given random samples from a uniform distribution; see [17] for details.

where $\mathbf{x}_0^{(i)}$ and $\nu_{0:\tau-1}^{(i)}$ are *known sampled values*, whereas $\mathbf{u}_{0:\tau-1}$ are control variables over which to optimize. Applying the same control sequence to each particle, the effect of a given control sequence on the particle distribution can be evaluated, as shown in Figure 3-1b).

Approximate the stochastic problem in terms of particles. Approximating the stochastic execution problem requires approximation of the chance constraints, the expected state constraints, and the expected cost. We consider first the approximation of chance constraints. The probability of any activity in the set $\mathcal{A}(c_c)$ failing is given exactly by:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) = \int_{\mathbf{x}_{1:N_p}} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}, T\Big) p(\mathbf{x}_{1:N_p}) d\mathbf{x}_{1:N_p}, \tag{3.15}$$

where $s(\cdot)$ is the function that indicates whether a set of activities fails or not, defined in (2.4); we repeat the definition here:

$$s(\mathcal{A}, \mathbf{x}_{c,1:N_p}{}^{(i)}, T) = \begin{cases} 0 & \text{All episodes } a \in \mathcal{A} \text{ succeed for } \mathbf{x}_{c,1:N_p}{}^{(i)} \text{ and schedule } T \\ 1 & \text{There exists an episode } a \in \mathcal{A} \text{ that fails for } \mathbf{x}_{c,1:N_p}{}^{(i)} \text{ and schedule } T. \end{cases} \tag{3.16}$$

Using (3.7), this probability can be approximated with the the generated particles as:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) \approx \frac{1}{N} \sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}^{(i)}, T\Big). \tag{3.17}$$

The chance constraint c_c is then approximated as follows:

$$\frac{1}{N} \sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}^{(i)}, T\Big) \le \delta(c_c). \tag{3.18}$$

In other words, the approximate chance constraint is that for no more than $\delta(c_c)$ of the particles can any activity in $\mathcal{A}(c_c)$ fail. This is ilustrated in Figure 3-1c). Note that a particle represents a state *trajectory* over the entire planning horizon.

We now turn our attention to expected state constraints. A constraint c_m on the expected state can be written:

$$s\Big(\mathcal{A}(c_m), E[\mathbf{x}_{1:N_p}], T\Big) = 0. \tag{3.19}$$

Using the sample mean approximation to the full expectation we have the approximated constraint:

$$s\left(\mathcal{A}(c_m), \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_{1:N_p}^{(i)}, T\right) = 0. \tag{3.20}$$

Finally, the cost function is approximated as follows:

$$\hat{F} \triangleq \frac{1}{N}\sum_{i=1}^{N} F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{1:N_p}^{(i)}, T) \approx E[F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{1:N_p}, T)] \tag{3.21}$$

These approximations are illustrated in Figure 3-1c). The approximations give a deterministic optimization problem, which is solved to give the optimal solution to the approximated problem.

Solve deterministic optimization problem. We use existing deterministic optimization approaches to solve the resulting optimization problem. We elaborate on the solution method for different plant types in Sections 3.5 through 3.8. The optimal solution for the AUV example is shown in Figure 3-1d).

Convergence property. As the number of particles tends to infinity, the approximated problem converges to the true problem from the results in Section 3.3. In particular, from (3.8), we have convergence of the approximated probability of episode failure as the number of particles tends to infinity:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) \longrightarrow \frac{1}{N}\sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}^{(i)}, T\Big). \tag{3.22}$$

We also have convergence of the expected state and the expected cost to their true values as the number of particles tends to infinity, from (3.3):

$$\begin{aligned} E[\mathbf{x}_{1:N_p}] &\longrightarrow \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_{1:N_p}^{(i)} \\ \frac{1}{N}\sum_{i=1}^{N} F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{1:N_p}^{(i)}, T) &\longrightarrow E[F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{1:N_p}, T)]. \end{aligned} \tag{3.23}$$

The Particle Control approach described in Table 3.1 applies to the general form of a Qualitative State Plan (Def. 3) and solves an approximation of the single-stage, limited horizon robust execution problem (Def. 25). In the special case that the Qualitative State Plan is a Configuration Goal Sequence (Def. 16), the schedule T is specified in the plan. In this case the deterministic optimization in Table 3.1 is over $\mathbf{u}_{0:N_p-1}$ only.

Remark 3.3. *The Particle Control approach generates an open-loop plan for a single planning horizon, meaning that the control input does not change depending on the actual noise values realized during execution of the plan. This strategy can cause the state uncertainty to become very large. In practice, we must usually assume that the open-loop plan issues commands to an inner-loop controller that* does *respond to disturbances reactively. Any such (fixed) inner-loop controller can be incorporated in the system dynamics (3.14) without loss of generality. In Section 3.6 we demonstrate the Particle Control approach with two scenarios involving inner-loop controllers.*

Function ParticleControlMain(P,$\mathcal{M}$,N,N_P) **returns** $(\mathbf{u}_{0:N_p-1}, T)$

1) *Generate N samples from the joint distribution of all uncertain variables.*

2) *Express the distribution of the future state trajectories approximately as a set of N particles.* Each particle $\mathbf{x}^{(i)}_{1:N_p}$ is an analytic function that corresponds to the state trajectory given a particular set of samples $\{\mathbf{x}^{(i)}_0, \nu^{(i)}_{0:N_p-1}\}$. The analytic function *depends explicitly on the control inputs* $\mathbf{u}_{0:N_p-1}$, which are yet to be generated.

$$\mathbf{x}^{(i)}_{1:N_p} \triangleq \langle \mathbf{x}^{(i)}_1 \mathbf{x}^{(i)}_2 \ldots \mathbf{x}^{(i)}_{N_p} \rangle \quad \mathbf{x}^{(i)}_\tau = f_\tau(\mathbf{x}^{(i)}_0, \mathbf{u}_{0:\tau-1}, \nu^{(i)}_{0:\tau-1}),$$

where $\mathbf{x}^{(i)}_0$ and $\nu^{(i)}_{0:\tau-1}$ are *known sampled values*, whereas $\mathbf{u}_{0:\tau-1}$ are control variables over which to optimize.

3) *Approximate the chance constraints in terms of the generated particles.* The probability of any activity in the set $\mathcal{A}(c_c)$ failing is given exactly by:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) = \int_{\mathbf{x}_{1:N_p}} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}, T\Big) p(\mathbf{x}_{1:N_p}) d\mathbf{x}_{1:N_p}, \tag{3.24}$$

where $s(\cdot)$ is the function that indicates whether a set of activities fails or not, defined in (2.4). Using (3.7), this probability can be approximated with the the generated particles as:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) \approx \frac{1}{N}\sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}^{(i)}_{1:N_p}, T\Big). \tag{3.25}$$

The chance constraint c_c is then approximated as follows:

$$\frac{1}{N}\sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}^{(i)}_{1:N_p}, T\Big) \leq \delta(c_c). \tag{3.26}$$

In other words, the approximate chance constraint is that for no more than $\delta(c_c)$ of the particles can any activity in $\mathcal{A}(c_c)$ fail. Note that a particle represents a state *trajectory* over the entire planning horizon.

4) *Approximate the constraints on the expected state.* A constraint c_m on the expected state can be written:

$$s\Big(\mathcal{A}(c_m), E[\mathbf{x}_{1:N_p}], T\Big) = 1. \tag{3.27}$$

Using the sample mean approximation to the full expectation we have the approximated constraint:

$$s\left(\mathcal{A}(c_m), \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}^{(i)}_{1:N_p}, T\right) = 1. \tag{3.28}$$

5) *Approximate the expectation of the cost function in terms of particles*

$$\hat{F} \triangleq \frac{1}{N}\sum_{i=1}^{N} F(\mathbf{u}_{0:N_p-1}, \mathbf{x}^{(i)}_{1:N_p}, T) \approx E[F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{1:N_p}, T)] \tag{3.29}$$

6) *Solve deterministic constrained optimization problem for control inputs* $\mathbf{u}_{0:N_p-1}$:
Minimize $\hat{F}(\mathbf{u}_{0:N_p-1}, \mathbf{x}^{(1)}_{1:N_p}, \cdots, \mathbf{x}^{(N)}_{1:N_p}, T)$ over $\mathbf{u}_{0:N_p-1}$ and T.
Subject to:

$$\frac{1}{N}\sum_{i=1}^{N} s\Big(\mathcal{A}(c_c), \mathbf{x}^{(i)}_{1:N_p}, T\Big) \leq \delta(c_c), \tag{3.30}$$

for all chance constraints c_c, and

$$s\left(\mathcal{A}(c_m), \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}^{(i)}_{1:N_p}, T\right) = 0, \tag{3.31}$$

for all expected state constraints c_m.

Table 3.1. Particle Control Algorithm

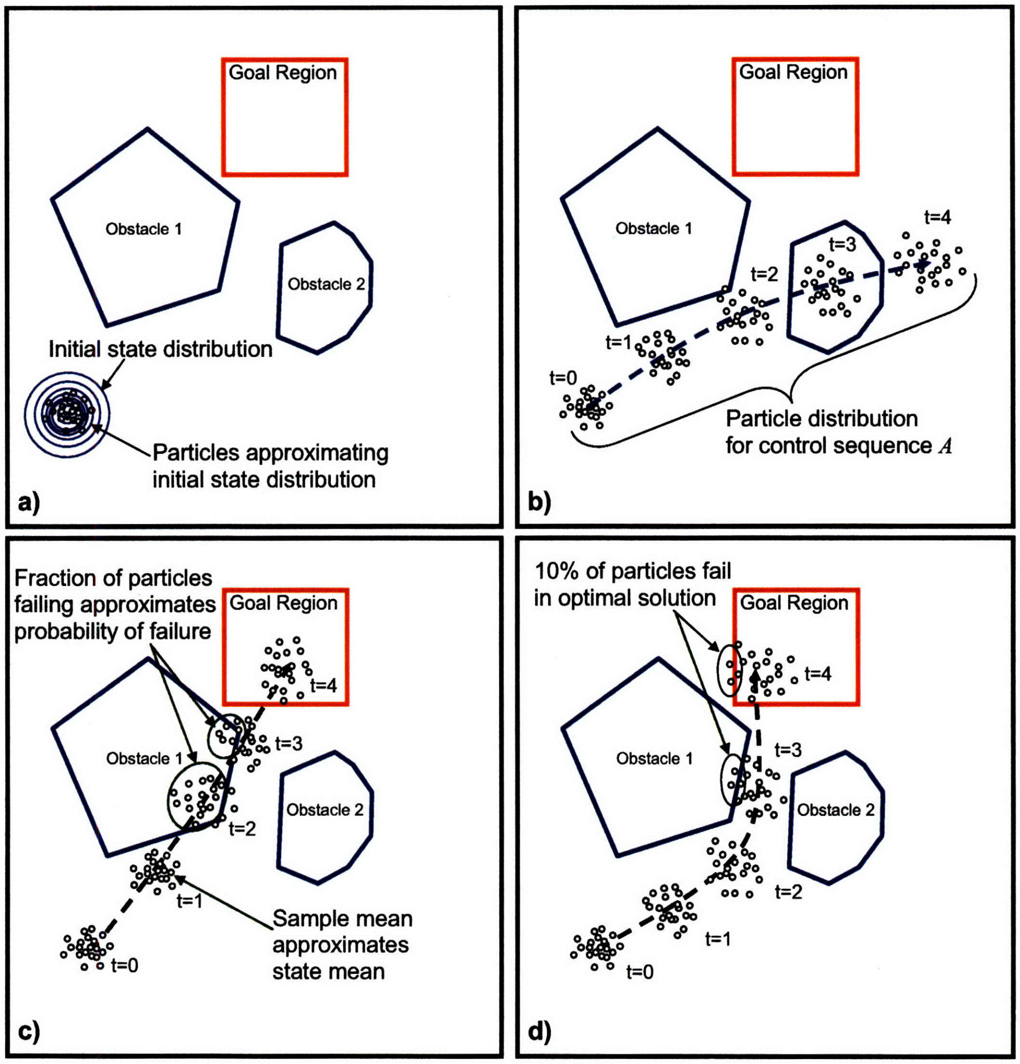

Goal Region
Obstacle 1
Obstacle 2
Initial state distribution
Particles approximating initial state distribution
a)
t=0
t=1
t=2
t=3
t=4
Particle distribution for control sequence A
b)
Fraction of particles failing approximates probability of failure
Sample mean approximates state mean
c)
10% of particles fail in optimal solution
d)

Figure 3-1. Illustration of Particle Control algorithm for chance constrained execution of Qualitative State Plans, applied to simple AUV scenario illustrated in Figure 1-1.

The general formulation presented in this section encompasses a very general set of dynamic systems, including ones with nonlinear dynamics and hybrid state; the key restriction being that the distributions of the uncertain variables must be independent of the control inputs and system state.[2] This restriction is necessary, since the particle control approach draws samples from all random distributions *before* searching for the optimal control sequence and corresponding state distribution. It is not necessarily true, however, that the optimization problem that results from this formulation is tractable. In Sections 3.5 and 3.7 we show that in the case of linear and Jump Markov Linear Systems and polygonal feasible regions, the optimization can be solved using efficient Mixed Integer Linear Programming methods.

3.5 Fixed-Schedule Particle Control for Linear Systems

In this section we consider the execution of a Configuration Goal Sequence, in which case the schedule T is specified in the plan. We also restrict our attention to the case of linear system dynamics and polygonal feasible regions. Furthermore, we assume that the cost function F is piecewise linear. Previous work has shown that optimal path planning with obstacles for vehicles such as aircraft or satellites can be posed as finite horizon control design for linear systems in polygonal feasible regions[111, 118]. Optimality can be defined in terms of fuel use or time, for example. We extend this work by showing that the particle control method outlined in Section 3.4 can be used to design control inputs for linear systems that are robust to probabilistic uncertainty in the initial state and disturbances.

We consider the linear discrete time system model:

$$\mathbf{x}_{c,\tau+1} = A\mathbf{x}_{c,\tau} + B\mathbf{u}_{\tau} + \nu_{\tau}, \tag{3.32}$$

where $\mathbf{x}_c \in \mathbb{R}^{n_x}$, $\mathbf{u} \in \mathbb{R}^{n_u}$ and $\nu \in \mathbb{R}^{n_x}$. Substituting this system model into (3.24) we obtain the following equation for ${\mathbf{x}_{c,\tau}}^{(i)}$, that is, the system state at time step τ for particle i:

$${\mathbf{x}_{c,\tau}}^{(i)} = \sum_{j=0}^{\tau-1} A^{\tau-j-1}B(\mathbf{u}_j + \nu_j^{(i)}) + A^{\tau}{\mathbf{x}_{c,0}}^{(i)}. \tag{3.33}$$

Note that this is a linear function of the control inputs, and that ${\mathbf{x}_{c,0}}^{(i)}$ and $\nu_j^{(i)}$ are known values. Hence each particle ${\mathbf{x}_{c,1:N_p}}^{(i)}$, which describes the entire state sequence for a given set of samples, is a known linear function of the control inputs. Furthermore, the sample mean of the particles, which is a sum of these functions, is a known linear function of the control inputs.

[2]This is true of Probabilistic Hybrid Automata that do not have autonomous mode transitions

3.5.1 Chance Constraints

In accordance with (3.30), for each chance constraint c_c we must constrain the number of particles for which any activity in the set $\mathcal{A}(c_c)$ of activities fails. For each chance constraint we define a set of N binary variables $\langle z_1(c_c), \cdots, z_N(c_c)\rangle$, where $z_i(c_c) \in \{0,1\}$ and N is the number of particles. These binary variables are defined so that $z_i(c_c) = 0$ implies that for particle i, all of the activities in $\mathcal{A}(c_c)$ succeed. We then constrain the sum of these binary variables:

$$\sum_{i=1}^{N} z_i(c_c) \leq \delta(c_c) \cdot N. \tag{3.34}$$

This constraint ensures that the fraction of particles for which the activities $\mathcal{A}(c_c)$ fail is at most $\delta(c_c)$. In addition we impose constraints such that:

$$z_i(c_c) = 0 \implies s\Big(\mathcal{A}(c_c), \mathbf{x}_{c,1:N_p}{}^{(i)}, T\Big) = 0. \tag{3.35}$$

Recall that $s(\cdot)$ is an indicator function describing whether a set of activities fails; we repeat the definition here:

$$s(\mathcal{A}, \mathbf{x}_{c,1:N_p}{}^{(i)}, T) = \begin{cases} 0 & \text{All episodes } a \in \mathcal{A} \text{ succeed for } \mathbf{x}_{c,1:N_p}{}^{(i)} \text{ and schedule } T \\ 1 & \text{There exists an episode } a \in \mathcal{A} \text{ that fails for } \mathbf{x}_{c,1:N_p}{}^{(i)} \text{ and schedule } T. \end{cases} \tag{3.36}$$

3.5.2 Activity Satisfaction

In this section we describe how to impose constraints such that $z_i(c_c) = 0$ implies that for particle i, every activity $a \in \mathcal{A}(c_c)$ succeeds. As noted in Section 2 we can, without loss of generality, consider only remain-in and end-in activities. We denote the subset of $\mathcal{A}(c_c)$ that are remain-in activities as $\mathcal{A}_\forall(c_c)$, and the subset of $\mathcal{A}(c_c)$ that are end-in activities as $\mathcal{A}_E(c_c)$.

A *remain in state* $R_\forall$ activity between events e_S and e_E imposes $\mathbf{x}_{c,\tau} \in R_\forall$ for all time steps $t_\tau \in [T(e_S), T(e_E)]$. In the single-stage, limited horizon robust execution problem, we are planning over a limited horizon of discrete time steps $\tau = 0, \ldots, N_p$. We, therefore, approximate the remain-in activity as:

$$\mathbf{x}_{c,\tau} \in R_\forall \quad \forall \tau \in \{0, \ldots, N_p\} \ s.t.\ t_\tau \in [T(e_S), T(e_E)]. \tag{3.37}$$

In other words, $\mathbf{x}_{c,\tau} \in R_\forall$ for all time steps $t_\tau \in [T(e_S), T(e_E)]$ that are *also within the planning horizon.*

An *end in* activity imposes $\mathbf{x}_c \in R_E$ at time $T(e_E)$. The end-in requirement is approximated as:

$$\mathbf{x}_{c,\tau} \in R_E \quad \forall \tau \in \{0, \ldots, N_p\} \ s.t.\ t_\tau \in \Big[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\Big]. \tag{3.38}$$

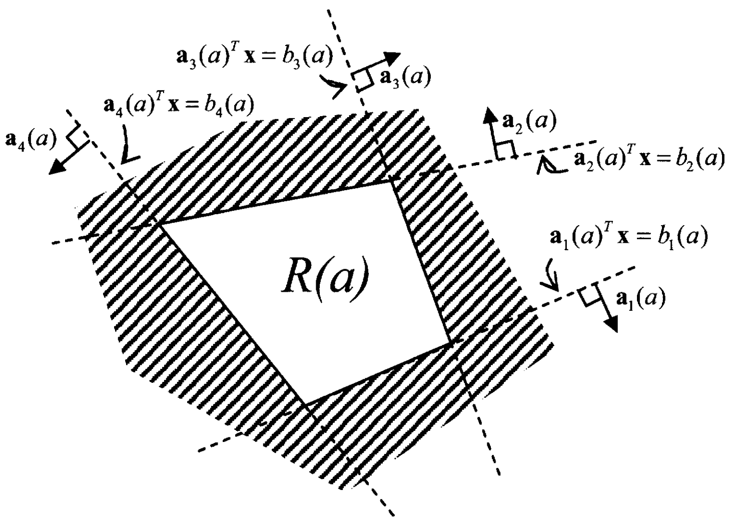

$\mathbf{a}_3(a)^T\mathbf{x} = b_3(a)$
$\mathbf{a}_3(a)$
$\mathbf{a}_4(a)^T\mathbf{x} = b_4(a)$
$\mathbf{a}_4(a)$
$\mathbf{a}_2(a)$
$\mathbf{a}_2(a)^T\mathbf{x} = b_2(a)$
$\mathbf{a}_1(a)^T\mathbf{x} = b_1(a)$
$\mathbf{a}_1(a)$
$R(a)$

Figure 3-2. Two-dimensional convex polygonal feasible region $R(a)$ associated with activity a. The vectors $\mathbf{a}_1(a), \cdots, \mathbf{a}_{N_a}(a)$ are the unit outward normals to the N_a line segments that define the feasible region.

In other words, $\mathbf{x}_c \in R_E$ at the time step closest to $T(e_E)$ as long as $T(e_E)$ is scheduled within the planning horizon. The approximations (3.37) and (3.38) are the same as those carried out by [76]. No constraint is imposed on the system state outside of the planning horizon. Information about end-in activities scheduled beyond the end of the planning horizon is encapsulated in the *guidance heuristic*. This is described in detail in Section 3.5.5.

We now show how to encode these requirements, for the case where $T(e_S)$ and $T(e_E)$ are specified in the plan. First we consider convex feasible regions, before generalizing to non-convex feasible regions.

Convex Feasible Regions

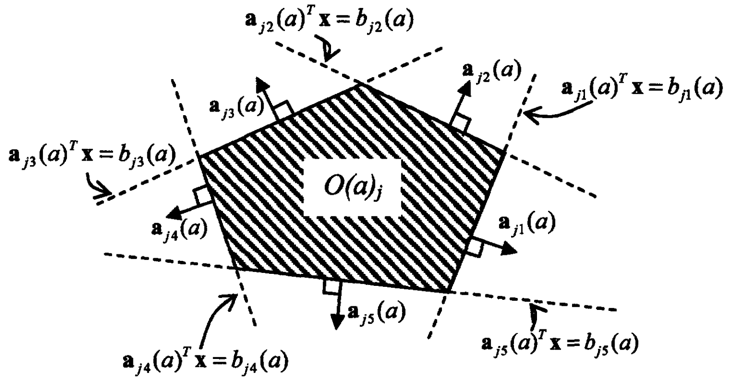
A convex polygonal feasible region $R(a)$ associated with activity a can be defined as a conjunction of linear constraints $\mathbf{a}_l^T(a)\mathbf{x}_c \leq b_l(a)$ for $l = 1, \cdots, N_a$, where $\mathbf{a}_l(a)$ is defined as pointing outwards from the polygonal region. Then $\mathbf{x}_c$ lies within $R(a)$ if and only if all of the constraints are satisfied, as illustrated in Figure 3-2:

$$\mathbf{x}_c \in R(a) \Longleftrightarrow \bigwedge_{l=1,\cdots,N_a} \mathbf{a}_l^T(a)\mathbf{x}_c \leq b_l(a). \tag{3.39}$$

For all remain-in activities $a \in \mathcal{A}_\forall(c_c)$ we now impose the following constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_{c,\tau}{}^{(i)} - b_l(a) \leq M \cdot z_i(c_c) \;\; \forall\tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall l \;\; \forall a \in \mathcal{A}_\forall(c_c), \tag{3.40}$$

where M is a large positive constant, and e_S and e_E are the start and end events for activity a. If $z_i(c_c) = 0$ then every constraint is satisfied for particle i; otherwise (for large enough M), particle i



$\mathbf{a}_{j2}(a)^T\mathbf{x} = b_{j2}(a)$
$\mathbf{a}_{j2}(a)$
$\mathbf{a}_{j1}(a)^T\mathbf{x} = b_{j1}(a)$
$\mathbf{a}_{j3}(a)$
$\mathbf{a}_{j3}(a)^T\mathbf{x} = b_{j3}(a)$
$O(a)_j$
$\mathbf{a}_{j4}(a)$
$\mathbf{a}_{j1}(a)$
$\mathbf{a}_{j5}(a)$
$\mathbf{a}_{j4}(a)^T\mathbf{x} = b_{j4}(a)$
$\mathbf{a}_{j5}(a)^T\mathbf{x} = b_{j5}(a)$

Figure 3-3. Two-dimensional obstacle $O_j(a)$ modeled as a convex polyhedron. The vectors $\mathbf{a}_{j1}(a), \cdots, \mathbf{a}_{jN_{aj}}(a)$ are the unit outward normals to the N_{aj} line segments that define the obstacle.

is unconstrained. For convex regions $R(a)$, (3.40) therefore encodes the following implication:

$$z_i(c_c) = 0 \implies \mathbf{x}_{c,\tau}{}^{(i)} \in R(a) \;\; \forall \tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall a \in \mathcal{A}_\forall, \tag{3.41}$$

as required. For all end-in activities $a \in \mathcal{A}_E$ we impose the following constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_{c,\tau}{}^{(i)} - b_l(a) \leq M \cdot z_i(c_c) \;\; \forall \tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall l \;\; \forall a \in \mathcal{A}_E(c_c), \tag{3.42}$$

where M is a large positive constant and e_E is the end event of activity a. If $z_i(c_c) = 0$ then every constraint is satisfied for particle i, otherwise particle i is unconstrained. For convex regions $R_E(a)$, (3.42) therefore encodes the following implication:

$$z_i(c_c) = 0 \implies \mathbf{x}_{c,\tau} \in R_E(a) \;\; \forall \tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall a \in \mathcal{A}_\forall(c_c), \tag{3.43}$$

as required.

Collectively, the constraints specified in this section ensure that $z_i(c_c) = 0$ indicates that all activities in c_c (both remain-in and end-in) with *convex* feasible regions are satisfied.

Non-convex Feasible Regions

A polytopic non-convex feasible region can be described as the complement of a number of polygonal infeasible regions, or obstacles. In other words, at time t_τ the state $\mathbf{x}_{c,\tau}$ is in the region if and only if all obstacles are avoided for all time steps. For each activity a with non-convex feasible region $R(a)$, we define a set $O(a) = \left\langle O(a)_1, \ldots, O(a)_{M_a} \right\rangle$ of obstacles that collectively form the complement of $R(a)$.

As noted by [118], avoidance of a polygonal obstacle can be expressed in terms of a disjunction of linear constraints. That is, the system state at time t_τ, $\mathbf{x}_{c,\tau}$, avoids the obstacle $O_j(a)$ shown in Figure 3-3 if and only if:

$$\bigvee_{l=1,\ldots,N_{aj}} \mathbf{a}_{jl}^T(a)\mathbf{x}_{c,\tau} \geq b_{jl}(a). \tag{3.44}$$

In a similar manner to [118], we introduce binary variables $d_{ij\tau l}(a) \in \{0,1\}$; here $d_{ij\tau l}(a) = 0$ indicates that constraint l of obstacle $O_j(a)$ is satisfied by particle i at a given time step τ. The constraints:

$$\begin{gathered} \mathbf{a}_{jl}^T(a)\mathbf{x}_{c,\tau}{}^{(i)} - \mathbf{b}_{jl}(a) + Md_{i\tau jl}(a) \geq 0 \\ \forall a \in \mathcal{A}_\forall(c_c) \;\; \forall i \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S),T(e_E)] \;\; \forall j \;\; \forall l, \end{gathered} \tag{3.45}$$

and:

$$\mathbf{a}_{jl}^T(a)\mathbf{x}_{c,\tau}{}^{(i)} - \mathbf{b}_{jl}(a) + Md_{i\tau jl}(a) \geq 0 \tag{3.46}$$

$$\forall a \in \mathcal{A}_E(c_c) \;\; \forall i \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall j \;\; \forall l, \tag{3.47}$$

specify that $d_{ij\tau l}(a) = 0$ implies that constraint l in obstacle $O_j(a)$ is satisfied by particle i at time step τ. Again M is a large positive constant.

We now introduce binary variables $e_{ij\tau}(a) \in \{0,1\}$ where $e_{ij\tau}(a) = 0$ indicates that obstacle $O_j(a)$ is avoided by particle i at time step τ. The constraints:

$$\begin{gathered} \sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq Me_{i\tau j}(a) \\ \forall a \in \mathcal{A}_\forall(c_c) \;\; \forall i \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S),T(e_E)] \;\; \forall j, \end{gathered} \tag{3.48}$$

and:

$$\begin{gathered} \sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq Me_{i\tau j}(a) \\ \forall a \in \mathcal{A}_E(c_c) \;\; \forall i \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall j, \end{gathered} \tag{3.49}$$

specify that $e_{i\tau j}(a) = 0$ implies that at least one constraint in obstacle $O_j(a)$ is satisfied by particle i at time step τ, where N_{aj} is the number of edges in obstacle $O_j(a)$. This in turn implies that obstacle $O_j(a)$ is avoided by particle i at time step τ.

In addition we impose the following constraint for all remain-in activities:

$$\sum_{j=1}^{M_a} e_{i\tau j}(a) \leq M \cdot z_i(c_c) \;\; \forall i \;\; \forall j \;\; \forall a \in \mathcal{A}_\forall(c_c) \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S),T(e_E)], \tag{3.50}$$

where M_a is the number of obstacles for activity a. Collectively the constraints (3.45), (3.48) and (3.50) encode the following implication, for remain-in activities a and non-convex feasible regions $R_\forall(a)$:

$$z_i(c_c) = 0 \implies \mathbf{x}_{c,\tau}{}^{(i)} \in R_\forall(a) \;\; \forall i \;\; \forall j \;\; \forall \tau \in \{0, \dots, N_p\} \;\; s.t. \;\; t_\tau \in [T(e_S), T(e_E)] \;\; \forall a \in \mathcal{A}_\forall(c_c), \tag{3.51}$$

as required. In other words, $z_i(c_c) = 0$ indicates that all remain-in activities in chance constraint c_c are satisfied. Turning our attention now to end-in activities a, we impose the constraint:

$$\sum_{j=1}^{M_a} e_{i\tau j}(a) \leq M \cdot z_i(c_c) \;\; \forall a \in \mathcal{A}_E(c_c) \;\; \forall i \;\; \forall \tau \in \{0, \dots, N_p\} \;\; s.t. \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]. \tag{3.52}$$

Collectively the constraints (3.46), (3.49) and (3.52) encode the following implication, this time for end-in activities a and non-convex regions $R_E(a)$:

$$z_i(c_c) = 0 \implies \left(\mathbf{x}_{c,\tau} \in R_E(a) \;\; \forall \tau \in \{0, \dots, N_p\} \;\; s.t. \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E)\right]\right), \tag{3.53}$$

as required. In other words, $z_i(c_c) = 0$ indicates that all end-in activities in chance constraint c_c are satisfied.

Collectively, the constraints specified in this section ensure that $z_i(c_c) = 0$ indicates that all activities in c_c (both remain-in and end-in) with *non-convex* feasible regions are satisfied.

3.5.3 Expected State Constraints

In accordance with (3.31), for each expected state constraint $c_m \in \mathcal{G}_m$ we must ensure that for the expected state, approximated using the sample mean of the particle set, all activities succeed in the set $\mathcal{A}(c_m)$. We use $\mathbf{x}_{c,\tau}{}^{(m)}$ to denote the sample mean:

$$\mathbf{x}_{c,\tau}{}^{(m)} \triangleq \sum_{i=1}^{N} \mathbf{x}_{c,\tau}{}^{(i)} = \sum_{i=1}^{N} \left(\sum_{j=0}^{\tau-1} A^{\tau-j-1} B(\mathbf{u}_j + \nu_j^{(i)}) + A^\tau \mathbf{x}_{c0}^{(i)} \right). \tag{3.54}$$

Expected state constraints can then be handled in a similar manner to chance constraints, except that expected state constraints *require* the corresponding activities to succeed, but only for the expected state $\mathbf{x}_{c,t}{}^{(m)}$. This is in contrast to chance constraints, which only require activities to succeed for some minimum number of particles. For convex regions the development is the same, except that, for end-in activities, instead of (3.42) we now impose the constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_{c,\tau}{}^{(m)} - b_l(a) \leq 0 \;\; \forall l \; \forall a \in \mathcal{A}_E(c_m) \;\; \forall \tau \in \{0, \dots, N_p\} \;\; s.t. \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]. \tag{3.55}$$

Note that the right hand side of 3.55 is zero, meaning that the convex state constraint *must* be satisfied by the expected state. By constrast, the right hand side of the chance-constrained version (3.42) is only zero if $z_i(c_c) = 0$ is zero; in other words the state constraint must only hold if $z_i(c_c)$ is zero.

For remain-in activities, instead of (3.40), we now impose the constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_{c,\tau}{}^{(m)} - b_l(a) \leq 0 \quad \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall l \;\; \forall a \in \mathcal{A}_\forall(c_m). \tag{3.56}$$

In the case of non-convex regions, instead of the constraints (3.45) and (3.46) we constrain the sample mean $\mathbf{x}_{c,\tau}{}^{(m)}$ as follows:

$$\begin{gathered}\mathbf{a}_{jl}^T(a)\mathbf{x}_{c,\tau}{}^{(m)} - \mathbf{b}_{jl}(a) + Md_{\tau jl}(a) \geq 0 \\ \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall j \;\; \forall l \;\; \forall a \in \mathcal{A}_\forall(c_m),\end{gathered} \tag{3.57}$$

and:

$$\begin{gathered}\mathbf{a}_{jl}^T(a)\mathbf{x}_{c,\tau}{}^{(m)} - \mathbf{b}_{jl}(a) + Md_{\tau jl}(a) \geq 0 \\ \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall j \;\; \forall l \;\; \forall a \in \mathcal{A}_E(c_m).\end{gathered} \tag{3.58}$$

Again, the right hand sides of (3.57) and (3.58) are zero, meaning that the state constraint *must* hold for the expected state if $d_{\tau jl}(a) = 0$. By constrast, the right hand sides of the chance-constrained versions (3.57) and (3.58) are only zero if $z_i(c_c) = 0$ is zero. Note that $d_{\tau jl}$ does not involve the particle number i, indicating that this variable applies to the expected state.

As with (3.48) and (3.49) we impose constraints to ensure that $e_{j\tau} = 0$ implies that at least one constraint in obstacle $O_j(a)$ is satisfied by the expected state:

$$\begin{gathered}\sum_{l=1}^{N_{aj}} d_{\tau jl}(a) - (N_{aj} - 1) \leq Me_{\tau j}(a) \\ \forall a \in \mathcal{A}_\forall(c_c) \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall j,\end{gathered} \tag{3.59}$$

and:

$$\begin{gathered}\sum_{l=1}^{N_{aj}} d_{\tau jl}(a) - (N_{aj} - 1) \leq Me_{\tau j}(a) \\ \forall a \in \mathcal{A}_E(c_c) \;\; \forall \tau \in \{0,\ldots,N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall j.\end{gathered} \tag{3.60}$$

In turn this means that $e_{ij\tau} = 0$ implies that obstacle $O_j(a)$ is avoided by the expected state. Again note that $e_{j\tau}$ does not involve the particle number i, indicating that it applies to the expected state.

We now impose constraints that ensure *all* obstacles are avoided by the expected state. Considering first end-in activities, instead of (3.52) we impose the constraint:

$$\sum_{j=1}^{M_a} e_{\tau j}(a) \leq 0 \;\; \forall \tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right] \;\; \forall j \;\; \forall a \in \mathcal{A}_E(c_m), \quad (3.61)$$

and for remain-in activities, instead of (3.50) we have the constraint:

$$\sum_{j=1}^{M_a} e_{\tau j}(a) \leq 0 \;\; \forall \tau \in \{0, \ldots, N_p\} \; s.t. \; t_\tau \in [T(e_S), T(e_E)] \;\; \forall j \;\; \forall a \in \mathcal{A}_\forall(c_m). \quad (3.62)$$

The constraints (3.61) and (3.62) *require* all activities $\forall a \in \mathcal{A}(c_m)$ to succeed for the expected state, as opposed to requiring that success occurs with a certain probability.

3.5.4 Cost Function

The cost function F is defined to be a function of the control inputs $\mathbf{u}_{0:N_p-1}$, the schedule T, and the system state trajectory $X_{1:N_p}$. Since the system state is uncertain, however, we minimize the expected value of this cost function. We approximate the expectation using the sample mean, since, from (3.3), the sample mean converges to the true expectation as the number of samples tends to infinity. The true expectation is given by:

$$E[F] = \int F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{c,1:N_p}, T) p(\mathbf{x}_{c,1:N_p}) d\mathbf{x}_{c,1:N_p}. \quad (3.63)$$

Since $p(\mathbf{x}_{c,1:N_p})$ can be an arbitrary distribution, this integral is intractable in most cases. The approximated expectation is given by:

$$\hat{F} = \frac{1}{N} \sum_{i=1}^{N} F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{c,1:N_p}{}^{(i)}, T), \quad (3.64)$$

and this expression can be evaluated without integration. As the number of particles tends to infinity, we have:

$$\hat{F} \longrightarrow E[F] \quad (3.65)$$

Furthermore, since we assume that F is a piecewise linear function of the state and control inputs, the expression for $\hat{F}$ in (3.64) is also piecewise linear.

3.5.5 Guidance Heuristic

As noted by [76], end-in activities scheduled outside of the planning horizon are challenging for receding horizon approaches to hybrid execution. Consider the execution of an end-in activity a with region R_E that is scheduled to start during the planning horizon ($T(e_S) \in [t_0, t_{N_p}]$), but a is scheduled to end after the end of the planning horizon, ($T(e_E) > t_{N_p}$). Intuitively, the executive should make sure that the system state makes progress towards R_E during the time period $[T(e_S), t_{N_p}]$. However, none of the constraints encoded in Section 3.5 so far ensure that this is the case. In a similar manner to [76, 14], we, therefore, use a *guidance heuristic* to steer the system state towards R_E when $T(e_S)$ is outside of the planning horizon.

The guidance heuristic employed by [76] is an estimate of the distance between the system state at the final time step in the horizon and the end-in region R_E. This *cost-to-go* is added to the cost function in the overall optimization problem; the resulting optimal solution therefore balances the cost incurred within the horizon with the cost-to-go.

We use a similar approach to [76] except that we estimate the distance between the *expected* system state and the end-in region R_E. An alternative approach is to calculate the *expected distance.* In the case of a nonlinear distance function, however, calculation of this value is more computationally intensive. In the case that the cost-to-go is a linear, the two values are equal. We assume from now that the guidance heuristic estimates the distance between the expected state and R_E.

As in (3.54) we approximate the expected state using the sample mean of the state. Adding the heuristic distance between the approximated expected state at the end of the planning horizon,$\mathbf{x}_{c,N_p}{}^{(m)}$, the objective in the constrained optimization problem is:

$$\text{Minimize } \hat{F} + H(\mathbf{x}_{c,N_p}{}^{(m)}), \tag{3.66}$$

where $H(\cdot)$ is the cost-to-go estimate. [76] gives an extensive discussion of the forms of $H(\cdot)$ that are suitable for MILP optimization. In the simplest case, $H(\cdot)$ is the 1-norm of the distance between $\mathbf{x}_{c,T}$ and R_E. In the case where there are only convex feasible regions in the plan, this is typically sufficient. With non-convex regions, for example in the case of obstacle avoidance in 2-D, better heuristic must be used. [14] and later [76] used a piecewise linear cost-to-go estimate based on a visibility graph planner. This heuristic, however, greatly increases the complexity of the resulting optimization. In our approach we use the results of [14] and [76], except with the expected end state in the heuristic $H(\cdot)$. The expected end state is a linear function of the control inputs; hence any heuristic function proposed by [14] or [76] can also be used for the robust hybrid execution problem in this thesis, while ensuring that the resulting optimization is a valid MILP.

3.5.6 Summary

In Sections 3.5.1 through 3.5.5 we encoded the constrained optimization problem resulting from the Particle Control approach. The constraints are summarized in Table 3.2 and Table 3.3. These constraints are linear, there are integer constraints on some decision variables, and the cost function is piecewise linear in the decision variables. Furthermore, the particles $\mathbf{x}_{c,1:T}{}^{(i)}$ and the approximate expected state trajectory $\mathbf{x}_{c,1:T}{}^{(m)}$ are linear functions of the control inputs. Hence the overall optimization is a Mixed Integer Linear Program. This means that it can be solved to global optimality using efficient commercially-available solvers.

Description	Constraint
For at most a fraction $\delta(c_c)$ of particles does any activity in $\mathcal{A}(c_c)$ fail.	$\frac{1}{N}\sum_{i=1}^{N} z_i(c_c) \leq \delta(c_c)$
	$\forall i \;\; \forall \tau \in \{0,\dots,N_p\}$ *s.t.* $t_\tau \in [T(e_S), T(e_E)]$:
$z_i(c_c) = 0$ implies all remain-in activities in $\mathcal{A}(c_c)$ with convex $R_\forall$ succeed for particle i.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \leq M \cdot z_i(c_c) \; \forall l \; \forall a \in \mathcal{A}_\forall(c_c)$
$z_i(c_c) = 0$ implies all remain-in activities in $\mathcal{A}(c_c)$ with non-convex $R_\forall$ succeed for particle i.	$\sum_{j=1}^{M_a} e_{i\tau j}(a) \leq M \cdot z_i(c_c) \; \forall a \in \mathcal{A}_\forall(c_c)$
	$\forall i \;\; \forall \tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]$:
$z_i(c_c) = 0$ implies all end-in activities in $\mathcal{A}(c_c)$ with convex R_E succeed for particle i.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \leq M \cdot z_i(c_c) \; \forall l \; \forall a \in \mathcal{A}_E(c_c)$
$z_i(c_c) = 0$ implies all end-in activities in $\mathcal{A}(c_c)$ with non-convex R_E succeed for particle i.	$\sum_{j=1}^{M_a} e_{i\tau j}(a) \leq M \cdot z_i(c_c) \; \forall a \in \mathcal{A}_E(c_c)$
	$\forall i \;\; \forall \tau \in \{0,\dots,N_p\}$ *s.t.* $t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]$
$d_{ij\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by particle i at time τ (end-in).	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(i)} - \mathbf{b}_{jl}(a) + M d_{i\tau jl}(a) \geq 0 \; \forall a \in \mathcal{A}_E(c_c) \; \forall j \;\; \forall l$
$e_{i\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by particle i at time t_τ (end-in).	$\sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq M e_{i\tau j}(a) \; \forall a \in \mathcal{A}_E(c_c) \; \forall j$
	$\forall i \;\; \forall \tau \in \{0,\dots,N_p\}$ *s.t.* $t_\tau \in \left[T(e_S), T(e_E)\right]$
$d_{ij\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by particle i at time τ (remain-in).	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(i)} - \mathbf{b}_{jl}(a) + M d_{i\tau jl}(a) \geq 0 \; \forall a \in \mathcal{A}_\forall(c_c) \; \forall j \;\; \forall l$
$e_{i\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by particle i at time t_τ (remain-in).	$\sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq M e_{i\tau j}(a) \; \forall a \in \mathcal{A}_\forall(c_c) \; \forall j$
Integer constraints	$z_i(c_c) \in \{0,1\}$ $d_{ij\tau l}(a) \in \{0,1\}$ $e_{i\tau j}(a) \in \{0,1\}$

Table 3.2. Fixed-schedule particle control: Constraints imposed for each chance constraint $c_c \in \mathcal{G}_c$

3.6 Simulation Results: Linear Systems

In this section we demonstrate the PC-LIN approach in simulation using two scenarios. In Section 3.6.1 the method is used to control an aircraft within a flight envelope in heavy turbulence, while in Section 3.6.3 the method is used to generate robust, optimal paths for an aircraft operating in an environment containing obstacles. In both cases the plant is modeled using a linear model as in (3.33).

3.6.1 Particle Control in a Convex Feasible Region

The PC-LIN method was used to generate robust predictive control inputs for an aircraft performing an altitude change maneuver in turbulence. In this scenario, successful execution means that the

Description	Constraint
	$\forall i \ \forall \tau \in \{0, \ldots, N_p\}$ *s.t.* $t_\tau \in [T(e_S), T(e_E)]$:
All remain-in activities in $\mathcal{A}(c_m)$ with convex $R_\forall$ succeed for expected state.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \leq 0 \ \forall l \ \forall a \in \mathcal{A}_\forall(c_m)$
All remain-in activities in $\mathcal{A}(c_m)$ with non-convex $R_\forall$ succeed for expected state.	$\sum_{j=1}^{N_{aj}} e_{\tau j}(a) \leq 0 \ \forall a \in \mathcal{A}_\forall(c_m)$
	$\forall \tau \in \{0, \ldots, N_p\}$ *s.t.* $t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]$:
All end-in activities in $\mathcal{A}(c_m)$ with convex R_E succeed for expected state.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \leq 0 \ \forall l \ \forall a \in \mathcal{A}_E(c_m)$
All end-in activities in $\mathcal{A}(c_c)$ with non-convex R_E succeed for expected state.	$\sum_{j=1}^{N_{aj}} e_{\tau j}(a) \leq 0 \ \forall a \in \mathcal{A}_E(c_c)$
	$\forall \tau \in \{0, \ldots, N_p\}$ *s.t.* $t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]$:
$d_{j\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by expected state at time τ (end-in).	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(m)} - \mathbf{b}_{jl}(a) + Md_{\tau jl}(a) \geq 0 \ \forall a \in \mathcal{A}_E(c_c) \ \forall j \ \ \forall l$
$e_{\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by expected state at time t_τ (end-in).	$\sum_{l=1}^{N_{aj}} d_{\tau jl}(a) - (N_j - 1) \leq Me_{\tau j}(a) \ \forall a \in \mathcal{A}_E(c_c) \ \forall j$
	$\forall \tau \in \{0, \ldots, N_p\}$ *s.t.* $t_\tau \in \left[T(e_S), T(e_S)\right]$:
$d_{j\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by expected state at time τ (remain-in).	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(m)} - \mathbf{b}_{jl}(a) + Md_{\tau jl}(a) \geq 0 \ \forall a \in \mathcal{A}_\forall(c_c) \ \forall j \ \ \forall l$
$e_{\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by expected state at time t_τ (remain-in).	$\sum_{l=1}^{N_{aj}} d_{\tau jl}(a) - (N_j - 1) \leq Me_{\tau j}(a) \ \forall a \in \mathcal{A}_\forall(c_c) \ \forall j$
Integer constraints	$d_{j\tau l}(a) \in \{0, 1\}$ $e_{\tau j}(a) \in \{0, 1\}$

Table 3.3. Fixed-schedule particle control: Constraints imposed for each expected state constraint $c_m \in \mathcal{G}_m$

aircraft remains within a defined flight envelope, which forms a convex region in the space of state trajectories. An example is shown in Figure 3-4.

Disturbances due to turbulence have been studied extensively and are modeled as stochastic noise processes that are far from Gaussian[11]. In this work we use the Dryden turbulence model described in Military Specification MIL-F-8785C [1]. We assume heavy turbulence, as defined in MIL-F-8785C, with a low-altitude wind velocity of $25m/s$. Designing control inputs robust to such a non-Gaussian disturbance process is a highly challenging problem, however the Particle Control formulation is able to deal with this example naturally.

For the aircraft model, we use the linearized, discrete time longitudinal dynamics of a Boeing 747 travelling at Mach 0.8. Time increments of two seconds were used. Since the angle of attack of most aircraft is low in normal operation, linearizing the dynamics about the equilibrium state or *trim state* of the aircraft yields a good approximation to the true dynamics, which can be used to develop control algorithms [88]. Consistent with prior approaches to robust predictive control[52, 110], we assume that there is an inner-loop controller issuing elevator commands, which is an altitude-hold autopilot. The aim of the predictive controller then, is to issue altitude commands to the autopilot in an optimal, robust manner, so that the aircraft breaks the flight envelope with a probability of at most δ. Optimality is defined in terms of fuel consumption, and we assume the following relationship between fuel consumption F and elevator angle at time t, a_t:

$$F = \sum_{t=0}^{T-1} |a_\tau| \tag{3.67}$$

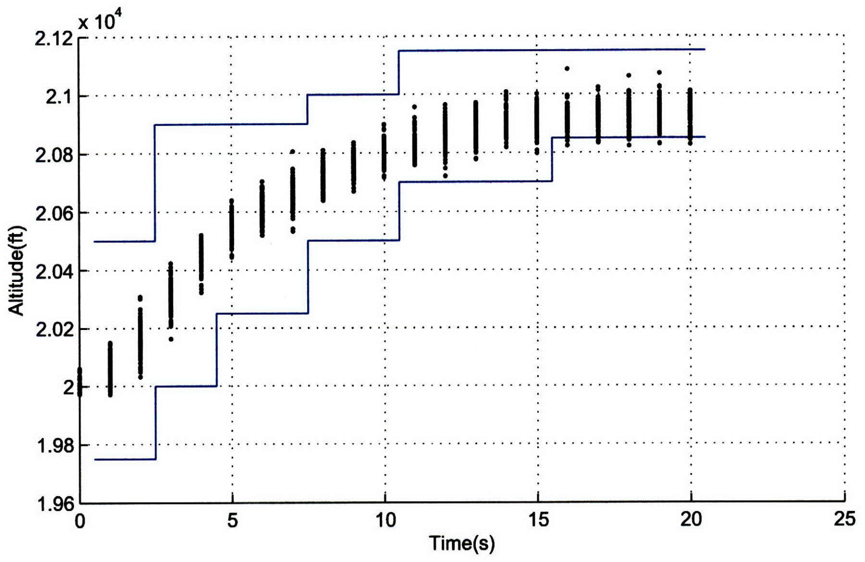
Since we assume an inner loop controller issues elevator angle commands, a_τ depends on the disturbances that act on the aircraft; for example, if a fixed altitude is commanded by the predictive controller, the autopilot will issue larger elevator commands in order to reject large disturbances than for smaller ones. Since the disturbances are stochastic, we cannot directly optimize the fuel consumption defined in (3.67). We can, however, optimize the expected fuel consumption, as described in Section 3.4.

We assume a maximum elevator deflection of 0.5 radians, due to actuator saturation. Again, since elevator deflection depends on stochastic disturbances, we cannot prevent actuator saturation with absolute certainty. We can, however, define a chance constraint that, approximated using the Particle Control method, ensures that actuator saturation occurs with at most a given probability. In the results shown here we define this probability to be zero, thereby ensuring that saturation occurs with approximately zero probability.

The initial state distribution was generated using a particle filter. The particle filter tracked the system state for ten time steps leading up to time step $\tau = 0$ while the aircraft held altitude. Observations of pitch rate and altitude were made subject to additive Rayleigh noise[108]. This non-Gaussian noise means that a particle filter will typically outperform a Kalman Filter. The number of particles used for estimation was the same as that used for control.

Figure 3-4 shows two typical solutions generated by the PC-LIN algorithm for 100 particles. In the first solution, the desired probability of error is 0.1, while in the second, the desired probability of error is 0.01. It can be seen that in the case of the lower value, the path taken by the particles is further away from the edges of the flight envelope, causing one particle only to fall outside of the envelope. This more conservative plan has a greater fuel cost of 3.82, compared to the less conservative one, which has a fuel cost of 3.62. Hence conservatism can be traded off against fuel efficiency; the empirical relationship between the two for this scenario is shown in Figure 3-5.

The Particle Control method solves an approximated stochastic control problem. The accuracy of the approximation was assessed by calculating the true probability of failure for a given plan. This probability was calculated by carrying out a very large number of random simulations. Since the generated plan depends on the values sampled from the various probability distributions, 20 plans were generated for each scenario. Figure 3-6 shows the results for a desired probability of failure of 0.1. It can be seen that the mean probability of error gets closer to the desired value as the number of particles increases, and that the variance decreases. For 100 particles, the approximation is close; the mean is 0.104 and the standard deviation is 0.024. Hence the PC-LIN algorithm can generate optimal solutions to problems that are close to the full stochastic control problem.



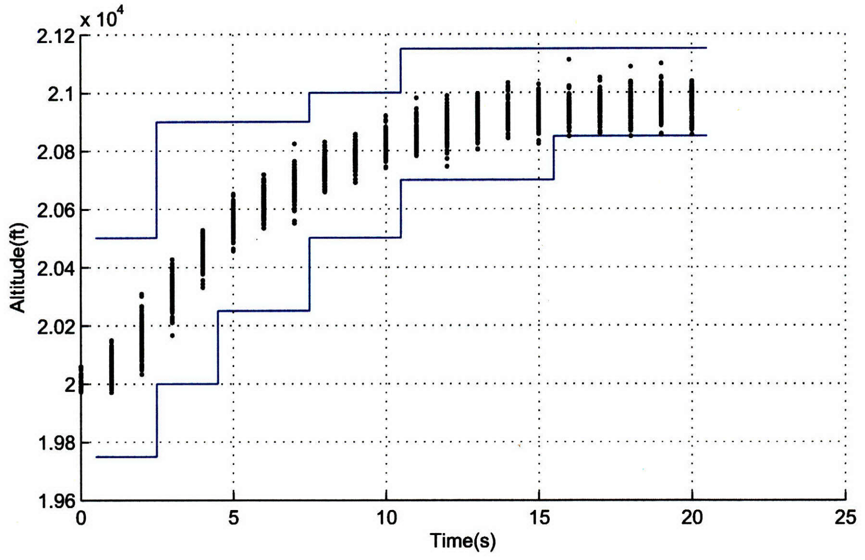


Figure 3-4. Path of particles for two typical solutions to the flight envelope scenario. 100 particles were used. Top: The desired probability of failure is 0.1, and ten particles fall outside of the flight envelope. Bottom: The desired probability of failure is 0.01, and one particle falls outside of the flight envelope, at $t = 18s$.

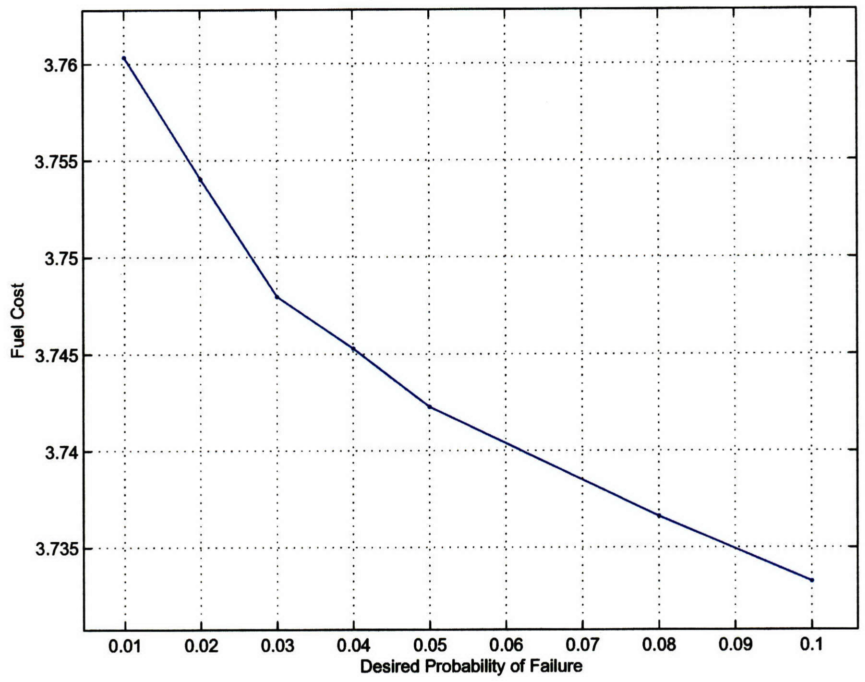


Figure 3-5. Mean fuel cost against desired probability of failure. The mean is taken over 20 different solutions to the same problem. As the allowed probability of failure increases, the plan becomes less conservative and the cost decreases.

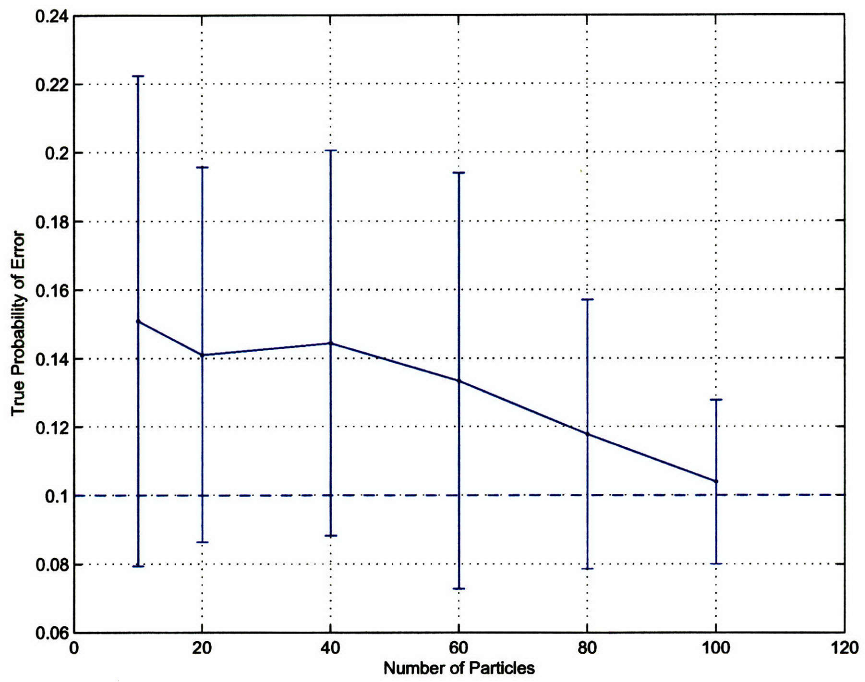


Figure 3-6. True probability of failure against number of particles used to design control inputs. The desired probability of error was 0.1, shown as the dashed line. The results shown are for 20 sets of designed control inputs, with the dots denoting the mean and the error bars denoting the standard deviation of the probability of error. As the number of particles increases, the mean probability of error approaches the desired probability of error and the variance decreases.

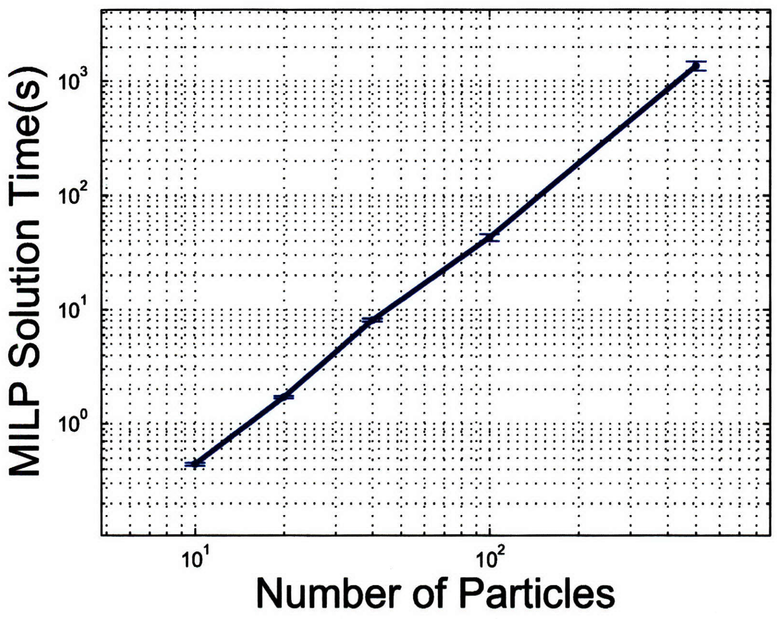


Figure 3-7. MILP solution time for PC-LIN algorithm with Boeing 747 example. The specified maximum probability of failure was 0.1. Note that the times shown here are those taken to reach a provable global optimum; in many cases a feasible solution at or close to the global optimum can be found in significantly less time.

3.6.2 Comparison with Monte Carlo Markov Chain Methods

An alternative approach to chance-constrained control with non-Gaussian uncertainty and continuous decision variables was previously proposed by [77]. This approach uses a Monte Carlo Markov Chain (MCMC) framework[115] to find an approximately optimal control input through simulation-based optimization. This works by estimating the distribution of a random variable, which is constructed so that the peaks of the distribution coincide with the optimal decision value. The estimation process is then carried out by MCMC[115]. In this section we first discuss differences between the two approaches, and then provide a performance comparison[3].

Approaches based on MCMC have two main advantages over the PC-LIN algorithm. First, they are not restricted to linear system dynamics and second, the distributions of the uncertain variables can be a function of the state and control inputs. The efficiency of PC-LIN comes from converting a stochastic problem to a deterministic Mixed Integer Linear Program, and this requires both linear system dynamics and the ability to sample all uncertain variables before optimization begins. We will demonstrate in this section that these restrictions give PC-LIN a huge improvement in performance over methods based on MCMC. The Particle Control approach is also able to encode constraints on the state distribution explicitly, while using the MCMC approach in [77], chance constraints must be dealt with using a penalty function formulation; this introduces sub-optimality in the solution. Particle Control can deal explicitly with not only chance constraints, but also constraints on the

[3]This work was carried out in conjunction with Masahiro Ono.

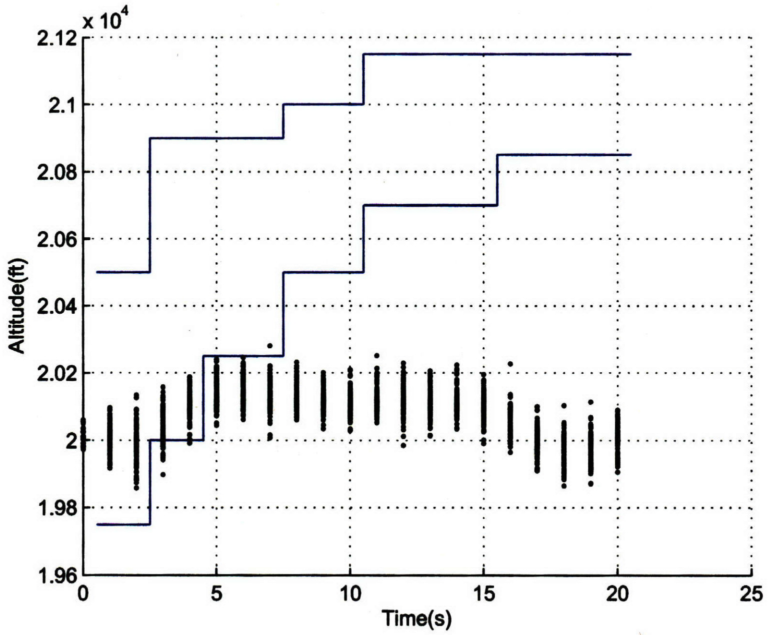


Figure 3-8. Typical solution after 2×10^6 iterations of MCMC approach. Even after more than 7 hours of computation, the approach is unable to find a feasible solution. In this case, the desired maximum probability of failure was 0.1, but Monte Carlo simulations show that the probability of failure is very close to 1.

expected system state or conditional expectations over the system state. This means that a richer set of problem encodings can be handled by the Particle Control approach.

In [77], the authors considered an Air-Traffic Control (ATC) scenario with *only two continuous decision variables*. They report optimization times of approximately 20 hours in order to obtain solutions that are relatively close to optimal. The PC-LIN algorithm therefore gives orders of magnitude faster solution time for much larger problems. In order to compare the performance of the two approaches empirically for large problems, we implemented the approach described by [77] and applied it to the Boeing 747 altitude envelope problem described in Section 3.6.1. In this problem there are 20 continuous decision variables. Figure 3-8 shows the best solution found using the MCMC approach after 2×10^6 iterations, or 7 hours of computation. After this many iterations, the solution is far from feasibility, with a probability of failure of approximately one. By contrast, the PC-LIN algorithm solves this problem in seconds for reasonable levels of approximation error (see Figures 3-7 and 3-6). We also used the MCMC approach starting from a known feasible solution, again stopping the optimization after 2×10^6 iterations. in this case the resulting solution was far from the optimum found by the PC-LIN approach; the MCMC approach gave a fuel cost of 6.08, while the PC-LIN solution, using 100 particles, gave an average cost of 3.73.

3.6.3 Particle Control in a Non-Convex Feasible Region

The new PC-LIN algorithm was also applied to a UAV path planning scenario with obstacles, wind and uncertain localization. In this scenario, successful execution means that the UAV is in the goal region at the end of the time horizon, and that the UAV avoids all obstacles at all time steps within the horizon.

Previous work[118] has shown that an aircraft operating in a two-dimensional environment can be modeled as a discrete-time linear system in the form of (3.33). We use the same aircraft model and assume a maximum aircraft velocity of $50m/s$, time steps of $1s$, and a planning horizon of $10s$.

As in Section 3.6.1, the goal of the robust control algorithm is to design a sequence of control inputs so that the probability of failure is at most δ. Uncertain disturbances, due to wind, act on the UAV. We use the Dryden wind model with a low-altitude wind speed of $15m/s$ and light turbulence, as defined in MIL-F-8785C. We assume an inner-loop controller that acts to reject disturbances. Also, as in Section 3.6.1, uncertainty in localization leads to uncertainty in the initial position of the UAV. The obstacle map used is shown in Figure 3-9. Optimality was again defined in terms of fuel consumption, which we assume is related to input acceleration as follows:

$$F = \sum_{t=0}^{T-1} |u_{x,\tau}| + |u_{y,\tau}|. \tag{3.68}$$

Here $u_{x,\tau}$ and $u_{y,\tau}$ are the commanded accelerations at time step τ in the x and y directions respectively. In order to reduce the complexity of the resulting MILP problem we employed heuristic pruning techniques to reduce the number of particles and number of obstacles considered that are considered, while still guaranteeing an optimal, feasible solution to the original problem. In this thesis we do not describe these techniques in detail; a principled development of this topic is the subject of future research.

Results for the scenario are shown in Figures 3-9 and 3-10. 50 particles were used for these examples. Figure 3-9 shows that if a probability of failure of 0.04 or above is acceptable, the planned path of the UAV can go through the narrow corridor at $(-50, 200)$. It can be seen that exactly two particles collide with the obstacles as expected. For a lower probability of failure, however, the planned path is more conservative as shown in Figure 3-10. This path avoids the narrow corridor at the expense of fuel efficiency.

These results show that the new PC-LIN algorithm is effective in designing optimal paths for an aircraft that are robust to probabilistic uncertainty.

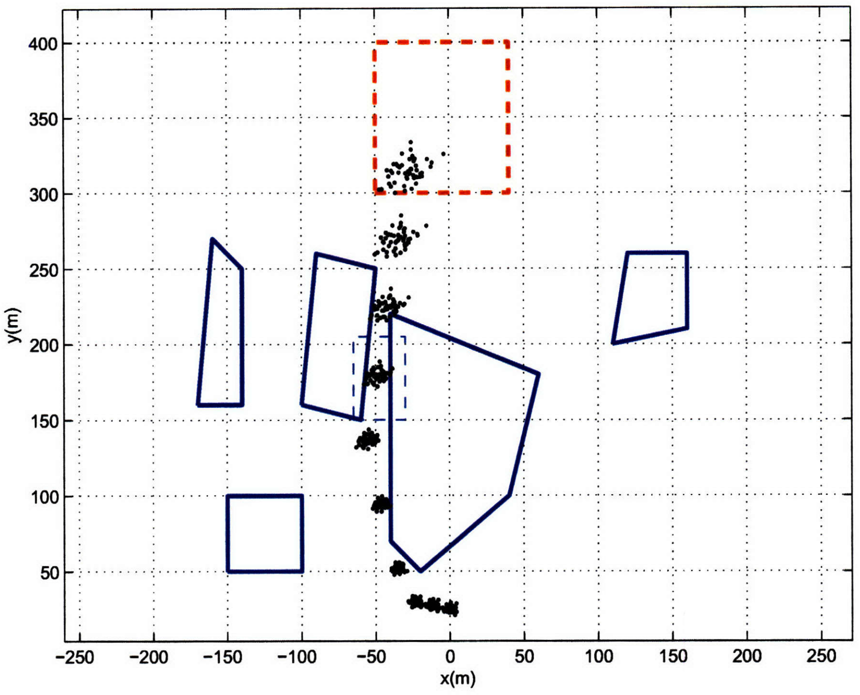


Figure 3-9. Path of particles for typical solution to UAV path planning problem for a probability of failure of 0.04. Obstacles are in blue, while the goal is in red and dashed. At this level of conservatism, the aircraft is able to pass through the narrow corridor highlighted by the thin blue dashed box. The PC-LIN algorithm ensures that at most two particles out of fifty collide with the obstacles. This solution has a fuel cost of 73.98.

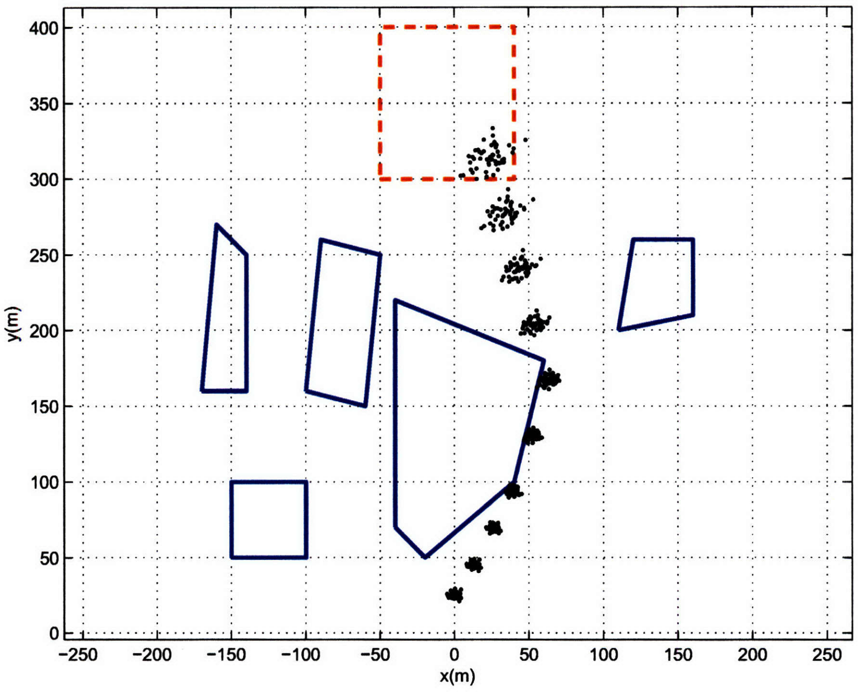


Figure 3-10. Path of particles for typical solution to UAV path planning problem for a probability of failure of 0.02. Obstacles are in blue, while the goal is in red and dashed. At this level of conservatism, the aircraft no longer passes through the narrow corridor, but goes around the largest obstacle. This solution has a fuel cost of 74.72. Hence the reduced probability of failure comes at the expense of fuel.

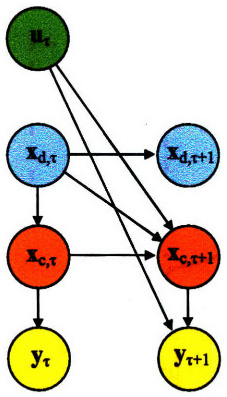


Figure 3-11. Bayes Network showing conditional dependencies between variables at time steps τ and $\tau + 1$ in a Jump Markov Linear System. Note that the discrete state does not depend on the control input or the continuous state, unlike in a general PHA.

3.7 Particle Control for Jump Markov Linear Systems

We now show how the particle control method described in Section 3.4 can be extended to robust control of Jump Markov Linear Systems (JMLS), to give the PC-JMLS algorithm. JMLS are an important subset of Probabilistic Hybrid Automata for which the continuous dynamics are linear, and mode transitions are Markovian; that is, transitions that are not conditioned on the continuous state. JMLS are effective representations of model continuous systems subject to intermittent failures; stochastic mode transitions are used to model faults, and the continuous dynamics depend on which fault (if any) has occurred. The Bayes Network representation of a JMLS is shown in Figure 3-11. The key idea behind extending particle control to JMLS is for the sampled variables for each particle to include discrete mode sequences as well as continuous disturbances. Given a discrete mode sequence and a sequence of continuous disturbances, the future system state trajectory is a known deterministic function of the control inputs. Each particle provides a sample of the future hybrid discrete-continuous state trajectory as a function of the control inputs.

We define a JMLS as having the following dynamics:

$$\mathbf{x}_{c,\tau+1} = A(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau} + B(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \omega_\tau, \tag{3.69}$$

where $\mathbf{x}_{d,t}$ is a Markov chain that evolves according to a transition matrix V such that:

$$p(\mathbf{x}_{d,\tau+1} = j | \mathbf{x}_{d,\tau} = i) = V_{ij}. \tag{3.70}$$

The system state is a hybrid of the discrete state $\mathbf{x}_{d,\tau}$, which we also refer to as the *mode*, and the continuous state $\mathbf{x}_{c,\tau}$. The variable ω_τ is a disturbance process whose distribution can take any form; the distribution itself is known, however. A key property of JMLS is that a mode sequence $\mathbf{x}_{d,0:N_p-1}$ of arbitrary length N_p is independent of the control inputs and the continuous state, and hence:

$$p(\mathbf{x}_{c,1:N_p}, \mathbf{x}_{d,0:N_p-1} | \mathbf{u}_{0:N_p-1}) = p(\mathbf{x}_{c,1:N_p} | \mathbf{x}_{d,0:N_p-1}, \mathbf{u}_{0:N_p-1}) p(\mathbf{x}_{d,0:N_p-1}), \tag{3.71}$$

where $p(\mathbf{x}_{d,0:N_p-1})$ is known. Exploiting this property, we first generate samples of the mode sequence $\mathbf{x}_{d,0:N_p-1}$, and for each sample $x^{(i)}_{d,0:N_p-1}$, we generate samples of the continuous disturbance variables. While there are $|\mathcal{X}_d|^{N_p}$ different mode sequences, sampling from the distribution $p(\mathbf{x}_{d,0:N_p-1})$ is straightforward due to the Markov property, which means that:

$$p(\mathbf{x}_{d,0:N_p-1}) = \prod_{t=1}^{N_p-1} p(\mathbf{x}_{d,\tau} | \mathbf{x}_{d,\tau-1}). \tag{3.72}$$

We exploit this property in generating a sample from the distribution over sequences $p(\mathbf{x}_{d,0:N_p-1})$ as follows. First we generate a sample $\mathbf{x}_{d,0}{}^{(i)}$ from the initial mode distribution $p(\mathbf{x}_{d,0})$, then we generate a sample from the distribution $p(\mathbf{x}_{d,1} | \mathbf{x}_{d,0}{}^{(i)})$. Continuing in a recursive fashion up to the end of the planning horizon, a sample for the full sequence $\mathbf{x}_{d,0:N_p-1}{}^{(i)}$ is generated.

The PC-JMLS algorithm is described in full in Table 3.4. From the results in Section 3.3 we have convergence of the approximated problem in Table 3.4 to the original stochastic problem as the number of particles tends to infinity.

Constraining Conditional Distributions The chance constraints and expected state constraints in the PC-JMLS algorithm can be imposed on *conditional* state distributions; that is, the distribution of the state conditioned on a particular mode sequence, denoted $p(\mathbf{x}_{c,1:N_p} | \mathbf{x}_{d,0:N_p-1})$, or the distribution of the state conditioned on a particular mode at a particular time step, denoted $p(\mathbf{x}_{c,\tau} | \mathbf{x}_{d,\tau})$. This is useful, for example, when placing constraints on the state distribution regardless of the mode sequence $p(\mathbf{x}_{c,1:N_p})$ is impractical. In the case of the AUV mission, we might constrain the system state to satisfy the *Remain in* `[bloom region]` episode *conditioned on no failures occurring.* On the other hand, the *Remain in* `[safe region]` episode must be satisfied whether or not a failure occurs. A chance constraint on the conditional distribution $p(\mathbf{x}_{c,1:N_p} | \mathbf{x}_{d,0:N_p-1} = \mathbf{x}_{d,0:N_p-1}{}^*)$, where $\mathbf{x}_{d,0:N_p-1}{}^*$ is a specified mode sequence, is:

$$p\Big(s(\mathcal{A}(c_c), \mathbf{x}_{c,1:N_p} | \mathbf{x}_{d,0:N_p-1} = \mathbf{x}_{d,0:N_p-1}{}^*, T) = 1\Big) \leq \delta(c_c). \tag{3.73}$$

3.7.1 MILP Solution of PC-JMLS

We now show that the MILP can be used to solve the deterministic optimization problem posed by the PC-JMLS algorithm (Item 3.7 of Table 3.4). From the definition of JMLS in (3.69) we use

```
Function PcJmlsMain($P$,$\mathcal{M}$,$N$,$N_P$) returns ($\mathbf{u}_{0:N_p-1}$,$T$)
1) Generate $N$ samples of the initial discrete mode $\{\mathbf{x}_{d,0}{}^{(1)}, \ldots, \mathbf{x}_{d,0}{}^{(N)}\}$ according to
   the distribution $p(\mathbf{x}_{d,0})$.
2) For each sample $\mathbf{x}_{d,0}{}^{(i)}$, generate a sample of the initial continuous state
   $\{\mathbf{x}_{c,0}^{(1)}, \ldots, \mathbf{x}_{c,0}^{(N)}\}$ according to $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$.
3) For each sample $\mathbf{x}_{d,0}{}^{(i)}$ generate a sample of the discrete mode sequence $\mathbf{x}_{d,0:N_p-1}^{(i)}$
   according to $p(\mathbf{x}_{d,0:N_p-1:|}\mathbf{x}_{d,0:})$.
4) For each sample $\mathbf{x}_{d,0:N_p-1}^{(i)}$ generate a sample of the disturbances $\{\nu_{0:N_p-1}^{(i)}\}$ from the
   distribution $p(\nu_{0:N_p-1}|\mathbf{x}_{d,0:N_p-1}^{(i)})$.
5) Express the distribution of the future state trajectories approximately as a set of $N$
   particles, where each particle $\mathbf{x}_{c,1:N_p}^{(i)}$ corresponds to the continuous state trajectory
   given a particular set of samples $\{\mathbf{x}_0^{(i)}, \mathbf{x}_{d,0:N_p-1}^{(i)}, \nu_{0:N_p-1}^{(i)}\}$. Each particle depends
   explicitly on the control inputs $\mathbf{u}_{0:N_p-1}$, which are yet to be generated.
6) Approximate the expected state constraints and chance constraints in terms of the
   generated particles, is the same manner as the Particle Control algorithm (Table 3.1).
7) Approximate the expected objective function in terms of particles (Table 3.1).
8) Solve deterministic constrained optimization problem for control inputs $\mathbf{u}_{0:N_p-1}$ and
   schedule $T$.
```

Table 3.4. PC-JMLS algorithm for robust execution of JMLS

(3.24) to obtain the following expression for the continuous state sequence for particle i:

$$\mathbf{x}_{c,\tau}^{(i)} = \sum_{j=0}^{\tau-1} \Big(\prod_{l=1}^{\tau-j-1} A(\mathbf{x}_{d,l}{}^{(i)}) \Big) \Big(B(\mathbf{x}_{d,j}{}^{(i)}) \mathbf{u}_j + \nu_j^{(i)} \Big) + \Big(\prod_{l=1}^{\tau} A(\mathbf{x}_{d,l}{}^{(i)}) \Big) \mathbf{x}_{c,0}^{(i)}. \tag{3.74}$$

Note that this is a linear function of the control inputs, and that $\mathbf{x}_{c,0}^{(i)}$, $\nu_j^{(i)}$ and $\mathbf{x}_{d,l}{}^{(i)}$ are all known (sampled) values. Hence each particle $\mathbf{x}_{c,1:N_p}^{(i)}$ is linear in the control inputs. The approximate chance constraints described in detail in Section 3.5 are Mixed Integer Linear constraints on the control inputs. Furthermore expected state constraints are linear constraints on the control inputs. In addition, a piecewise-linear cost function defined over the control inputs and the system state can be approximated as described in Section 3.5 as a piecewise-linear function of the control inputs. For Jump Markov Linear Systems, therefore, the deterministic optimization problem that results from the PC-JMLS algorithm can be solved using Mixed Integer Linear Programming.

3.7.2 Summary

In this section we extended the PC-LIN algorithm to enable robust control of Jump Markov Linear Systems, to give the PC-JMLS algorithm. In Section 3.11 we demonstrate, in simulation, that the PC-JMLS algorithm can be used for robust control of a ground vehicle with failure-prone brakes. This results show, however, thaat the PC-JMLS algorithm often fails to take into account very low probability events, such as brake failure. In Section 3.10 we present the PC-WEIGHTED algorithm, which extends the PC-JMLS algorithm to overcome this problem.

3.8 Flexible-Schedule Particle Control for Linear Systems

In this section we extend the Particle Control approach to general Qualitative State Plans, in which schedules are only partially specified, through qualitative temporal constraints. This is an important extension, since the temporal flexibility in a Qualitative State Plan can be exploited to optimize the objective function, or to add robustness. The key idea behind the extension is to introduce binary decision variables that indicate the temporal relation between a given time step and a given activity. This idea was introduced in [76] for an executive that does not explicitly reason about uncertainty; we extend this to execution under stochastic uncertainty. In Section 3.8.1 we encode the qualitative temporal constraints (Def. 8). In Sections 3.8.2 through 3.8.4 we encode the chance constraints in the temporally flexible plan (Def. 11). Finally, in Section 3.8.5 we encode the expected state constraints in the temporally flexible plan.

3.8.1 Temporal Constraints

Recall (from Section 2.2.3) that a simple temporal constraint $c = \langle e_A, e_B, \Delta T^{min}_{e_A \rightarrow e_B}, \Delta T^{max}_{e_A \rightarrow e_B} \rangle$ specifies that the duration from any event e_A to any other event e_B be in the real-valued interval $[\Delta T^{min}_{e_A \rightarrow e_B}, \Delta T^{max}_{e_A \rightarrow e_B}] \subseteq [0, +\infty]$. The time points of events are no longer fixed. We encode each temporal constraint c as follows:

$$\begin{aligned} T(e_E) - T(e_S) &\geq \Delta T^{min}_{e_S \rightarrow e_E} \\ T(e_E) - T(e_S) &\leq \Delta T^{max}_{e_S \rightarrow e_E}. \end{aligned} \tag{3.75}$$

3.8.2 Chance Constraints for Temporally Flexible Plans

We now encode chance constraints for temporally flexible plans, In order to encode the chance constraint c_c, we retain the overall constraint:

$$\frac{1}{N} \sum_{i=1}^{N} z_i(c_c) \leq \delta(c_c). \tag{3.76}$$

The challenge is now to ensure that:

$$z_i(c_c) = 0 \implies s(\mathcal{A}(c_c), \mathbf{x}^{(i)}_{1:N_p}, T) = 0. \tag{3.77}$$

for Qualitative State Plans with simple temporal constraints, where the schedule T is not fixed.

Recall that a *remain in state* R_V activity between events e_S and e_E imposes $\mathbf{x}_\tau \in R_V$ for all time steps $t_\tau \in [T(e_S), T(e_E)]$. Recall also that an end-in activity imposes $\mathbf{x} \in R_E$ at time $T(e_E)$, that is, when event e_E is scheduled. In an identical manner to [76] we approximate this requirement as:

$$\begin{aligned}
&\left(\exists \tau \in \{0, \ldots, N_p\} \;\; \mathbf{x}_\tau \in R_E \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]\right) \\
\vee \;\; & T(e_E) \leq 0 - \frac{\Delta t}{2} \\
\vee \;\; & T(e_E) \geq t_{N_p} + \frac{\Delta t}{2}.
\end{aligned} \tag{3.78}$$

In other words, either $T(e_E)$ is scheduled within the planning horizon, in which case $\mathbf{x}_\tau \in R_E$ at the time step closest to $T(e_E)$, or $T(e_E)$ is scheduled outside the planning horizon.

In order to be encode these constraints when $T(e_S)$ and $T(e_E)$ are not fixed, we need to be able to 1) refer to the times $T(e_S)$ and $T(e_E)$ symbolically, without knowing their assigned values, and 2) express the episode constraint a in terms of the symbolic schedule. In Section 3.8.3 and Section 3.8.4, respectively, we specify constraints to achieve these.

3.8.3 Symbolic Encoding of Events

We must refer to $T(e_S)$ and $T(e_E)$ symbolically, without knowing their assigned values. In particular, we need to encode whether a time step τ falls in the interval $[T(e_S), T(e_E)]$. In order to do so, for each remain-in activity a, we define binary variables $b_{after}(\tau, e_S)$, $b_{before}(\tau, e_E)$ and $b_{during}(\tau, e_S, e_E)$, for all $\tau \in \{0, \ldots, N_p\}$. The binaries $b_{after}(\tau, e_S)$, $b_{before}(\tau, e_E)$ and $b_{during}(\tau, e_S, e_E)$ are predicates that, if zero, indicate that time step τ occurs after e_S (the start of episode a), before e_E (the end of episode a), or during episode a, respectively. N_p is the length of the planning horizon (Def. 25). We encode the predicates using (3.79):

$$\begin{aligned}
t_\tau - T(e_S) &\leq M \cdot (1 - b_{after})(\tau, e_S) \;\; \forall \tau \in \{0, \ldots, N_p\} \;\; \forall e_S \\
T(e_E) - t_\tau &\leq M \cdot (1 - b_{before})(\tau, e_E) \;\; \forall \tau \in \{0, \ldots, N_p\} \;\; \forall e_E \\
b_{during}(\tau, e_S, e_E) &\leq M(b_{after} + b_{before}) \;\; \forall \tau \in \{0, \ldots, N_p\} \;\; \forall e_E \;\; \forall e_S,
\end{aligned} \tag{3.79}$$

which ensures that:

$$t_\tau \in [T(e_S), T(e_E)] \implies b_{during}(\tau, e_S, e_E) = 0, \tag{3.80}$$

as required.

Turning our attention now to end-in activities, we define binary variables $b_{at}(\tau, e_E)$ for each time step $\tau \in \{0, \ldots, N_p\}$ and each event e_E. If $b_{at}(\tau, e_E)$ is zero, then time step τ is the one closest to e_E. We also define binary variables $b_{before}(e_E)$ and $b_{after}(e_E)$ for each event e_E. If $b_{before}(e_E) = 0$, then e_E is scheduled before the planning horizon. If $b_{after}(e_E) = 0$, then e_E is scheduled after the end of the planning horizon. We encode these predicates using the following constraints:

$$\begin{aligned}
t_\tau - \frac{\Delta t}{2} - T(e_E) &\le M \cdot b_{at}(\tau, e_E) \;\; \forall \tau \in \{0, \dots, N_p\} \;\; \forall e_E \\
T(e_E) - t_\tau - \frac{\Delta t}{2} &\le M \cdot b_{at}(t, e_E) \;\; \forall t \in \{0, \dots, N_p\} \;\; \forall e_E \\
T(e_E) &\le M \cdot b_{before}(e_E) \;\; \forall e_E \\
\frac{\Delta t}{2} + t_{N_p} - T(e_E) &\le M \cdot b_{after}(e_E) \;\; \forall e_E \\
t_{N_p} - \frac{\Delta t}{2} + 1 &\ge b_{before}(e_E) + b_{after}(e_E) + \sum_{\tau=0}^{N_p} b_{at}(\tau, e_E) \;\; \forall e_E.
\end{aligned} \tag{3.81}$$

These constraints ensure that either some time step τ occurs within $\frac{\Delta t}{2}$ of $T(e_E)$, and that $b_{at}(\tau, e_E) = 0$ for that τ, or $T(e_E)$ is scheduled outside of the planning horizon, as required.

3.8.4 Expressing Episode Constraints in Terms of Symbolic Events

Recall that, to encode the chance constraints in the robust execution problem, we must encode the implication $z_i(c_c) = 0 \implies s(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}^{(i)}, T) = 1$. In Section 3.5.2 we showed how to do this for execution when the schedule is fixed. In this section we use the binaries defined in Section 3.8.3 to encode this implication in terms of the events e_E and e_S, which now have flexible schedules.

Convex Feasible Regions

We must ensure for a remain-in episode a with a convex feasible region $R_\forall$ that $\mathbf{x}_\tau \in R_\forall$ for all time steps τ such that $t_\tau \in [T(e_S), T(e_E)]$ if $z_i(c_c) = 0$. For an end-in episode a with a convex feasible region R_E we must ensure that, if $z_i(c_c) = 0$:

$$\begin{aligned}
&\left(\exists \tau \in \{0, \dots, N_p\} \;\; \mathbf{x}_\tau \in R_E \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]\right) \\
\vee \;\; & T(e_E) \le 0 - \frac{\Delta t}{2} \\
\vee \;\; & T(e_E) \ge t_{N_p} + \frac{\Delta t}{2}.
\end{aligned} \tag{3.82}$$

As in Section 3.5.2 we define a convex polygonal feasible region $R(a)$ associated with activity a as a conjunction of linear constraints $\mathbf{a}_l^T(a)\mathbf{x} \le b_l(a)$ for $l = 1, \dots, N_a$, where $\mathbf{a}_l(a)$ is defined as pointing outwards from the polygonal region. For all remain-in activities $a \in \mathcal{A}_\forall(c_c)$ we now impose the following constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \le M\Big(z_i(c_c) + b_{during}(\tau, e_S, e_E)\Big) \;\; \forall \tau \in \{0, \dots, N_p\} \;\; \forall l \;\; \forall a \in \mathcal{A}_\forall(c_c), \tag{3.83}$$

where M is a large positive constant, and e_S and e_E are the start and end events for activity a. If

$z_i(c_c) = 0$ and $b_{during}(\tau, e_S, e_E) = 0$ then every constraint is satisfied for particle i, otherwise (for large enough M), particle i is unconstrained. For convex regions $R(a)$, we therefore have:

$$z_i(c_c) = 0 \implies \mathbf{x}_\tau^{(i)} \in R(a) \ \ \forall t_\tau \in [T(e_S), T(e_E)] \ \ \forall a \in \mathcal{A}_\forall(c_c), \tag{3.84}$$

as required. For all end-in activities $a \in \mathcal{A}_E$ we impose the following constraint:

$$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \leq M\Big(z_i(c_c) + b_{at}(\tau, e_E)\Big) \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall l \ \ \forall a \in \mathcal{A}_E(c_c), \tag{3.85}$$

where M is a large positive constant and e_E is the end event of activity a. If $z_i(c_c) = 0$ and $b_{at}(\tau, e_E) = 0$ then every constraint is satisfied for particle i, otherwise particle i is unconstrained. For convex regions $R_E(a)$ we therefore have:

$$z_i(c_c) = 0 \implies \left(\exists \tau \ \ \mathbf{x}_\tau \in R_E(a) \ \ t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]\right) \ \ \forall a \in \mathcal{A}_\forall(c_c). \tag{3.86}$$

as required.

Non-convex Feasible Regions

As in Section 3.5.2, we consider a polygonal non-convex feasible region as the complement of a number of polygonal infeasible regions, or obstacles. For the temporally flexible case, we modify the constraints (3.45) and (3.46) to give:

$$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(i)} - \mathbf{b}_{jl}(a) + Md_{i\tau jl}(a) \geq 0 \ \ \forall a \in \mathcal{A}_\forall(c_c) \ \ \forall i \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall j \ \ \forall l \tag{3.87}$$

$$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(i)} - \mathbf{b}_{jl}(a) + Md_{i\tau jl}(a) \geq 0 \ \ \forall a \in \mathcal{A}_E(c_c) \ \ \forall i \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall j \ \ \forall l. \tag{3.88}$$

We modify the constraints (3.48) and (3.49) to give:

$$\sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq Me_{i\tau j}(a) \ \ \forall a \in \mathcal{A}_\forall(c_c) \ \ \forall i \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall j \tag{3.89}$$

$$\sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq Me_{i\tau j}(a) \ \ \forall a \in \mathcal{A}_E(c_c) \ \ \forall i \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall j. \tag{3.90}$$

Note that these constraints apply over *all* time steps τ, since the time points of events are no longer known *a priori*. Instead of (3.52) we impose, for all remain-in activities:

$$\sum_{j=1}^{M_a} e_{i\tau j}(a) \leq M \cdot \Big(z_i(c_c) + b_{during}(\tau, e_S, e_E)\Big) \ \ \forall \tau \in \{0, \ldots, N_p\} \ \ \forall a \in \mathcal{A}_\forall(c_c) \ \ \forall i, \tag{3.91}$$

which ensures that, for non-convex feasible regions $R_\forall(a)$,

$$z_i(c_c) = 0 \implies \mathbf{x}_\tau^{(i)} \in R_\forall(a) \;\; \forall\tau \in [T(e_S), T(e_E)] \;\; \forall a \in \mathcal{A}_\forall(c_c) \;\; \forall i, \tag{3.92}$$

as required. For all end-in activities, we impose the constraint:

$$\sum_{j=1}^{M_a} e_{i\tau j}(a) \le M \cdot \Big(z_i(c_c) + b_{at}(\tau, e_S, e_E)\Big) \;\; \forall\tau \in \{0, \ldots, N_p\} \;\; \forall a \in \mathcal{A}_E(c_c) \;\; \forall i, \tag{3.93}$$

which ensures that, for non-convex regions $R_E(a)$,

$$z_i(c_c) = 0 \implies \left(\exists\tau \;\; \mathbf{x}_\tau \in R_E(a) \;\; t_\tau \in \left[T(e_E) - \frac{\Delta t}{2}, T(e_E) + \frac{\Delta t}{2}\right]\right) \;\; \forall a \in \mathcal{A}_E(c_c), \tag{3.94}$$

as required.

The key difference between the temporally-flexible constraints (3.91),(3.93) and their fixed-schedule counterparts (3.50),(3.52) is that the temporally-flexible constraints include the binaries defined in Section 3.8.3 that encode events symbolically.

3.8.5 Expected State Constraints

We now extend the expected state constraints in Section 3.5.3 to deal with temporally flexibility. Again we use $\mathbf{x}_\tau^{(m)}$ to denote the sample mean. We modify constraints (3.55) through (3.62) to include the temporal relation binaries; now the constraints on the expected state are only imposed at times when $b_{at} = 0$ for end-in episodes and only at times when $b_{during} = 0$ for remain-in episodes . We impose the constraints:

$$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \le M \cdot b_{at}(\tau, e_E) \;\; \forall\tau \in \{0, \ldots, N_p\} \;\; \forall l \;\; \forall a \in \mathcal{A}_E(c_m) \tag{3.95}$$

$$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \le M \cdot b_{during}(\tau, e_S, e_E) \;\; \forall\tau \;\; \forall l \;\; \forall a \in \mathcal{A}_\forall(c_m) \tag{3.96}$$

$$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(m)} - \mathbf{b}_{jl}(a) + M d_{\tau jl}(a) \ge 0 \;\; \forall\tau \in \{0, \ldots, N_p\} \;\; \forall l \;\; \forall j \;\; \forall a \in \mathcal{A}(c_m) \tag{3.97}$$

$$\sum_{j=1}^{M_a} e_{\tau j}(a) \le M \cdot b_{at}(\tau, e_S, e_E) \;\; \forall\tau \in \{0, \ldots, N_p\} \;\; \forall a \in \mathcal{A}_E(c_m) \tag{3.98}$$

$$\sum_{j=1}^{M_a} e_{\tau j}(a) \le M \cdot b_{during}(\tau, e_S, e_E) \;\; \forall\tau \in \{0, \ldots, N_p\} \;\; \forall a \in \mathcal{A}_\forall(c_m). \tag{3.99}$$

Note that these constraints are defined over *all* time steps in the planning horizon, since the schedule of events is not known *a priori*.

3.8.6 Summary

In Sections 3.8.1 through 3.8.5 we encoded the chance-constrained Particle Control problem for temporally-flexible Qualitative State Plans as a constrained optimization. The constraints are summarized in Tables 3.5 through 3.7. Since these constraints are linear, there are integer constraints on some decision variables, and the cost function is piecewise linear in the decision variables, the overall optimization is a Mixed Integer Linear Program. This means that it can be solved to global optimality using MILP.

Description	Constraint
For at most a fraction $\delta(c_c)$ of particles does any activity in $\mathcal{A}(c_c)$ fail.	$\frac{1}{N}\sum_{i=1}^{N} z_i(c_c) \leq \delta(c_c)$
	$\forall i \;\; \forall \tau \in \{0,\dots,N_p\}$:
$z_i(c_c) = 0$ implies all remain-in activities in $\mathcal{A}(c_c)$ with convex $R_{\forall}$ succeed for particle i.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \leq M\Big(z_i(c_c) + b_{during}(\tau, e_E)\Big) \; \forall l \; \forall a \in \mathcal{A}_{\forall}(c_c)$
$z_i(c_c) = 0$ implies all end-in activities in $\mathcal{A}(c_c)$ with convex R_E succeed for particle i.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(i)} - b_l(a) \leq M\Big(z_i(c_c) + b_{at}(\tau, e_E)\Big) \; \forall l \; \forall a \in \mathcal{A}_E(c_c)$
$z_i(c_c) = 0$ implies all remain-in activities in $\mathcal{A}(c_c)$ with non-convex $R_{\forall}$ succeed for particle i.	$\sum_{j=1}^{N_{aj}} e_{i\tau j}(a) \leq M(z_i(c_c) + b_{during}(\tau, e_S, e_E)) \; \forall a \in \mathcal{A}_{\forall}(c_c)$
$z_i(c_c) = 0$ implies all end-in activities in $\mathcal{A}(c_c)$ with non-convex R_E succeed for particle i.	$\sum_{j=1}^{N_{aj}} e_{i\tau j}(a) \leq M(z_i(c_c) + b_{at}(\tau, e_S, e_E)) \; \forall a \in \mathcal{A}_E(c_c)$
	$\forall i \;\; \forall \tau \in \{0,\dots,N_p\}$
$d_{ij\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by particle i at time τ.	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(i)} - \mathbf{b}_{jl}(a) + M d_{i\tau jl}(a) \geq 0 \; \forall a \in \mathcal{A}(c_c) \; \forall j \;\; \forall l$
$e_{i\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by particle i at time τ.	$\sum_{l=1}^{N_{aj}} d_{i\tau jl}(a) - (N_{aj} - 1) \leq M e_{i\tau j}(a) \; \forall a \in \mathcal{A}(c_c) \; \forall j$
Integer constraints	$z_i(c_c) \in \{0,1\}$ $d_{ij\tau l}(a) \in \{0,1\}$ $e_{i\tau j}(a) \in \{0,1\}$

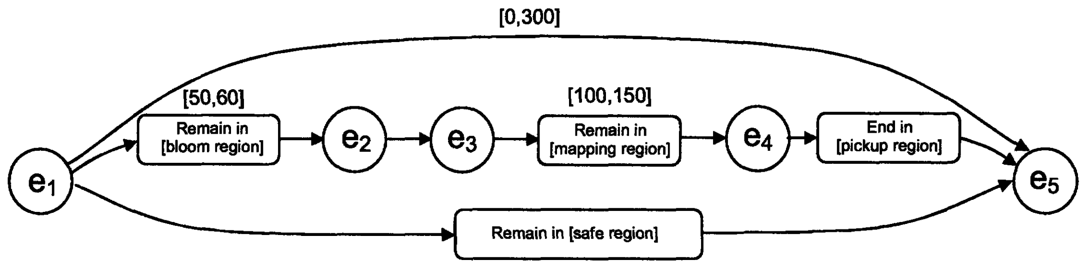
Table 3.5. Flexible-Schedule Particle Control: Constraints imposed for each chance constraint $c_c \in \mathcal{G}_c$

Description	Constraint
	$\forall \tau \in \{0,\dots,N_p\}$:
All remain-in activities in $\mathcal{A}(c_m)$ with convex $R_{\forall}$ succeed for expected state.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \leq M \cdot b_{at}(\tau, e_E) \; \forall l \; \forall a \in \mathcal{A}_{\forall}(c_c)$
All end-in activities in $\mathcal{A}(c_m)$ with convex R_E succeed for expected state.	$\mathbf{a}_l^T(a)\mathbf{x}_\tau^{(m)} - b_l(a) \leq M \cdot b_{at}(\tau, e_E) \; \forall l \; \forall a \in \mathcal{A}_E(c_m)$
All remain-in activities in $\mathcal{A}(c_m)$ with non-convex $R_{\forall}$ succeed for expected state.	$\sum_{j=1}^{N_{aj}} e_{\tau j}(a) \leq M \cdot b_{during}(\tau, e_S, e_E) \; \forall a \in \mathcal{A}_{\forall}(c_m)$
All end-in activities in $\mathcal{A}(c_m)$ with non-convex R_E succeed for expected state.	$\sum_{j=1}^{N_{aj}} e_{\tau j}(a) \leq M \cdot b_{at}(\tau, e_S, e_E) \; \forall a \in \mathcal{A}_E(c_m)$
	$\forall \tau \in \{0,\dots,N_p\}$:
$d_{j\tau l}(a) = 0$ implies constraint l in obstacle $O(a)_j$ satisfied by expected state at time τ.	$\mathbf{a}_{jl}^T(a)\mathbf{x}_\tau^{(m)} - \mathbf{b}_{jl}(a) + M d_{j\tau l}(a) \geq 0 \; \forall a \in \mathcal{A}(c_m) \; \forall j \;\; \forall l$
$e_{\tau j}(a) = 0$ implies obstacle $O(a)_j$ avoided by expected state at time τ.	$\sum_{l=1}^{N_{aj}} d_{\tau jl}(a) - (N_{aj} - 1) \leq M e_{\tau j}(a) \; \forall a \in \mathcal{A}(c_m) \; \forall j$
Integer constraints	$d_{j\tau l}(a) \in \{0,1\}$ $e_{\tau j}(a) \in \{0,1\}$

Table 3.6. Flexible-Schedule Particle Control: Constraints imposed for each expected state constraint $c_m \in \mathcal{G}_m$

Description	Constraint
$T(e_S) \leq \tau \leq T(e_E)$ implies $b_{during}(\tau, e_S, e_E) = 0$	$\forall \tau \in \{0, \ldots, N_p\}$: $t_\tau - T(e_S) \leq M \cdot b_{after}(\tau, e_S)\ \ \forall e_S$ $T(e_E) - t_\tau \leq M \cdot b_{before}(\tau, e_E)\ \ \forall e_E$ $b_{during}(\tau, e_S, e_E) \leq M(2 - b_{after} - b_{before})\ \ \forall e_S\ \ \forall e_E$
Some t_τ is $\frac{\Delta t}{2}$-close to $T(e_E)$, and $b_{at}(\tau, e_E) = 0$, or $T(e_E)$ is outside of the planning horizon.	$\forall \tau \in \{0, \ldots, N_p\}\ \ \forall e_E$: $t_\tau - \frac{\Delta t}{2} - T(e_E) \leq M \cdot b_{at}(\tau, e_E)$ $T(e_E) - t_\tau - \frac{\Delta t}{2} \leq M \cdot b_{at}(\tau, e_E)$ $\frac{\Delta t}{2} + T(e_E) \leq M \cdot b_{before}(e_E)\ \ \forall e_E$ $N_p - \frac{\Delta t}{2} - T(e_E) \leq M \cdot b_{after}(e_E)\ \ \forall e_E$ $b_{before}(e_E) + b_{after}(e_E) + \sum_{\tau=0}^{N_p} b_{at}(\tau, e_E) \leq N_p + 1$
Temporal constraints	$T(e_E) - T(e_S) \geq \Delta T^{min}_{e_S \rightarrow e_E}$ $T(e_E) - T(e_S) \leq \Delta T^{max}_{e_S \rightarrow e_E}$
Integer constraints	$b_{at}(\tau, e_E) \in \{0,1\}\quad b_{during}(\tau, e_S, e_E) \in \{0,1\}$ $b_{after}(\tau, e_S) \in \{0,1\}\quad b_{after}(e_E) \in \{0,1\}$ $b_{before}(\tau, e_S) \in \{0,1\}\quad b_{before}(e_E) \in \{0,1\}$

Table 3.7. Flexible-Schedule Particle Control: Constraints for temporal relations



$p($End in [goal region]$fails$ OR Remain in [safe region]$fails) < 0.01$

$p($Remain in [bloom region]$fails$ OR Remain in [mapping region]$fails) < 0.1$

Figure 3-12. Qualitative State Plan for AUV science mission. The mission involves collecting data from an algal bloom and mapping the sea floor before being picked up by a surface vessel.

3.9 Simulation Results: Temporally-Flexible Execution

The PC-QSP algorithm was applied to an AUV science mission scenario adapted from the one described in Section 2. We retain the same Qualitative State Plan for this mission, which we repeat in Figure 3-12. In this section we assume that waypoints in the horizontal plane have already been specified. The task for the executive is to plan depth waypoints to ensure that the chance constrained Qualitative State Plan is satisfied. We perform this decoupling for two reasons. First, applying the PC-QSP algorithm to the full 3-D planning problem is highly computationally intensive; planning only depth waypoints simplifies the problem greatly. Second, this decoupling mirrors the current procedure at the Monterey Bay Aquarium Research Institute (MBARI), which is as follows. First a human operator specifies a set of waypoints in the horizontal plane using a sophisticated visualization tool[24]. Then an automated script generates a set of depth waypoints using the best knowledge of the sea floor topology. An ad-hoc procedure is used to smooth the depth waypoints; this ensures that the waypoints can be followed (at least approximately), given the dynamic constraints of the AUV. Finally, an onboard, inner-loop feedback controller uses meaurements from the Inertial Navigation

System (INS) to track the waypoints as closely as possible.

The executive developed in this thesis could, in principle, replace the existing ad-hoc depth planning script with an on-board planning capability that specifies depth waypoints to the inner-loop controller. Using the PC-QSP algorithm, a Qualitative State Plan describing the AUV's mission, and a stochastic model describing the AUV, the executive would reason explicitly about dynamic constraints, uncertainty due to currents, uncertain localization and incomplete knowledge of the environment in order to generate depth waypoints that are optimal while guaranteeing robustness. We demonstrate this capability in simulation in this section.

We model the AUV using the linearized discrete-time closed-loop longitudinal dynamics derived by [86]. This model is of considerable size, having 8 continuous state variables ($n_x = 8$), and has been validated against experimental data. The AUV travels at a constant velocity of 3.0 knots; this is the value used for MBARI missions for reasons of fuel-minimization. The model derived by [86] has a sampling time of $0.2s$. For the purposes of planning, we transform the model to give a sampling time of $10s$. This reduces the number of time steps over which the executive must plan, assuming the length of the planning horizon in seconds is fixed. In addition, the time constant of the close-loop dynamics is large enough that a $10s$ sampling time is small enough to capture the dynamics of the AUV. The AUV must not exceed a pitch angle of $\pm 15°$ in order to ensure safety, and validity of the linearized model. We use the time of the last scheduled event as the optimality criterion to minimize. We use a planning horizon of 20 time steps, which covers a duration of 200 seconds. We model uncertainty in the AUV state as well as disturbances due to water currents.

Figure 3-13 shows a typical solution to the single-stage, limited horizon planning problem, generated using the PC-QSP algorithm with 50 particles. At the beginning of this planning stage, the AUV is in the process of executing the *Remain in* `[bloom region]` episode . By making use of the full pitch angle capabilities of the AUV, the executive minimizes the time of the end of the science mission so that the goal region is reached within the planning horizon. By reasoning explicitly about stochastic models of disturbances and localization uncertainty the executive ensures that the (approximated) probability of any episode relating to science goals failing is less than 0.1, while the (approximated) probability of any safety-critical episode failing is less than 0.01. In Figure 3-14 we show the MILP solution time for the PC-QSP algorithm as a function of the number of particles used. The solution time is significantly greater for the PC-QSP algorithm than for the fixed-schedule PC-LIN algorithm shown in Section 3.6. The increase is primarily due to the binary variables introduced to encode multiple chance constraints, and to encode the times of events symbolically.

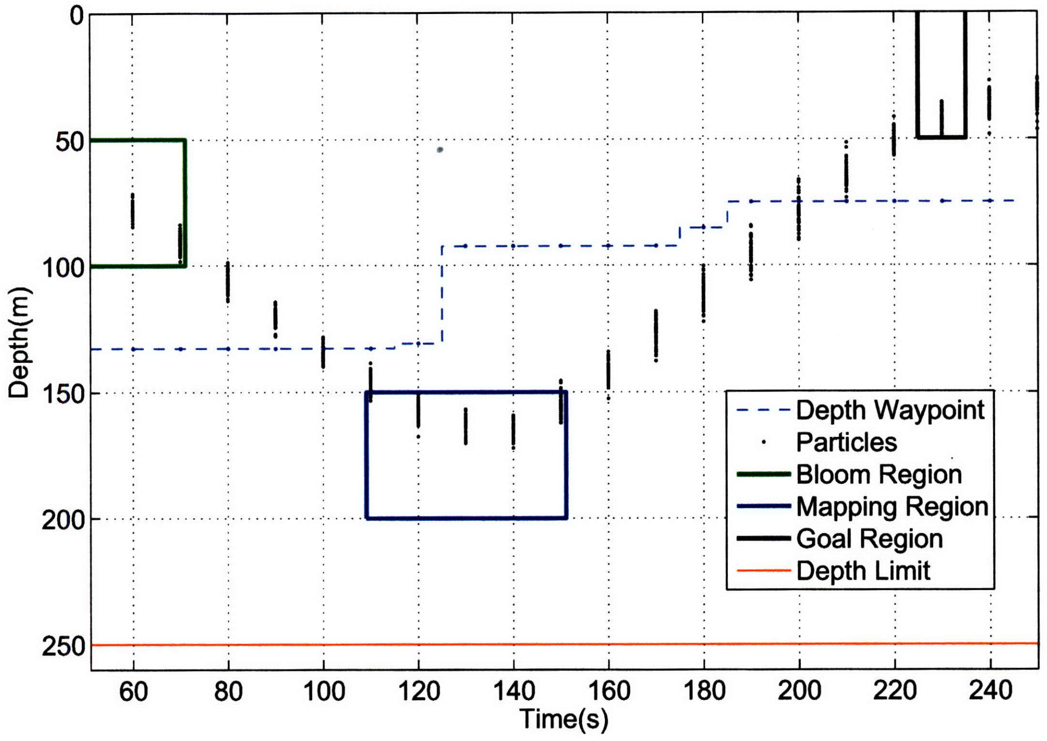


Figure 3-13. Typical solution to single-stage, limited horizon planning problem for AUV science mission, generated using PC-QSP algorithm with 50 particles. The schedule for this plan is $T(e_1) = 20$, $T(e_2) = 70$, $T(e_3) = 110$, $T(e_4) = 150$, $T(e_5) = 230$. The thin blue line denotes the depth waypoints planned by the PC-QSP algorithm. In order to minimize the time of the last event, the planner uses the full $\pm 15°$ pitch capability of the AUV. The approximated chance constraints are satisfied; for 5 out of the 50 particles either *Remain in* `[bloom region]` or *Remain in* `[mapping region]` fails, while the *Remain in* `[safe region]` and *End in* `[goal region]` episodes are satisfied for all particles.

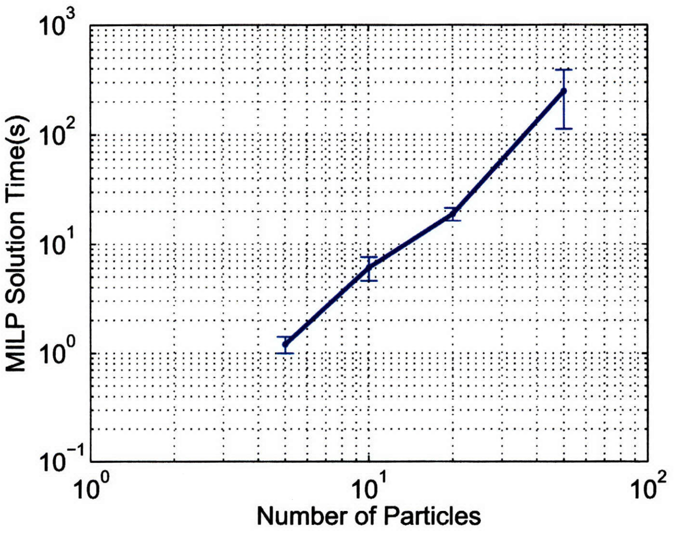


Figure 3-14. Average MILP solution time against size of particle set for PC-QSP algorithm with AUV science mission scenario. Error bars represent one standard deviation.

3.10 Weighted Particle Control for JMLS

We now extend the PC-JMLS method to deal more effectively with low probability mode transitions such as failures. Such transitions are challenging because a control strategy that is robust to failures is often very different to one that is not. The PC-JMLS algorithm will only generate controls that are robust to failures if it samples the failure transition. Since the failure transitions have very low probabilities, such transitions are rarely sampled. With a finite number of particles, it becomes highly likely that no failure transitions are sampled in the particle set. In this case PC-JMLS will design a control sequence that does not take into account failures at all, which will be far from the true optimal, robust solution.

In order to overcome this problem we take inspiration from prior work in estimation[37] that approximates one distribution by sampling from an alternative, *proposal distribution* and defining *importance weights*[37] for the samples. By proper selection of the importance weights, the weighted particle distribution converges to the original distribution, as described in Section 3.3. We apply this to the problem of sampling from the discrete mode sequences. The key idea is to sample mode transitions from a proposal distribution that makes it more likely that failure transitions are sampled. By applying importance weighting to these particles, we ensure that the resulting *weighted particle control* problem converges to the exact stochastic control problem as the number of particles tends to infinity. In Section 3.10.1 we described the PC-WEIGHTED algorithm for a general proposal distribution, before choosing a proposal distribution in Section 3.10.2. In Section 3.11 we give simulation results that show that the PC-WEIGHTED algorithm designs control sequences that perform much better than for the unweighted PC-JMLS algorithm; that is, the resulting probability of episode failure is closer to the specified chance constraints.

3.10.1 The Weighted Particle Control Algorithm

In the weighted particle control approach, discrete mode sequences are sampled from a proposal distribution that is designed to ensure that low probability events such as failures are more likely to be taken into consideration. That is, the proposal distribution raises the probability of sampling from sequences with failures compared to a fair sampling scheme. Drawing on the idea of importance weighting in particle filtering[37], the discrepancy between the actual distribution over mode sequences and the proposal distribution is represented using an analytical weighting. In doing so, we retain the convergence property that the approximate problem converges to the original stochastic problem as the number of particles tends to infinity, from the results described in Section 3.3.

Consider the case of a ground vehicle with failure-prone brakes. There is a small probability ϵ that the brakes will transition from a working mode to a faulty mode during the planning horizon, in which they have no effect. The time-optimal control strategy is to accelerate hard and brake hard, but a strategy that is robust to brake failure should accelerate gently and brake gently since, in the case of brake failure, friction can still bring the car to rest before a collision. The (unweighted) PC-JMLS algorithm will only design a control sequence that is robust to brake failure if it samples the brake failure; if it does not, the resulting control sequence will be the time-optimal, aggressive one. With N particles, the probability that PC-JMLS will sample the brake failure is approximately $N \cdot \epsilon$, from the binomial theorem. Typical values might be $\epsilon = 10^{-5}$ and $N = 1000$, which means there is a 0.01 probability of the failure sequence being sampled. The (weighted) PC-WEIGHTED algorithm, on the other hand, increases the likelihood of sampling the failure transition; with the proposal described in Section 3.10.2, the probability that PC-WEIGHTED samples the failure is approximately 0.95 for the same values of N and ϵ. This makes it much more likely that the control sequence resulting from PC-WEIGHTED is in fact robust to brake failures.

The PC-WEIGHTED algorithm is described in Table 3.8.

We now show how to calculate the weights w_i used in (3.101) and (3.102). As described in Section 3.3, for the approximated problem to converge to the original stochastic problem as the number of particles tends to infinity, weights must be assigned according to:

$$w_i = \frac{p(\mathbf{x}^{(i)}_{c,1:N_p}, \mathbf{x}^{(i)}_{d,0:N_p-1} | \mathbf{u}_{0:N_p-1})}{q(\mathbf{x}^{(i)}_{c,1:N_p}, \mathbf{x}^{(i)}_{d,0:N_p-1} | \mathbf{u}_{0:N_p-1})}. \tag{3.103}$$

Since we sample the disturbances from their true distributions, the joint proposal $q(\mathbf{x}_{c,1:N_p}, \mathbf{x}_{d,0:N_p-1:})$ can be written in terms of the proposal over mode sequences, to give:

$$w_i = \frac{p(\mathbf{x}^{(i)}_{c,1:N_p} | \mathbf{x}^{(i)}_{d,0:N_p-1}, \mathbf{u}_{0:N_p-1}) p(\mathbf{x}^{(i)}_{d,0:N_p-1})}{p(\mathbf{x}^{(i)}_{c,1:N_p} | \mathbf{x}^{(i)}_{d,0:N_p-1}, \mathbf{u}_{0:N_p-1}) q(\mathbf{x}^{(i)}_{d,0:N_p-1})} = \frac{p(\mathbf{x}^{(i)}_{d,0:N_p-1})}{q(\mathbf{x}^{(i)}_{d,0:N_p-1})}. \tag{3.104}$$

In others words, the weight is the ratio between the true probability of a mode sequence, $p(\mathbf{x}^{(i)}_{d,0:N_p-1})$, and the probability of a mode sequence according to the proposal distribution, $q(\mathbf{x}^{(i)}_{d,0:N_p-1})$. Since both $p(\mathbf{x}^{(i)}_{d,0:N_p-1})$ and $q(\mathbf{x}^{(i)}_{d,0:N_p-1})$ are straightforward to calculate, calculating the weight to assign to a sampled mode sequence is also.

We now show that for JMLS, the deterministic optimization resulting from the PC-WEIGHTED algorithm can be solved using MILP. The key insight is that, since the weights do not depend on the control inputs $\mathbf{u}_{0:N_p-1}$, incorporating weighted particles does not affect the form of the optimization problem. The weighted particle control problem can be formulated in the same manner as the unweighted PC-JMLS approach described in Section 3.7, with small modifications to the approximate chance constraints, the approximate expected state constraints, and the approximate cost function.

Each chance constraint places a maximum value on the probability of a set of episodes failing,

```
Function PCWEIGHTEDMAIN($P$,$\mathcal{M}$,$N$,$N_P$) returns ($\mathbf{u}_{0:N_p-1}$,$T$)
1)  Generate $N$ samples of the initial discrete mode $\{\mathbf{x}_{d,0}{}^{(1)},\dots,\mathbf{x}_{d,0}{}^{(N)}\}$ according to the distribution $p(\mathbf{x}_{d,0})$.
2)  For each sample $\mathbf{x}_{d,0}{}^{(i)}$, generate a sample of the initial continuous state $\{\mathbf{x}_{c,0}^{(1)},\dots,\mathbf{x}_{c,0}^{(N)}\}$ according to $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0}^{(i)})$.
3)  For each sample $\mathbf{x}_{d,0}{}^{(i)}$ generate a sample of the discrete mode sequence $\mathbf{x}_{d,0:N_p-1}^{(i)}$ according to the proposal distribution $q(\mathbf{x}_{d,0:N_p-1})$, defined in Section 3.10.2.
4)  For each sample $\mathbf{x}_{d,0:N_p-1}^{(i)}$ generate a sample of the disturbances $\{\nu_{0:N_p-1}^{(i)}\}$ from the distribution $p(\nu_{0:N_p-1}|\mathbf{x}_{d,0:N_p-1}^{(i)})$.
5)  For each sample $\mathbf{x}_{d,0:N_p-1}^{(i)}$ calculate $p(\mathbf{x}_{d,0:N_p-1}^{(i)})$ and assign a weight $w_i$ according to (3.104).
6)  Express the distribution of the future state trajectories approximately as a set of $N$ particles, where each particle $\mathbf{x}_{c,1:N_p}^{(i)}$ corresponds to the continuous state trajectory given a particular set of samples $\{\mathbf{x}_0^{(i)},\mathbf{x}_{d,0:N_p-1}^{(i)},\nu_{0:N_p-1}^{(i)}\}$.
7)  Approximate the chance constraints in terms of the weighted particles. Using the result in (3.13) the probability of any activity in the set $\mathcal{A}(c_c)$ failing is given by:

        $$p(\mathcal{A}(c_c)\text{ fails}) \approx \frac{1}{N}\sum_{i=1}^{N} w_i \cdot s\Big(\mathcal{A}(c_c),\mathbf{x}_{1:N_p}^{(i)},T\Big). \quad (3.100)$$

    The chance constraint $c_c$ is then approximated as follows:

        $$\frac{1}{N}\sum_{i=1}^{N} w_i \cdot s\Big(\mathcal{A}(c_c),\mathbf{x}_{1:N_p}^{(i)},T\Big) \le \delta. \quad (3.101)$$

8)  Approximate the expected state constraints using the weighted sample mean approximation, for example:

        $$s\Big(\mathcal{A}(c_m),\frac{1}{N}\sum_{i=1}^{N} w_i\mathbf{x}_{1:N_p}^{(i)},T\Big) = 1. \quad (3.102)$$

9)  Approximate the cost function in terms of weighted particles.
10) Solve the deterministic constrained optimization problem for control inputs $\mathbf{u}_{0:N_p-1}$.
```

Table 3.8. PC-WEIGHTED algorithm for robust execution with JMLS. The key difference between this and the unweighted PC-JMLS algorithm is that mode sequences are drawn from the proposal distribution $q(\mathbf{x}_{d,0:N_p-1})$ instead of the true distribution $p(\mathbf{x}_{d,0:N_p-1})$.

which as in (3.24) can be written:

$$p\Big(\mathcal{A}(c_c) \text{ fails}\Big) = \int_{\mathbf{x}_{c,1:N_p}} s\Big(\mathcal{A}(c_c), \mathbf{x}_{c,1:N_p}, T\Big) p(\mathbf{x}_{c,1:N_p}) d\mathbf{x}_{c,1:N_p} \leq \delta(c_c). \tag{3.105}$$

With unweighted particles, we approximate the probability of failure as the fraction of particles that fail, as in (3.26). With weighted particles, we instead use the *weighted fraction* of particles to approximate the probability of failure. Intuitively, the weights account for the discrepancy between the proposal distribution and the true distribution in calculating the failure probability. The chance constraint (3.105) is approximated as:

$$\frac{1}{N}\sum_{i=1}^{N} w_i \cdot s\Big(\mathcal{A}(c_c), \mathbf{x}_{c,1:N_p}{}^{(i)}, T\Big) \leq \delta(c_c). \tag{3.106}$$

From the results in Section 3.3.1 we have convergence of the approximated chance constraint to the original chance constraint as the N tends to infinity. In order to encode this constraint we define a set of N binary variables for each chance constraint, so that $z_i(c_c) = 0$ implies that for particle i all of the activities in $\mathcal{A}(c_c)$ succeed. In addition we constrain the *weighted* sum of the binary variables:

$$\frac{1}{N}\sum_{i=1}^{N} w_i z_i \leq \delta. \tag{3.107}$$

For the (unweighted) PC-JMLS algorithm use binary variables $z_i(c_c)$ defined in an identical manner, except that we constrain their sum, rather than the weighted sum.

A key property is that the weights w_i do not depend on the control inputs, as shown in (3.104). Hence (3.107) is a linear constraint on the binary variables z_i; we exploit this property in solving the optimization problem arising from the PC-WEIGHTED algorithm using MILP.

The expected state is approximated using the *weighted* sample mean as follows:

$$E[\mathbf{x}_{c,1:N_p}] \approx \mathbf{x}_{c,1:N_p}^{(m)} \triangleq \frac{1}{N}\sum_{i=1}^{N} w_i \mathbf{x}_{c,1:N_p}^{(i)}. \tag{3.108}$$

From the result in (3.12) we have convergence of (3.108) to the true mean as the number of particles converges to infinity. Again, since the weights do not depend on the control inputs, (3.108) is a linear function of the control inputs. This means that expected state constraints of the form:

$$s\left(\mathcal{A}(c_m), \mathbf{x}_{1:N_p}^{(m)}, T\right) = 1. \tag{3.109}$$

can be handled in an identical manner to the unweighted case in Section 3.7. The expected cost is approximated using the *weighted* sample mean as follows:

$$E[F] \approx \hat{F} = \frac{1}{N}\sum_{i=1}^{N} w_i F(\mathbf{u}_{0:N_p-1}, \mathbf{x}_{c,1:N_p}^{(i)}). \tag{3.110}$$

As the number of particles tends to infinity, we have the convergence result:

$$\hat{F} \longrightarrow E[F]. \tag{3.111}$$

Furthermore, since the weights w_i do not depend on the control inputs, the approximate value $\hat{F}$ is piecewise-linear in the control inputs assuming a piecewise-linear cost function F.

In summary, therefore, the weighted particle control problem for JMLS can be posed as a Mixed Integer Linear Program. It now remains to choose a proposal distribution $q(\mathbf{x}_{d,0:N_p-1}^{(i)})$.

3.10.2 Choosing a Proposal Distribution

The convergence of the approximate problem to the original deterministic problem applies for any choice of the proposal distribution $q(\mathbf{x}_{d,0:N_p-1})$ subject to the constraint that $q(\mathbf{x}_{d,0:N_p-1}) > 0$ wherever $p(\mathbf{x}_{d,0:N_p-1}) > 0$. However for a finite number of particles the performance of the PC-WEIGHTED algorithm is affected greatly by the choice of $q(\mathbf{x}_{d,0:N_p-1})$ as we show empirically in Section 3.11. As in particle filtering, the optimal choice of $q(\mathbf{x}_{d,0:N_p-1})$ depends on the application, and in estimation a great deal of work has focussed on developing proposal distributions for specific applications, for example[31, 38, 44, 73, 92, 128]. We now introduce a proposal distribution designed to improve the performance of the particle control approach for JMLS when dealing with low-probability transitions, such as faults.

To motivate our proposal distribution, we consider two unsuitable proposal distributions; first, a proposal equal to the true mode sequence distribution:

$$q(\mathbf{x}_{d,0:N_p-1}) = p(\mathbf{x}_{d,0:N_p-1}). \tag{3.112}$$

As noted in Section 3.10.1, in a JMLS with low-probability transitions, such as faults, there is a high probability that no fault transitions will be sampled, if this proposal is used. For example, if there is a probability of 10^{-5} of a failure transition during the planning horizon, there is approximately a 0.01 probability of a failure being sampled in 1000 particles.

The second unsuitable proposal is the pseudo-uniform distribution $q(\mathbf{x}_{d,0:N_p-1}) = U(\mathbf{x}_{d,0:N_p-1})$, where $U(\cdot)$ assigns an equal probability to each mode sequence for which $p(\mathbf{x}_{d,0:N_p-1}) > 0$. We call this pseudo-uniform because it does not assign a finite probability to sequences for which

$p(\mathbf{x}_{d,0:N_p-1}) = 0$. More precisely:

$$U(\mathbf{x}_{d,0:N_p-1}) = \begin{cases} 1/m & p(\mathbf{x}_{d,0:N_p-1}) > 0 \\ 0 & p(\mathbf{x}_{d,0:N_p-1}) = 0, \end{cases} \tag{3.113}$$

where m is the number of mode sequences for which $p(\mathbf{x}_{d,0:N_p-1}) > 0$. Using this proposal ensures that sequences involving faults are sampled with the same likelihood as mode sequences that contain no failures; even though these, in reality have a much higher probability. We assume for now that there is only one mode sequence without failures, which we refer to as the *nominal* mode sequence ${\mathbf{x}_{d,0:N_p-1}}^{nom}$. The drawback in using this proposal is that there is a significant likelihood that the nominal mode sequence is not sampled. If this occurs, the deterministic optimization will typically be infeasible, since achieving most control tasks requires nominal operation of the system components with non-zero probability.

We, therefore, use a proposal distribution $q^*(\mathbf{x}_{d,0:N_p-1})$ that increases the probability of sampling failure sequences, while ensuring that the nominal mode sequence is sampled at least once with a probability λ:

$$q^*(\mathbf{x}_{d,0:N_p-1}) = \begin{cases} P_{nom} & \mathbf{x}_{d,0:N_p-1} = {\mathbf{x}_{d,0:N_p-1}}^{nom} \\ \frac{1-P_{nom}}{m-1} & \mathbf{x}_{d,0:N_p-1} \neq {\mathbf{x}_{d,0:N_p-1}}^{nom}, \end{cases} \tag{3.114}$$

where:

$$P_{nom} = 1 - (1-\lambda)^{1/N}. \tag{3.115}$$

Recall that N is the number of particles used and m is the number of mode sequences for which $p(\mathbf{x}_{d,0:N_p-1}) > 0$. The proposal distribution $q^*(\mathbf{x}_{d,0:N_p-1})$, therefore, ensures a minimum probability of sampling the nominal mode sequence and shares the remaining probability space evenly among the remaining mode sequences. For simplicity of exposition, the proposal $q^*(\mathbf{x}_{d,0:N_p-1})$ described here assumes a single nominal mode sequence. The extension to multiple nominal mode sequences is straightforward.

3.10.3 Summary

In this section we extended the PC-JMLS algorithm to deal more effectively with low probability mode transitions such as failures. The resulting PC-WEIGHTED algorithm samples mode sequence from a proposal distribution that increases the probability of sampling a sequence containing a failure, then uses importance-weighting to ensure convergence of the approximated control problem to the original stochastic problem as the number of particle tends to infinity. In Section 3.11 we give an empirical analysis that shows that using this proposal distribution described in Section 3.10.2

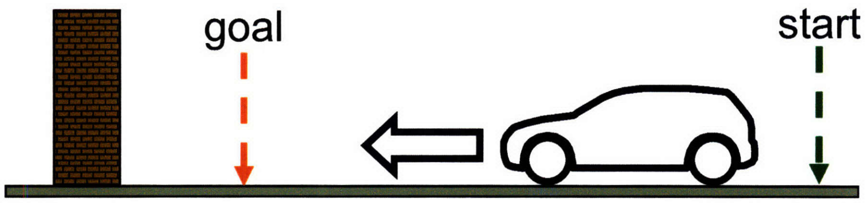


Figure 3-15. Illustration of ground vehicle brake failure scenario. The expected vehicle position must arrive at the goal in the minimum possible time, while avoiding collision with the wall.

the PC-WEIGHTED algorithm significantly outperforms the PC-JMLS algorithm for JMLS when there are low-probability transitions such as failures.

3.11 Simulation Results: Control of Jump Markov Linear Systems using Weighted and Unweighted Approaches

The PC-JMLS and PC-WEIGHTED algorithms were applied to a ground vehicle brake failure scenario.[4] In the brake failure scenario, a wheeled vehicle starts at rest at a distance of $8m$ from its goal. The vehicle can apply acceleration and braking inputs. The brakes can fail, however, in which case braking has no effect. The vehicle's expected position, conditioned on nominal brake operation, must arrive at the goal in the minimum possible time. Overall failure of the plan is defined as collision of the vehicle with a wall, which is situated $4m$ past the goal. The situation is illustrated in Figure 3-15.

The vehicle is modeled as a Jump Markov Linear System with two operational modes: in mode 1 (*ok*) the brakes are functional and in mode 2 (*faulty*) braking has no effect. The transition matrix (3.70) is given by:

$$T = \begin{bmatrix} 0.999 & 0.001 \\ 0.0 & 1.0 \end{bmatrix}. \tag{3.116}$$

If the brakes are functional, at every time step there is a probability of 0.001 that they become faulty. Once faulty, the brakes remain faulty. The brakes are initially functional, and the initial vehicle state is known exactly, however acceleration disturbances act on the vehicle during the planning horizon. A planning horizon of 20 time steps was used, with time intervals of $1s$.

The executive must generate control inputs that are robust to both continuous disturbances and to the possibility of brake failure. Intuitively, the optimal strategy depends heavily on the desired level of conservatism. A race car driver, who can tolerate a relatively high probability of failure, would accelerate as hard as possible and brake as hard as possible to achieve the minimum-time solution. A bus driver, on the other hand, must achieve a probability of failure of close to zero,

[4]The simulation results provided in this section were created in conjunction with Askar Bektassov and first published in [18].

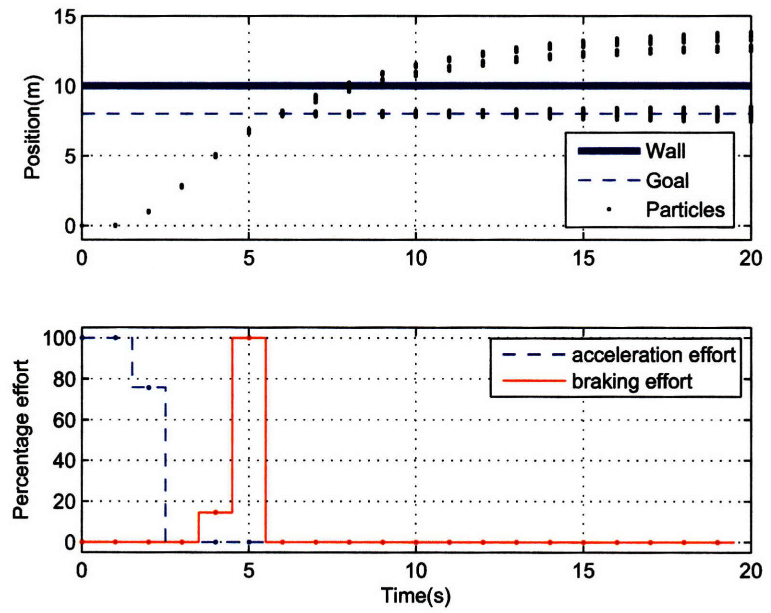


Figure 3-16. Typical solution generated by PC-WEIGHTED algorithm for a maximum probability of failure of 0.01, with 100 particles. The vehicles arrives at the goal within $6s$, but will collide with the wall if a brake failure occurs. This particular solution gives a true probability of failure of approximately 0.006.

and would therefore accelerate more slowly and brake more gradually. Both of these strategies are generated by the PC-WEIGHTED method. Figure 3-16 shows a typical solution generated by the PC-WEIGHTED algorithm for a maximum probability of failure of 0.01. Figure 3-17 shows a typical solution for a maximum probability of failure of 10^{-6}. The more conservative solution takes $9s$, while the more aggressive solution takes only $6s$.

We now demonstrate that the PC-JMLS fails to take into account low probability events, such as brake failure, but that the PC-WEIGHTED algorithm overcomes this by using weighted particles. Figure 3-18 shows a typical solution generated by the PC-JMLS algorithm, without using weighting. In this case, the algorithm did not sample any of the failure transitions and so has generated an inappropriately aggressive control policy that does not take into account the possibility of brake failure. Figure 3-19 shows a typical solution generated by PC-WEIGHTED , using weighted particles. By increasing the probability of sampling failure transitions, the weighted algorithm has taken into account brake failure, generating an appropriately conservative plan. Figure 3-20 compares the PC-JMLS algorithm against the PC-WEIGHTED algorithm in terms of the true probability of failure. In this example the desired probability of failure was 10^{-6}. The PC-WEIGHTED algorithm achieves a true probability of failure dramatically closer to the desired value than the PC-JMLS . Notice also that for larger particle sets PC-JMLS approaches the PC-WEIGHTED result, except that the variance is much greater in the case of PC-JMLS . This is because on the rare occasion that brake failure transitions are sampled, the solution is very different from the average case. This

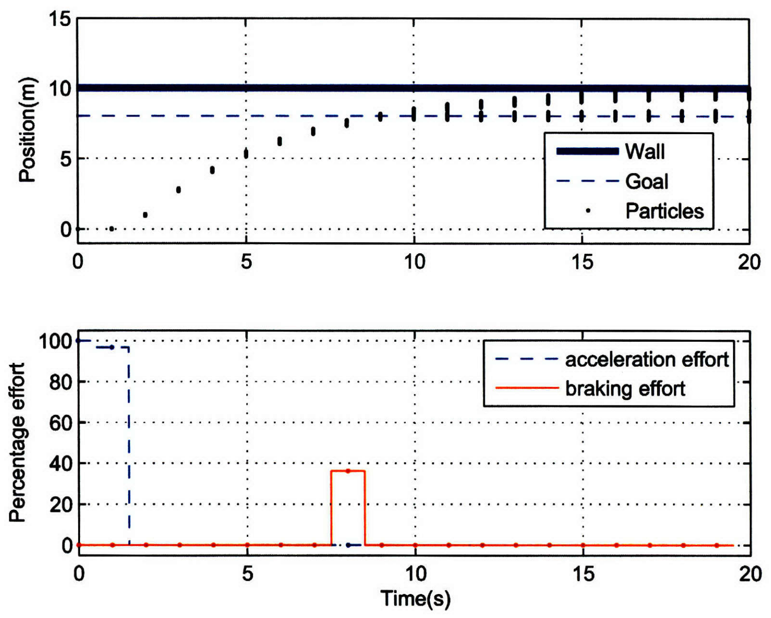


Figure 3-17. Typical solution generated by PC-WEIGHTED for a maximum probability of failure of 10^{-6}, with 100 particles. The vehicle travels more slowly and arrives within $9s$, which is later than with the more aggressive solution. In the case of brake failure, however, friction brings the vehicle to rest before collision with the wall. This solution is therefore robust to brake failure, giving a probability of failure of approximately 1.0×10^{-6}.

variance is particularly undesirable for control.

3.12 Complexity Analysis

In this section we analyze the computational complexity of the Particle Control algorithms, introduced in this chapter, namely PC-LIN , PC-JMLS , PC-QSP and PC-WEIGHTED . The complexity of PC-LIN and PC-JMLS are the same; the complexity of PC-QSP and PC-WEIGHTED are the same. In this section we therefore consider PC-JMLS and PC-QSP only. The complexity of each algorithm can be seperated into two parts; first, the generation of the approximated, deterministic optimization problem (in the form of a Mixed Integer Linear Program) and second, the solution of this optimization problem.

3.12.1 Complexity of Optimization Problem Generation

The most computationally demanding aspect of the MILP generation stage is sampling from the various noise distributions. The algorithms generate a full set of samples for each of the N particles. A set of samples corresponds to a mode sequence, an initial state, and a sequence of disturbances. Assuming that generate random sequences is linear in the length of the sequence, this means that the sampling step has complexity $\mathcal{O}(N \cdot N_p)$. Generating the MILP is therefore linear in the length

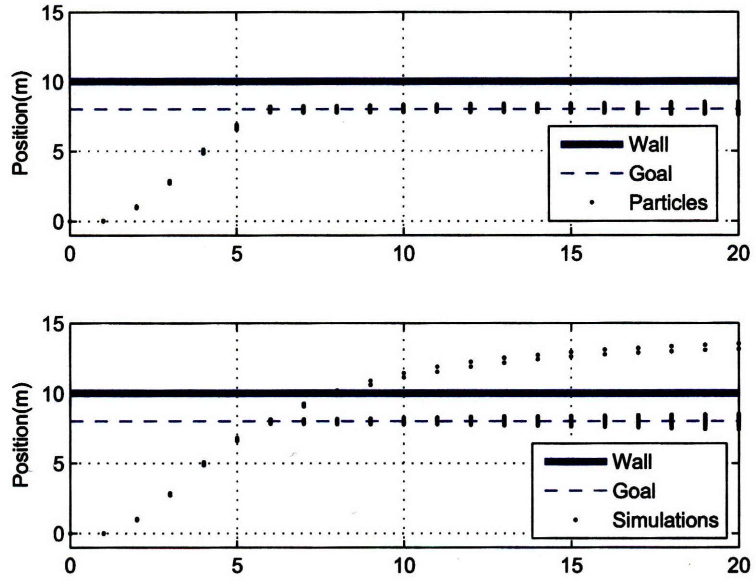


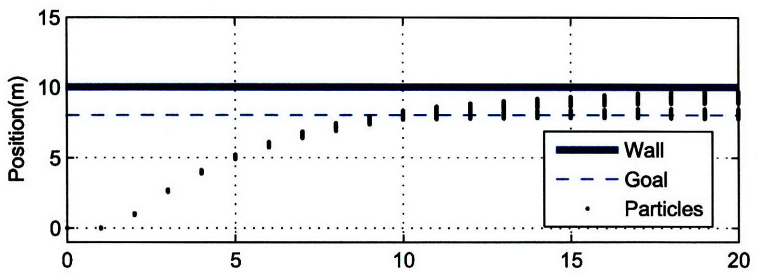
Figure 3-18. Typical solution *without* weighting for a maximum probability of failure of 10^{-6}, generated by the PC-JMLS algorithm using 100 particles. Top: Planned particle distribution. Because no particles have sampled the brake failure, the controller plans aggressively. Bottom: Monte Carlo simulations of true state trajectory. In reality, there is a probability of approximately 0.0050 that a brake failure occurs at or before $t = 5s$, causing the vehicle to collide with the wall.

of the planning horizon and the number of particles used.

3.12.2 Complexity of Optimization Problem Solution

All of the Particle Control algorithms generate Mixed Integer Linear Programs (MILPs). We must therefore analyze the complexity of a MILP in terms of the parameters of the Particle Control problem.

MILP Complexity Solution of a MILP can be viewed as a tree search, where nodes correspond to partial assignments to the integer variables in the problem. At each node a Linear Program (LP) is solved with the corresponding partial assignment to variables, and the integrality constraint relaxed on the remaining variables. The cost of the relaxed problem is a lower bound on the cost of all solutions lower down in the tree (corresponding to more integer assignments to variables); if this bound is worse than the current best feasible solution, there is no need to evaluate the LPs for the rest of the branch. This *branch and bound* approach enables a MILP to be solved efficiently in practice; cutting plane algorithms and a variety of heuristics also improve the average-case performance. In the worst case, however, a MILP requires the solution of an LP for each possible assignment to the integer variables. In the Particle Control algorithms, all integer variables are binary, hence the MILP solution requires $\mathcal{O}(2^{n_b})$ LP solutions, where n_b is the number of binary variables.



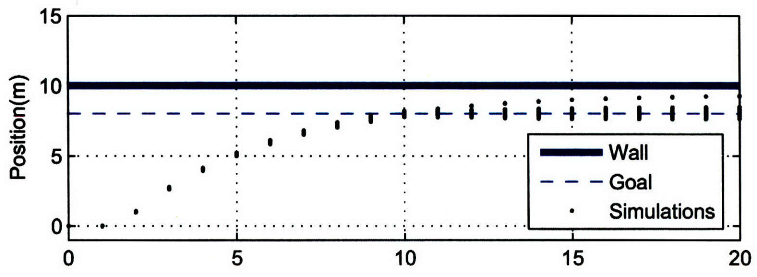


Figure 3-19. Typical solution *with* weighting for a maximum probability of failure of 10^{-6}, generated by the PC-WEIGHTED algorithm using 100 particles. Top: Planned weighted particle distribution (particle weights not shown). Many particles have sampled brake failures, hence the controller plans taking brake failures into account. Bottom: Monte Carlo simulations of true state trajectory. The vehicle collides with the wall with a probability of approximately 1.0×10^{-6}.

In Tables 3.9 and 3.10 we analyze the number of binary variables for each of the Particle Control algorithms. We use the notation that N is the number of particles; $|\mathcal{G}_c|$ is the number of chance constraints in the Qualitative State Plan; N_p is the number of time steps in the planning horizon; $|\mathcal{E}|$ is the number of events in the plan; $|\mathcal{A}_\forall|$ is the number of remain-in episodes in the plan and $|\mathcal{A}_E|$ is the number of end-in episodes in the plan. The tables show that, for PC-JMLS , the number of binaries is:

$$n_b = N \cdot |\mathcal{G}_c| + N_p(N+1)(N_{obs} + N_{edge}), \tag{3.117}$$

while for PC-QSP the total number of binaries is:

$$n_b = N \cdot |\mathcal{G}_c| + N_p(N+1)(N_{obs} + N_{edge}) + 2|\mathcal{A}_E| + N_p(|\mathcal{A}_E| + 3|\mathcal{A}_\forall|). \tag{3.118}$$

Note that handling temporal flexibility requires additional binary variables.

Description	Number of Binary Variables	Total
Particle success indicator	One for each particle for each chance constraint	$\|\mathcal{G}_c\| \times N$
Obstacle avoidance indicator	One for each particle and the expected state, for each obstacle, for each time step	$(N+1) \times N_p \times N_{obs}$
Edge avoidance indicator	One for each particle and the expected state, for each edge, for each time step	$(N+1) \times N_p \times N_{edge}$

Table 3.9. Number of binary variables for PC-JMLS algorithm.

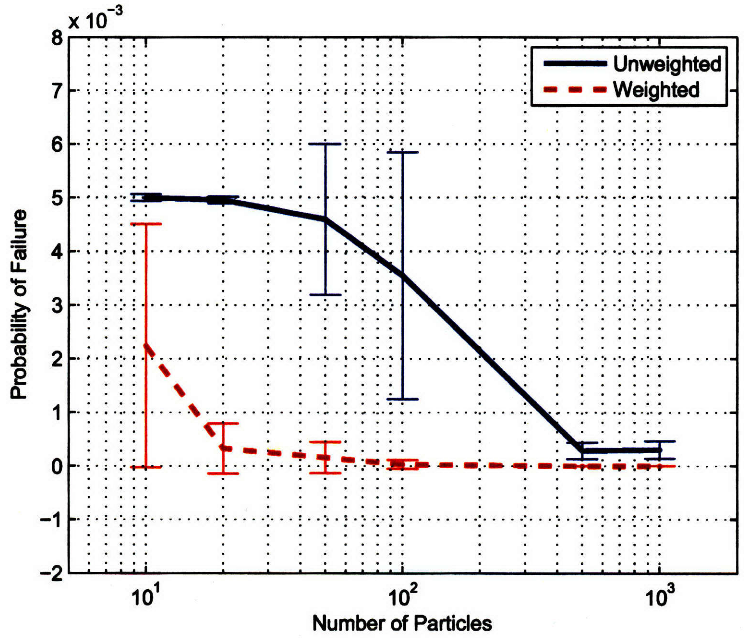


Figure 3-20. True probability of failure against number of particles for PC-WEIGHTED method and PC-JMLS method. The desired probability of failure was 10^{-6}. By using weighted particle, PC-WEIGHTED achieves a true probability of failure dramatically closer to the desired value than PC-JMLS . With a very small particle set, the effect of weighting is diminished since the probability of sampling the nominal sequence must be high in order to satisfy constraints on the probability of a feasible solution.

These results assume a linear, rather than a piecewise-linear, cost function. Piecewise-linear cost functions require additional binary variables to encode which region of state-space applies to a particular linear segment of the function.

LP Complexity Since the MILP solution requires $\mathcal{O}(2^{n_b})$ LP solutions, we must analyze the complexity of solving a Linear Program. Karmarkar's algorithm provided the polynomial upper bound $\mathcal{O}(N_v^{3.5}L)$ on the complexity of a Linear Program, where N_v is the total number of decision variables in the program, and L is the total number of bits of all numbers given to the algorithm (in the constraints and cost function)[89]. Assuming a maximum size for all numbers, this means that the worst-case complexity of an LP is $\mathcal{O}\Big(N_v^{3.5}ln\big(N_v(N_c+1)\big)\Big)$, where N_c is the number of constraints in the program.

In Tables 3.11 through 3.14 we list the continuous variables and constraints in each LP solved as part of the MILP formulation in the Particle Control algorithms. Note that in solving the MILP, the integer variables also become continuous variables. For PC-JMLS , the maximum number of continuous variables in each LP is:

$$N_v = N \cdot |\mathcal{G}_c| + N_p(N+1)(N_{obs}+N_{edge}) + N_p\Big((N+1)n_x + n_u\Big), \tag{3.119}$$

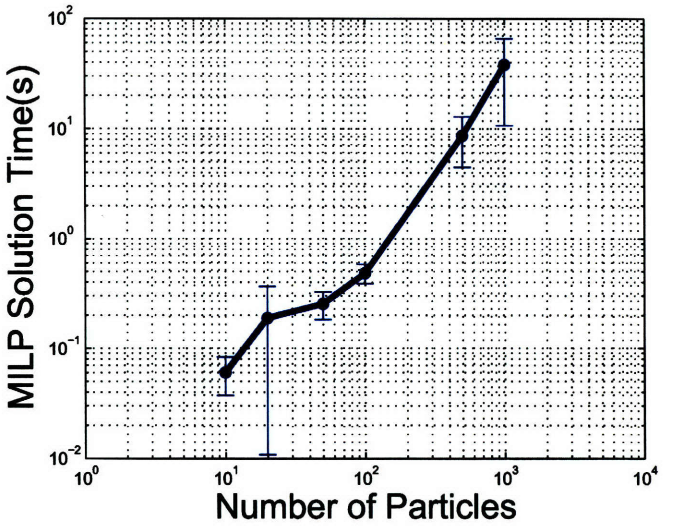


Figure 3-21. MILP solution time for PC-WEIGHTED approach with ground vehicle braking scenario. The specified maximum probability of failure was 0.01.

while for PC-QSP , the maximum number of continuous variables in each LP is:

$$N_v = N\cdot|\mathcal{G}_c|+N_p(N+1)(N_{obs}+N_{edge})+2|\mathcal{A}_E|+N_p(|\mathcal{A}_E|+3|\mathcal{A}_\forall|)+N_p\Big((N+1)n_x+n_u\Big)+|\mathcal{E}|. \quad (3.120)$$

As in Def. 2, n_x is the number of continuous state variables in the Probabilistic Hybrid Automaton, and n_u is the number of control inputs.

Complexity of Particle Control We therefore conclude the following regarding the worst-case complexity of the Particle Control algorithms:

1. The PC-LIN , PC-JMLS , PC-QSP and PC-WEIGHTED algorithms have a worst-case running time exponential in almost all problem parameters; this includes the number of particles; the length of the planning horizon; the number of chance constraints, mean constraints, and episodes ; the size of the PHA state vector and control input vector; the number of obstacles in any non-convex feasible region; the number of edges in the obstacles.

2. The PC-QSP algorithm has a greater complexity than the PC-JMLS algorithm, due to the additional binaries used for scheduling events. As well being worst-case exponential in the parameters listed above, PC-QSP is also worst-case exponential in the number of events in the plan.

Despite this, we showed empirically in Sections 3.6, 3.9 and 3.11 that, if the problem size is restricted, the resulting MILP can be solved in seconds. This is in contrast to existing MCMC methods, which take many hours to solve far smaller problems. The practical usefulness of MILP formulations

Description	Number of Binary Variables	Total
Particle success indicator	One for each particle for each chance constraint	$\|\mathcal{G}_c\| \times N$
Obstacle avoidance indicator	One for each particle and the expected state, for each obstacle, for each time step	$(N+1) \times N_p \times N_{obs}$
Edge avoidance indicator	One for each particle and the expected state, for each edge, for each time step	$(N+1) \times N_p \times N_{edge}$
'During episode' indicator	One for each remain-in episode, for each time step	$N_p \times \|\mathcal{A}_\forall\|$
'At event' indicator	One for each end-in episode, for each time step	$N_p \times \|\mathcal{A}_E\|$
'Before event' indicator	One for each remain-in episode , for each time step	$N_p \times \|\mathcal{A}_\forall\|$
'After event' indicator	One for each end-in episode , for each time step	$N_p \times \|\mathcal{A}_\forall\|$
'After end of horizon' indicator	One for each end-in episode	$\|\mathcal{A}_E\|$
'Before end of horizon' indicator	One for each end-in episode	$\|\mathcal{A}_E\|$

Table 3.10. Number of binary variables for PC-QSP algorithm.

Description	Number of Variables
Control input sequence	$N_p \times n_u$
State sequence for each particle	$N \times N_p \times n_x$
Expected state sequence	$N_p \times n_x$

Table 3.11. Continuous variables for each LP with PC-JMLS .

Description	Max. Number of Constraints
Dynamics constraints	$N \times N_p \times n_x$
Expected state constraint	$N_p \times n_x$
Remain-in state constraints (convex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times \|\mathcal{A}_\forall\| \times N \times N_p$
End-in state constraints (convex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times \|\mathcal{A}_E\| \times N$
Remain-in state constraints (nonconvex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times N + \|\mathcal{A}_\forall\| \times N \times N_p \times N_{obs} \times (N_{edge}+1)$
End-in state constraints (nonconvex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times N + \|\mathcal{A}_E\| \times N \times N_p \times N_{obs} \times (N_{edge}+1)$

Table 3.12. Constraints in each LP with PC-JMLS .

Description	Number of Variables
Control input sequence	$N_p \times n_u$
State sequence for each particle	$N \times N_p \times n_x$
Expected state sequence	$N_p \times n_x$
Event time	$\|\mathcal{E}\|$

Table 3.13. Continuous variables for each LP with PC-QSP .

Description	Max. Number of Constraints
Dynamics constraints	$N \times N_p \times n_x$
Expected state constraint	$N_p \times n_x$
Remain-in state constraints (convex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times \|\mathcal{A}_\forall\| \times N \times N_p$
End-in state constraints (convex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times \|\mathcal{A}_E\| \times N \times N_p$
Remain-in state constraints (nonconvex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times N + \|\mathcal{A}_\forall\| \times N \times N_p \times N_{obs} \times (N_{edge}+1)$
End-in state constraints (nonconvex)	$(\|\mathcal{G}_c\| + \|\mathcal{G}_m\|) \times N + \|\mathcal{A}_E\| \times N \times N_p \times N_{obs} \times (N_{edge}+1)$
Temporal constraints	$\|\mathcal{C}\|$
Symbolic encodings for remain-in episodes	$3N_p \times \|\mathcal{A}_\forall\|$
Symbolic encodings for end-in episodes	$\|\mathcal{A}_E\| \times (2N_p + 3)$

Table 3.14. Constraints in each LP with PC-QSP .

despite the worst-case exponential complexity has been noted in many different cases, for example [113, 112, 118]; we exploit this in this thesis to solve the robust execution problem.

3.13 Conclusion

In this chapter we presented a novel approach, called Particle Control , to optimal, robust execution of state plans that takes into account probabilistic uncertainty due to 1) continuous disturbances, 2) uncertain state estimation, 3) modeling error, 4) stochastic mode transitions and 5) uncertainty in obstacle location; execution ensures that the probability of failure of sets of activities is at most a user-specified threshold. The Particle Control method approximates the original stochastic problem as a deterministic one using a number of particles that sample all stochastic variables. By controlling the trajectories of these particles in a manner that is optimal with regard to the approximated problem, the method generates approximately optimal solutions to the original stochastic problem. Furthermore, the approximation error tends to zero as the number of particles tends to infinity. By using a sample-based approach, the new Particle Control method is able to handle arbitrary probability distributions. We demonstrate the method in simulation and show that the true probability of failure tends to the desired probability of failure as the number of particles used increases. Furthermore the new approach is orders of magnitude faster than existing Monte Carlo Markov Chain approaches to chance-constrained control.

Chapter 4

Active Hybrid Estimation

In this chapter we consider the problem of estimating the hybrid discrete-continuous state of a Probabilistic Hybrid Automaton given the plant model, past observations and control inputs. Previous work has shown that exact state estimation in hybrid discrete-continuous systems is, in general, intractable[81]. A number of tractable algorithms have been proposed that approximate the true belief state[25, 61, 39]. One common approach is to store a finite subset of the possible discrete mode sequences[80, 10]. However, by approximating the true belief state it is possible to lose track of the true mode sequence, at which point the estimator diverges. Previous work has highlighted this problem and suggested a number of solutions[19, 80, 79, 45, 65].

These approaches are 'passive' in the sense that they attempt to do the best possible with the observations that are made available during nominal operation. In many cases, however, it is possible to obtain a great deal more information about the state of a hybrid system by issuing appropriate control inputs. For example, in the case of detecting a fault in an actuator, a change in the actuator dynamics will not be apparent in the observations unless some effort is requested from that actuator.

In this chapter we introduce an *active* hybrid estimation approach that uses the additional latitude specified in the Qualitative State Plan to improve state estimation. The algorithm generates control inputs to minimize the probability of the estimator losing the true mode sequence, while ensuring that the chance-constrained Qualitative State Plan is satisfied.

In order to develop the active hybrid estimation capability we provide four main technical contributions. First, we develop a method by which a finite, constrained sequence of control inputs is designed in order to discriminate optimally between a finite number of linear dynamic system models. We call this the AE-MM algorithm. Within a Bayesian framework, optimal discrimination corresponds to minimization of the probability of model selection error. While the probability of model selection error cannot be calculated in closed form[40], we derive a novel upper bound on this value that applies to an arbitrary number of models. Then we minimize this upper bound subject to hard constraints on the control inputs and expected state. We show that the resulting optimization problem can be solved using Sequential Quadratic Programming[99].

Our second contribution is to extend this multiple-model discrimination method to an active estimation capability for Jump Markov Linear Systems, which we refer to as the AE-JMLS algorithm. The key insight is that for a given discrete mode sequence, the system dynamics, although time-varying, are fully known. By extending the error bound derived for discrimination between different models to time-varying systems, we create a closed-form upper bound on the probability of a hybrid estimator losing track of the true mode sequence. We then use a constrained finite horizon control design approach to minimize this upper bound subject to hard constraints on the control inputs and the expected system state.

The problem with this approach is that the closed-form bound considers each possible mode sequence over a finite horizon. The number of mode sequences is exponential in the number of discrete modes and in the length of the design horizon. In practice this means that an active hybrid estimation approach can only consider a subset of the possible mode sequences. Our third contribution is therefore to introduce an efficient pruning method. This method ensures that the control design only takes into account mode sequences that are *a priori* likely to contribute to the probability of losing the true mode sequence. The result is a tractable optimization problem that can be solved using Sequential Quadratic Programming[99], for example.

Our final contribution is to extend active hybrid estimation to chance-constrained Qualitative State Plan execution, to give the AE-QSP algorithm. The nonlinearity of the cost function for active estimation, combined with the nonconvex, disjunctive nature of the constraints imposed by chance-constrained QSP execution make this a challenging problem. In order to solve this problem, we first generate a feasible solution to the problem using the PC-QSP algorithm described in Chapter 3. This solution satisfies the chance-constrained Qualitative State Plan, while minimizing resource usage (for example fuel). We then perform a convex tightening of the constraints around this feasible solution. Finally, we search within the convex constraints for the solution that is optimal with regard to active estimation; this solution also ensures chance-constrained execution of the QSP.

There are a number of alternative approaches to solving the active estimation problem. Instead of planning a finite control sequence, we could find the optimal input at each time step. This has several disadvantages, however, which we describe in detail in Section 4.9. It would also be possible to formulate the problem so that we *constrain* the probability of model selection error, and minimize another criterion such as fuel use. This is an interesting avenue for future research, which we consider in Section 6. Finally, instead of our *open-loop* formulation we could consider the case where future control actions are allowed to depend explicitly on future observations. This, however, greatly increases the complexity of the problem; hence we defer this to future work.

The contributions in this chapter apply to a subset of Probabilistic Hybrid Automata (Def. 2), in particular Jump Markov Linear Systems (JMLS) with Gaussian uncertainty. JMLS are Probabilistic Hybrid Automata for which the discrete transitions do not depend on the continuous state, and the continuous dynamics in each mode are linear. In Section 4.1 we review related work. In Section 4.2 we describe the AE-MM approach to optimal discrimination between a finite number of linear dynamic models, and give relevant simulation results in Section 4.3. In Section 4.4 we extend this to active estimation for Jump Markov Linear Systems and describe the AE-JMLS algorithm. We demonstrate the AE-JMLS algorithm in simulation in Section 4.5. Finally, in Section 4.6 we describe our AE-QSP approach for active hybrid estimation with Qualitative State Plans, and provide empirical validation in Section 4.7.

4.1 Related Work

The idea of taking actions to reduce uncertainty has been applied in many domains. In the case of parameter estimation for dynamic systems, existing methods design control inputs to improve the convergence of parameter estimates; see [50] for a summary of this research line. The case of control design for state estimation was investigated by [64], building on ideas of *active perception* from the field of computer vision; see [120] for a summary. These research lines typically consider controls that can take on continuous values, whereas the Artificial Intelligence community has primarily focussed on discrete-valued control actions. The work of [87, 126] aimed to formalize the problem of generating tests for the purpose of reasoning about different hypotheses. In the case of *active diagnosis*, [30] proposed an approach for choosing the discrete action that minimizes the expected entropy of the belief state across all possible diagnoses after one observation. In the case of *active learning*, [129, 93] use control actions (also called *interventions*) to learn the causal structure of a Bayesian Network.

In this thesis we are concerned with state estimation in hybrid discrete-continuous systems. State estimation in hybrid discrete-continuous systems can be viewed as an extension of Multiple Model (MM) fault detection (also known as Multiple Model selection). Previous work has shown that Multiple Model fault detection is an effective approach for detecting faults in dynamic systems, for example [106, 56, 135]. This approach determines the most likely model from a finite set, given observations. The idea of designing control inputs to improve the performance of the selection algorithm has its origins in optimal experiment design[41, 7]. Many authors have proposed algorithms to achieve this [41, 7, 98, 97, 138, 71, 103, 122] for the case of Multiple Model fault detection. We now elaborate on some of these approaches.

Typical approaches for discrimination between linear dynamic systems, for example [98, 138, 71], design *auxiliary* signals that are added to the nominal control signal for the purpose of model discrimination. The auxiliary signal has low power so that its effect on the system state is limited.

This, however, also restricts the discrimination power of the signal. Our new approach, by contrast, designs control inputs with respect to hard constraints. These constraints can be used to ensure that a certain task, defined in terms of the system state, is fulfilled, or that hard constraints such as actuator saturation are not violated. By optimizing subject to hard constraints, the method can generate signals that are far more effective in discrimination than a limited power auxiliary signal. In addition, control inputs are chosen from a continuous set. This is in contrast with approaches such as [122] that choose control inputs from a finite set.

Previous approaches to the problem of control design for model discrimination have suggested a number of different criteria for optimization, for example expected information gain, or the distance between the observation distributions conditioned on different models. [138] designs control inputs to maximize the mean information increment and shows that this is equivalent to maximizing the Kullback-Liebler divergence between the observation distributions corresponding to two models. In a fault detection setting, [71] minimizes the time between occurrence and detection of the fault, while ensuring that the false alarm rate remains constant. [23, 98, 97] considered the case where it is possible to design control inputs that *guarantee* that the correct model is selected. These methods assume that either there are no noise processes (in other words, they do not model disturbances and observation noise), or that the noise processes are set-bounded. This enables [23], for example, to design control inputs that ensure the sets of possible observations for each model do not overlap; this guarantees that the correct model can be selected.

Maximizing the Kullback-Liebler divergence between the distribution of predicted observations is a useful heuristic for model discrimination; intuitively, if the distributions corresponding to different models are far apart, then it is 'easy' in some sense for model selection to decide between them. However this heuristic does not have an interpretation relating to the performance of the model selection algorithm. The same applies to expected information gain (also known as entropy reduction), which has a rigorous mathematical interpretation, but does not explicitly characterize the performance of the selection algorithm. Since we are concerned with Bayesian Multiple-Model selection, we argue that the probability of selecting the wrong model is the ideal criterion to minimize. Assuming an equal cost for selecting each wrong model, this criterion also measures the expected cost of the decision; in this case, minimizing the probability of model selection error leads to the decision-theoretic optimal strategy. The probability of model selection error can be evaluated in closed form if there is a single (one-dimensional) measurement and a single control variable[42]. In this thesis we are concerned with systems that have multiple outputs and multiple inputs, and must therefore consider the more general case of multiple observations, for which the probability of model selection error cannot be calculated in closed form. To overcome this, we derive a novel upper bound on the probability of model selection error. This upper bound applies to an arbitrary number of models, in contrast to existing bounds that apply to selection between two models[40].

We show that the new upper bound can be evaluated in closed form; this enables us to design control inputs that minimize a criterion that explicitly characterizes the predicted performance of the model selection algorithm. While we do not prove analytically that minimizing the upper bound decreases the true probability of error, we do show empirically that this is the case. Furthermore, the new upper bound does *not* reduce to previously used heuristics, such as the Kullback-Liebler divergence, in the special cases of two hypotheses.

Estimation in hybrid discrete-continuous systems can be considered a generalization of Multiple Model detection where the discrete mode is considered part of the system state, and has stochastic dynamics. While much recent work has gone into developing tractable approximate algorithms for hybrid state estimation[80, 10, 19, 80, 79, 45, 65], our approach is (to the author's knowledge) the first to design control inputs to improve the performance of such estimation algorithms.

4.2 Finite Horizon Control Design for Optimal Model Discrimination

4.2.1 Problem Statement

In this section we consider the linear discrete-time dynamic system described by:

$$\begin{aligned} \mathbf{x}_{c,\tau+1} &= A\mathbf{x}_{c,\tau} + B\mathbf{u}_\tau + \omega_\tau \\ \mathbf{y}_{\tau+1} &= C\mathbf{x}_{c,\tau+1} + D\mathbf{u}_\tau + \nu_\tau. \end{aligned} \tag{4.1}$$

The variables ω and ν are process and observation noise, respectively, which we restrict to be zero-mean, Gaussian white noise with covariance Q and R, respectively. The initial distribution $p(\mathbf{x}_{c,0})$ is a Gaussian with mean μ and covariance V. We use the same variable notation as Def. 1, which we repeat here:

Definition 27. *We use subscript notation to denote the value of a variable at a given time, e.g. $\mathbf{x}_{c,\tau}$ is the value of $\mathbf{x}_c$ at time step τ, where τ is an integer. We denote the time at which time step τ occurs as t_τ. We assume a fixed time interval Δt, hence $t_\tau = t_0 + \tau \cdot \Delta t$. We use $\mathbf{x}_{c,0:N}$ to denote the finite sequence $\langle \mathbf{x}_{c,0}, \ldots, \mathbf{x}_{c,N} \rangle$. The infinite sequence $\langle \mathbf{x}_{c,0}, \mathbf{x}_{c,1} \ldots \rangle$ is denoted $\mathbf{x}_{c,0:\infty}$. The realization of a random variable $\mathbf{x}$ is denoted using the upper case X. A sample drawn from a random variable $\mathbf{x}$ is denoted $\mathbf{x}^{(i)}$ where i is the sample number.*

In the Multiple Model selection problem, the parameters $\{A, B, C, D, Q, R\}$ are unknown, but we assume that they take values from a finite set of 'models' or 'hypotheses' $\mathcal{H}$. Under model $H_i \in \mathcal{H}$, the system parameters are $\{A(i),B(i),C(i),D(i),Q(i),R(i)\}$. We assume that for each i the parameters $\{A(i),B(i),B(i),D(i),Q(i),R(i)\}$ are fully known and that the true model persists indefinitely. In other words, in this section we do not allow switching between the different models; switching dynamics are considered in Section 4.4.

Multiple Model selection uses Bayesian hypothesis selection to determine the most likely system parameters given a prior distribution over models, a sequence of observations $\mathbf{y}_{\tau:\tau+h}$ and a sequence of control inputs $\mathbf{u}_{\tau:\tau+h-1}$. We review Multiple Model selection in Section 4.2.2. In this section we aim to design control inputs to aid model selection. The problem is stated formally as follows:

Definition 28. *At time τ, given a set of models $\mathcal{H}$ and a prior probability for each model $H_i \in \mathcal{H}$, the* Optimal Model Discrimination Problem *consists of designing a finite sequence of control inputs $\mathbf{u}_{\tau:\tau+h-1}$ that minimizes the probability of selection error when using Bayesian model selection.*

In Section 4.2.2 we define Bayesian model selection and the probability of model selection error. In Sections 4.2.3 and 4.2.4 we show that, while the probability of model selection error cannot be evaluated in closed form, it can be upper bounded. In our approach to solving the Optimal Model Discrimination Problem (Def. 28) we therefore minimize an upper bound on the model selection error. The approach is described in detail in Section 4.2.5.

4.2.2 Hypothesis Selection and Bayes Risk

Bayesian hypothesis selection between an arbitrary number of models given a general vector of observations $\mathbb{Y}$ can be expressed as follows:

Select H_i where $i = \arg\max_j p(H_j|\mathbb{Y}, \mathbb{U})$.

Using Bayes' rule, this selection is equivalently given by:

Select H_i where $i = \arg\max_j p(\mathbb{Y}|H_j, \mathbb{U})p(H_j)$.

The term $p(H_j)$ represents prior probabilities for model j. The priors for each model can be calculated in a number of different ways: there may be explicit knowledge about how *a priori* likely the different models are, or the prior can represent the current belief state calculated by an on-line multiple model estimation scheme. Methods for calculating this belief state have been investigated by [56, 135, 26, 130] among others.

As shown in Figure 4-1, Bayesian selection yields a number of decision regions $\Re_i$, in which H_i is the most probable hypothesis:

$$\Re_i = \left\{\mathbb{Y}|p(H_i, \mathbb{Y}|\mathbb{U}) > p(H_l, \mathbb{Y}|\mathbb{U}) \quad \forall l \neq i\right\}. \tag{4.2}$$

Model i is selected if the observation $\mathbb{Y}$ falls in region $\Re_i$. This Bayesian selection rule minimizes the likelihood of selecting an incorrect model given the available information. As shown in Figure 4-1,

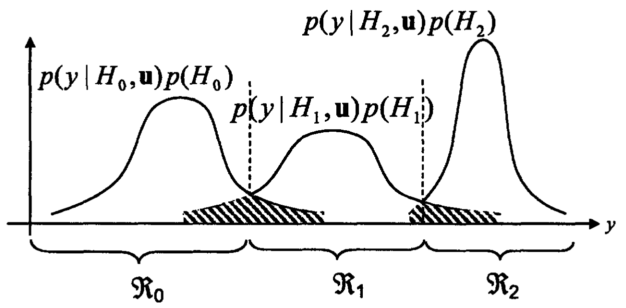

$p(y \mid H_2,\mathbf{u})p(H_2)$
$p(y \mid H_0,\mathbf{u})p(H_0)$
$p(y \mid H_1,\mathbf{u})p(H_1)$
y
$\Re_0$
$\Re_1$
$\Re_2$

Figure 4-1. Selection between multiple models given an observation y and a prior[40]. In general Bayesian selection between multiple hypotheses yields a number of *decision regions* in the space of possible observations; in each decision region a particular hypothesis is most likely. If the observation y falls into set $\Re_i$ then the classifier selects H_i. Even with Bayes optimal selection there is a finite probability of error given by the *Bayes Risk*, denoted by the shaded region.

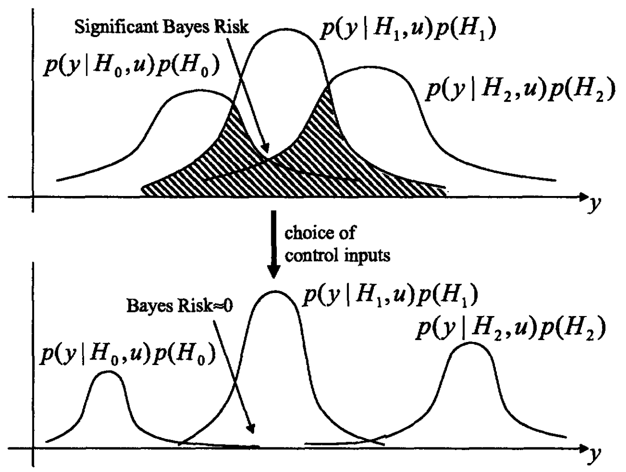


Figure 4-2. Graph showing $p(y|H_0,u)$, $p(y|H_1,u)$ and $p(y|H_2,u)$ for two different choices of u. In the upper figure, the predicted distributions overlap significantly, leading to a large Bayes risk. In the lower figure, a different selection of u has separated the distributions, meaning that when the observation y is made, the correct model can be selected with high confidence. The Bayes risk is very low, meaning that the probability of error is very low.

the Bayesian optimal classifier has a finite probability of selecting the incorrect model, known as the *Bayes Risk*. The Bayes risk is given by:

$$\begin{aligned} P(error) &= \sum_i \sum_{j \neq i} P(\mathbb{Y} \in \Re_j, H_i|\mathbb{U}) = \sum_i \sum_{j \neq i} P(\mathbb{Y} \in \Re_j|H_i,\mathbb{U})P(H_i) \\ &= \sum_i \sum_{j \neq i} \int_{\Re_j} p(\mathbb{Y}|H_i,\mathbb{U})P(H_i)d\mathbb{Y}. \end{aligned} \tag{4.3}$$

Since the Bayes Risk is the probability of error when using the optimal classifier, we would like to optimize our control inputs to the system to minimize this measure. The effect of input choice on the Bayes Risk is illustrated in Figure 4-2.

4.2.3 Bounding the Bayes Risk for Two Models

The Bayes Risk is unsuitable as an optimization criterion, since the finite integral in (4.3) cannot, in general, be evaluated in closed form. It is, however, possible to *bound* the Bayes Risk in closed form. For the special case of two models H_0 and H_1, the *Battacharyya Bound* [40] applies, and we show in this section that this leads to a quadratic cost function for model discrimination. In Section 4.2.4 we develop a novel bound that applies to more than two models.

The Battacharyya Bound is given by the integral:

$$P(error) \leq P(H_0)^{\frac{1}{2}} P(H_1)^{\frac{1}{2}} \int \sqrt{p(\mathbb{Y}|H_0)p(\mathbb{Y}|H_1)} d\mathbb{Y}. \tag{4.4}$$

If the distributions are Gaussian such that $p(\mathbb{Y}|H_0)$ has mean $\mu(0)$ and variance $\Sigma(0)$, and $p(\mathbb{Y}|H_1)$ has mean $\mu(1)$ and variance $\Sigma(1)$, the above value can be calculated without the need for integration:

$$P(error) \leq P(H_0)^{\frac{1}{2}} P(H_1)^{\frac{1}{2}} \exp\{-k\},$$

where:

$$k = \frac{1}{4}(\mu(1) - \mu(0))' \left[\Sigma(0) + \Sigma(1)\right]^{-1} (\mu(1) - \mu(0)) + \frac{1}{2} \ln \frac{\left|\frac{\Sigma(0)+\Sigma(1)}{2}\right|}{\sqrt{|\Sigma(0)||\Sigma(1)|}}. \tag{4.5}$$

Since the logarithm is a monotonically increasing function, the value of x that optimizes $f(x)$ is also the value that optimizes $\ln[f(x)]$. We therefore take the logarithm of the Battacharyya bound for Gaussian distributions to yield the following cost function:

$$\begin{aligned} J =& \frac{1}{2} \ln[P(H_0)P(H_1)] - \frac{1}{2} \ln \frac{\left|\frac{\Sigma(0)+\Sigma(1)}{2}\right|}{\sqrt{|\Sigma(0)||\Sigma(1)|}} \\ & - \frac{1}{4}(\mu(1) - \mu(0))' \left[\Sigma(0) + \Sigma(1)\right]^{-1} (\mu(1) - \mu(0)). \end{aligned} \tag{4.6}$$

Minimizing this cost function will therefore minimize an upper bound on the probability of error when using Bayesian selection to decide between two models based on a vector of observations $\mathbb{Y}$. Notice that the cost function is quadratic in the observation means $\mu(0)$ and $\mu(1)$. In Section 4.2.5 we show that this means Quadratic Programming can be used in the special case of discrimination between two models. This use of the Battacharyya Bound in model discrimination is novel, and furthermore the criterion is different from existing criteria for optimal model discrimination that have been proposed by other authors. The most similar criterion is the KL divergence, which was used by [138]. The Kullback-Liebler (KL) divergence from the distribution $\mathcal{N}(\mu(0), \Sigma(0))$ to the distribution $\mathcal{N}(\mu(1), \Sigma(1))$ is given by:

$$\frac{1}{2}\left(\ln\left(\frac{|\Sigma(1)|}{|\Sigma(0)|}\right) + tr(\Sigma(1)^{-1}\Sigma(0)) + (\mu(1) - \mu(0))'\Sigma(1)^{-1}(\mu(1) - \mu(0)) - N\right), \tag{4.7}$$

where N is the dimensionality of the distribution. The 'symmetrized' KL divergence between two distributions is given by:

$$tr(\Sigma(1)^{-1}\Sigma(0)) + tr(\Sigma(0)^{-1}\Sigma(1)) + (\mu(1) - \mu(0))'[\Sigma(1)^{-1} + \Sigma(0)^{-1}](\mu(1) - \mu(0)) - 2N\Big). \tag{4.8}$$

Clearly, both of these criteria are different from the Battacharrya bound based criterion given in (4.6), which we use for model discrimination.

4.2.4 Bounding the Bayes Risk for Multiple Models

In this section we introduce a new bound on the probability of model selection error that applies to an arbitrary number of models, or hypotheses.

Theorem 4.1. *When performing hypothesis selection between an arbitrary number of hypotheses, for Gaussian observation distributions such that* $p(\mathbb{Y}|H_i) = \mathcal{N}(\mu(i), \Sigma(i))$ *and* $p(\mathbb{Y}|H_j) = \mathcal{N}(\mu(j), \Sigma(j))$, *the Bayes Risk is upper bounded as follows:*

$$P(error) \leq \sum_i \sum_{j>i} P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \tag{4.9}$$

where:

$$k(i,j) = \frac{1}{4}(\mu(j) - \mu(i))' \left[\Sigma(i) + \Sigma(j)\right]^{-1} (\mu(j) - \mu(i)) + \frac{1}{2} \ln \frac{\left|\frac{\Sigma(i)+\Sigma(j)}{2}\right|}{\sqrt{|\Sigma(i)||\Sigma(j)|}}.$$

Proof: The key idea is to express the n-hypothesis Bayes Risk in terms of the Bayes Risk between pairs of hypotheses, and then to bound the term for each pair in a manner analogous to the Bhattacharrya bound.

The n-hypothesis Bayes Risk given by (4.3) can be expressed in terms of hypothesis pairs as follows:

$$P(error) = \sum_i \sum_{j \neq i} P(\mathbb{Y} \in \Re_j, H_i|\mathbb{U}) = \sum_i \sum_{j>i} \Big(P(\mathbb{Y} \in \Re_j, H_i|\mathbb{U}) + P(\mathbb{Y} \in \Re_i, H_j|\mathbb{U})\Big). \tag{4.10}$$

Here, each term in the summation is the Bayes Risk between hypothesis i and hypothesis j. This can be written exactly as:

$$\Big(P(\mathbb{Y} \in \Re_j, H_i|\mathbb{U}) + P(\mathbb{Y} \in \Re_i, H_j|\mathbb{U})\Big) = \int_{\Re_j} p(\mathbb{Y}|H_i, \mathbb{U})P(H_i)d\mathbb{Y} + \int_{\Re_i} p(\mathbb{Y}|H_j, \mathbb{U})P(H_j)d\mathbb{Y}. \tag{4.11}$$

We now define two regions $\Re_A$ and $\Re_B$ as follows:

$$\begin{aligned} \Re_A &= \{\mathbb{Y}|p(H_i, \mathbb{Y}|\mathbb{U}) > p(H_j, \mathbb{Y}|\mathbb{U})\} \\ \Re_B &= \{\mathbb{Y}|p(H_j, \mathbb{Y}|\mathbb{U}) > p(H_i, \mathbb{Y}|\mathbb{U})\}. \end{aligned} \tag{4.12}$$

We can relate these regions to the decision regions $\Re_i$ and $\Re_j$. From the definition of hypothesis selection given in Section 4.2.2, decision region $\Re_i$ is where the likelihood of hypothesis i is greater than all other hypotheses:

$$\begin{aligned} \Re_i &= \{\mathbb{Y}|p(H_i, \mathbb{Y}|\mathbb{U}) > p(H_l, \mathbb{Y}|\mathbb{U}) \quad \forall l \neq i\} \\ \Re_j &= \{\mathbb{Y}|p(H_j, \mathbb{Y}|\mathbb{U}) > p(H_l, \mathbb{Y}|\mathbb{U}) \quad \forall l \neq j\}. \end{aligned} \tag{4.13}$$

It is clear from (4.12) and (4.13) that the decision regions are subsets of the regions $\Re_A$ and $\Re_B$, such that $\Re_i \subseteq \Re_A$ and $\Re_j \subseteq \Re_B$. We can therefore bound the 2-hypothesis Bayes Risk term in (4.11) as follows:

$$\begin{aligned} &\textstyle\int_{\Re_j} p(\mathbb{Y}|H_i, \mathbb{U})P(H_i)d\mathbb{Y} + \int_{\Re_i} p(\mathbb{Y}|H_j, \mathbb{U})P(H_j)d\mathbb{Y} \leq \\ &\textstyle\int_{\Re_B} p(\mathbb{Y}|H_i, \mathbb{U})P(H_i)d\mathbb{Y} + \int_{\Re_A} p(\mathbb{Y}|H_j, \mathbb{U})P(H_j)d\mathbb{Y}. \end{aligned} \tag{4.14}$$

The key idea in the derivation of the Bhattacharyya bound is to express two integrals over different decision regions, as a single integral over the entire space of $\mathbb{Y}$. The integral over decision regions is intractable, because it involves integration of Gaussian distributions with definite limits. The integral over the entire space, on the other hand, can be evaluated in closed form. In an analogous manner, we now note that the union of the regions $\Re_A$ and $\Re_B$ is the entire space of $\mathbb{Y}$. Using the definitions in (4.12), we can therefore write the integrals in (4.14) as a single integral over the entire space of $\mathbb{Y}$:

$$\begin{aligned} &\int_{\Re_B} p(\mathbb{Y}|H_i, \mathbb{U})P(H_i)d\mathbb{Y} + \int_{\Re_A} p(\mathbb{Y}|H_j, \mathbb{U})P(H_j)d\mathbb{Y} \\ &= \int_{\mathbb{Y}} \min\{p(\mathbb{Y}|H_i, \mathbb{U})P(H_i), p(\mathbb{Y}|H_j, \mathbb{U})P(H_j)\}d\mathbb{Y}. \end{aligned} \tag{4.15}$$

We now use the following inequality[40]:

$$\min\{a,b\} \leq a^{\frac{1}{2}} b^{\frac{1}{2}}, \tag{4.16}$$

which means that the integral over $\mathbb{Y}$ can be bounded as follows:

$$\begin{aligned}&\int_{\mathbb{Y}} \min\{p(\mathbb{Y}|H_i,\mathbb{U})P(H_i), p(\mathbb{Y}|H_j,\mathbb{U})P(H_j)\} d\mathbb{Y} \\ &\leq P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} \int_{\mathbb{Y}} p^{\frac{1}{2}}(\mathbb{Y}|H_i,\mathbb{U}) p^{\frac{1}{2}}(\mathbb{Y}|H_j,\mathbb{U}) d\mathbb{Y}.\end{aligned} \tag{4.17}$$

Furthermore, if the distributions are Gaussian, such that $p(\mathbb{Y}|H_i) = \mathcal{N}(\mu(i), \Sigma(i))$ and $p(\mathbb{Y}|H_j) = \mathcal{N}(\mu(j), \Sigma(j))$, then this integral can be evaluated in closed form to give:

$$\int_{\mathbb{Y}} p^{\frac{1}{2}}(\mathbb{Y}|H_i,\mathbb{U}) p^{\frac{1}{2}}(\mathbb{Y}|H_j,\mathbb{U}) d\mathbb{Y} = e^{-k(i,j)}, \tag{4.18}$$

where:

$$k(i,j) = \frac{1}{4}(\mu(\mathbf{j}) - \mu(\mathbf{i}))' \left[\Sigma(i) + \Sigma(j)\right]^{-1} (\mu(\mathbf{j}) - \mu(\mathbf{i})) + \frac{1}{2} \ln \frac{\left|\frac{\Sigma(i)+\Sigma(j)}{2}\right|}{\sqrt{|\Sigma(i)||\Sigma(j)|}}. \tag{4.19}$$

We therefore have:

$$P(\mathbb{Y} \in \Re_j, H_i|\mathbb{U}) + P(\mathbb{Y} \in \Re_i, H_j|\mathbb{U}) \leq P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \tag{4.20}$$

which leads to a new upper bound on the probability of model selection error:

$$P(error) \leq \sum_i \sum_{j>i} P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \tag{4.21}$$

where $k(i,j)$ is defined in (4.19). □

We have therefore introduced a new upper bound on the probability of model selection error between an arbitrary number of models. This bound can be evaluated in closed form, and is a summation over Gaussian-like forms. Notice that in the special case of only two hypotheses, the bound reduces to the Battacharyya bound mentioned in Section 4.2.3.

In Section 4.2.4 we analyze the tightness of the new bound empirically, and in Section 4.2.5 we describe how the new bound can be used as an optimization criterion for finite horizon control design.

1)	*Randomly generate M models.* Each models H_i has a prior probability $p(H_i)$, a mean $\mu(i)$ and a covariance $\Sigma(i)$ for the observation distribution.
2)	*Evaluate upper bound on error probability.* The upper bound (4.21) is evaluated for the generated model set.
3)	*Simulate observations.* The true model is chosen at random, according to the prior distribution $p(H_i)$. Then an observation $\mathbf{y}$ is drawn from the probability distribution $\mathcal{N}(\mu(i), \Sigma(i))$.
4)	*Select most likely model.* The probability $p(H_i\|\mathbf{y})$ is evaluated for each H_i, and the most likely model is identified.
5)	*Record errors.* If the most likely model is not the true model, record the selection as an error.
6)	*Repeat and calculate error probability.* Steps 1 through 5 are repeated a large number of times. The probability of model selection error is approximated as the fraction of errors recorded.

Table 4.1. Experimental process for analyzing tightness of bound on probability of Multiple Model selection error.

Empirical Tightness Evaluation

We now analyze the tightness of the bound (4.21) empirically using simulations. We use the process in Table 4.1. For analysis of the bound, we use scalar observations. Figure 4-3 shows the upper bound and the (estimated) true probability of error for $M = 5$ and a variety of randomly generated mean values $\mu(i)$; to aid visualization, in this case we have fixed $\Sigma(i)$ and $p(H_i)$. The x-axis of Figure 4-3 is the average Euclidean distance between the observation means; we choose this measure to enable visualization on a single figure. Figure 4-3 shows that the bound is not particularly tight. This is not surprising, since prior work has shown that the Battacharyya Bound is relatively loose, and the new bound (4.21) is at least as loose as the Battacharyya Bound. The bound does, however, follow the same trend as the true probability of error. This is empirical motivation for using the bound as an optimization criterion in the place of the probability of error; as the bound is minimized, the probability of error is minimized also. In Section 4.3 we show empirically with a fault detection scenario that minimizing the upper bound does, indeed, minimize the probability of error. Figure 4-4 shows the ratio between the upper bound and the true probability of error as a function of M, the number of models. In this case all of $\mu(i)$, $\Sigma(i)$ and $p(H_i)$ are generated randomly. Figure 4-4 shows that the bound becomes less tight as the number of models increases, and that the relationship is approximately linear after $M = 4$.

4.2.5 Finite Horizon Formulation

The design of control inputs to minimize the Bayes risk, as given in Def. 28, is a problem of optimal trajectory design for dynamic systems. In typical optimal trajectory design problems, an optimized sequence of control inputs is designed so that a system passes through a sequence of predicted states that minimize some cost function. In the model discrimination problem, we would like to design an optimal sequence of control inputs so that the system passes through a trajectory of state

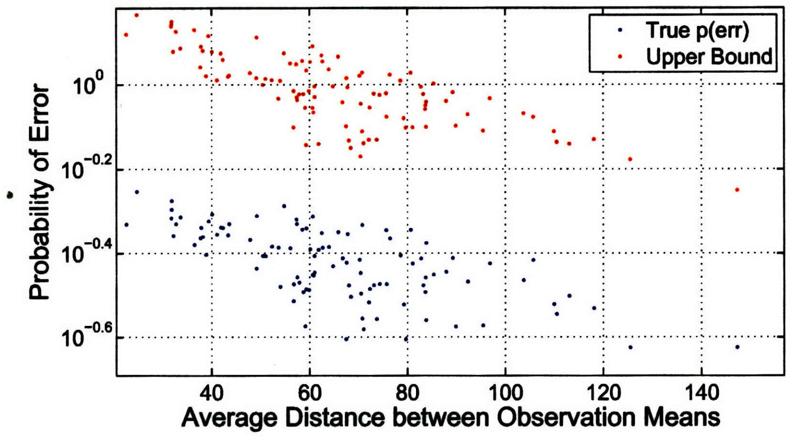


Figure 4-3. Tightness of new upper bound on the probability of Multiple Model selection error. Here there are 5 models, with randomly generated observation means. The true probability of error is estimated using a large number of simulations. The bound is not particularly tight, but follows the same trend as the probability of error.

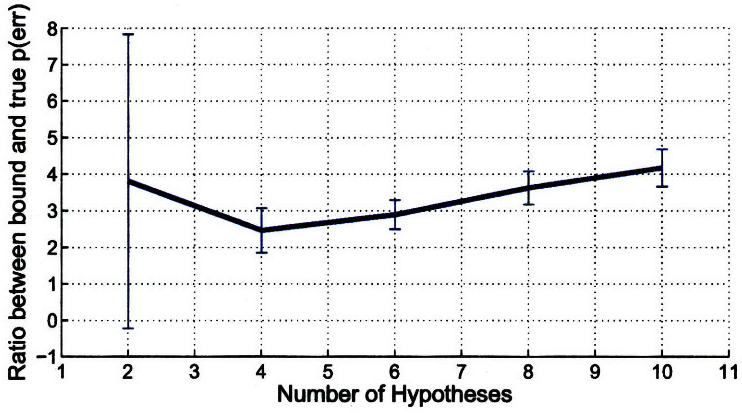


Figure 4-4. Ratio of new upper bound to the probability of Multiple Model selection error. All parameters $\mu(i)$, $\Sigma(i)$ and $p(H_i)$ were generated randomly. The bound becomes less tight as the number of models increases, the relationship is approximately linear after $M = 4$.

distributions that minimize the probability of model selection error (Bayes Risk).

Optimal trajectory design for linear, discrete-time systems can be posed as a constrained optimization problem[118]. Given a tractable optimization criterion, this problem can be solved using techniques such as Sequential Quadratic Programming (SQP)[99]. In this section we pose the problem of model discrimination between an arbitrary number of models as a finite horizon trajectory design problem, and show that it can be solved using SQP. For the special case of two models, the optimization problem can be solved using Quadratic Programming (QP) to global optimality.

A Tractable Cost Function for Model Discrimination

Optimal Model Discrimination Problem (Def. 28) consists of planning a finite sequence of inputs in order to minimize the probability of error. This form of planning is known as *finite horizon* planning. In this case, if the horizon is of length h, we are concerned with a sequence of observations $\mathbf{y}_{\tau+1}, \ldots, \mathbf{y}_{\tau+h}$ and a sequence of inputs $\mathbf{u}_{\tau}, \ldots, \mathbf{u}_{\tau+h-1}$. Since evaluation of the probability of error is intractable, we instead minimize an upper bound on the probability of error, described in Section 4.2.4. In this section we described how this bound can be used in a finite-horizon context. We define:

$$\begin{aligned} \mathbb{Y} &= \left[\begin{array}{cccc} \mathbf{y}_{\tau+1}{}' & \mathbf{y}_{\tau+2}{}' & \ldots & \mathbf{y}_{\tau+h}{}' \end{array} \right]' \\ \mathbb{U} &= \left[\begin{array}{cccc} \mathbf{u}_{\tau}{}' & \mathbf{u}_{\tau+1}{}' & \ldots & \mathbf{u}_{\tau+h-1}{}' \end{array} \right]'. \end{aligned} \tag{4.22}$$

We assume that model selection is carried out by a Bayes optimal classifier that makes its decision based on all h observations within the horizon. Due to uncertainty in the initial state and noise, the future observations $\mathbf{y}_{\tau+1}, \ldots, \mathbf{y}_{\tau+h}$ are random variables. Under the assumptions in Section 4.2.1, $\mathbf{y}_{\tau+i}$ is normally distributed given a sequence of inputs $\mathbb{U}$ and given a model H_l. We now define $\mu_{\tau+i}(l)$ and $\Sigma_{\tau+i}(l)$ for time steps $i = 1, \ldots, h$ and models H_l such that:

$$p_{\mathbf{y}_{\tau+i}(l)}(\mathbb{Y}|H_l, \mathbb{U}) = \mathcal{N}(\mu_{\tau+i}(l), \Sigma_{\tau+i}(l)). \tag{4.23}$$

Then the vector of all observations $\mathbb{Y} = [\mathbf{y}'_{\tau+1}, \ldots, \mathbf{y}'_{\tau+h}]'$ is a vector of normally distributed random variables, given a sequence of inputs and a model. We define $\mu(l)$ and $\Sigma(l)$ to be the mean and covariance of the vector of all observations such that:

$$p_{\mathbb{Y}(l)}(\mathbb{Y}|H_l, \mathbb{U}) = \mathcal{N}(\mu(l), \Sigma(l)). \tag{4.24}$$

From the above definitions the distribution of $\mathbb{Y}(l)$ is given by:

$$\mu(l) = [\mu_{\tau+1}(l)', \ldots, \mu_{\tau+h}(l)']' \tag{4.25}$$

$$[\Sigma(l)]_{i,j} = E\Big[\big([\mathbb{Y}]_i - [\mu(l)]_i\big)\big([\mathbb{Y}]_j - [\mu(l)]_j\big)\big|H_l\Big]. \tag{4.26}$$

Here $[\cdot]_i$ denotes the value at index i into the vector, and similarly $[\cdot]_{i,j}$ denotes the value at index (i, j) into the matrix.

Having defined $\mu(l)$ and $\Sigma(l)$ for all l, the bound given in (4.9) provides an upper bound for the probability of error when using the entire sequence of observations from time $\tau + 1$ to time $\tau + h$. We therefore use this bound as a cost function on the finite horizon optimization formulation:

$$J = \sum_i \sum_{j>i} P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \tag{4.27}$$

where $k(i, j)$ is defined in (4.19).

Given a model H_l, the system equations (4.1) are fully known. Hence the distribution $p(\mathbb{Y}|H_l, \mathbb{U})$ can be calculated for all l. Explicit expressions for $\mu(l)$ and $\Sigma(l)$ are found by applying the system equations (4.1) recursively to give:

$$\begin{aligned} \mathbf{y}_{\tau+i}(l) &= C(l)A(l)^i\mathbf{x_t} + D(l)\mathbf{u}_{\tau+i-1} + \nu_{\tau+i-1} \\ &\quad + C(l)\sum_{\rho=0}^{i-1} A(l)^{i-\rho-1}(B(l)\mathbf{u}_{\tau+\rho} + \omega_{\tau+\rho}). \end{aligned} \tag{4.28}$$

Given a distribution for the initial state of the system $\mathcal{N}(\hat{x}(h), P)$, the mean $\mu(h)$ and covariance $\Sigma(h)$ as defined in (4.25) and (4.26) can be calculated. The results are given here for a system where $\mathbf{y}_\tau \in \Re^n$.

Define:

$$i = n(p-1) + q, \tag{4.29}$$

where:

$$1 \le q \le n \quad 1 \le p \le h \quad p, q \in \mathbb{Z}. \tag{4.30}$$

Then following from (4.22) and (4.28):

$$\begin{aligned} [\mu(l)]_i &= [\mu_{\tau+p}(l)]_q \\ &= \left[C(l)A(l)^p\hat{\mathbf{x}}(l) + C(h)\sum_{\rho=0}^{p-1} A(l)^{p-\rho-1}B(l)u_{\tau+\rho} + D(l)u_{\tau+p-1} \right]_q \end{aligned} \tag{4.31}$$

Defining $j = n(r-1) + s$ in a similar manner to (4.29), the expression for the covariance is:

$$\begin{aligned} [\Sigma(l)]_{i,j} &= [R(l)'(p,r) + C(l)A(l)^p P(l) A(l)'^r C(l)']_{q,s} \\ &\quad + \left[\sum_{\rho=0}^{m-1} C(l)A(l)^{(p-\rho-1)} Q(l) A(l)'^{(r-\rho-1)} C(l)' \right]_{q,s} \end{aligned} \tag{4.32}$$

where $m = min\{i, j\}$ and:

$$R(l)'(p,r) = \begin{cases} R(l) & p = r \\ 0 & p \neq r. \end{cases} \tag{4.33}$$

Under the assumptions mentioned in Section 4.2.1, these equations give an expression for the belief state $p(\mathbb{Y}|H_l, \mathbb{U})$ in terms of a sufficient statistic for $\mathbb{Y}$, namely the mean and covariance of the distribution. There are two important properties to note:

1. The equation for the mean of the predicted distribution of $\mathbb{Y}$ is linear in the control inputs $\mathbb{U}$.

2. The covariance of the predicted distribution of $\mathbb{Y}$ is not a function of the control inputs $\mathbb{U}$.

These properties mean that the multiple model criterion in (4.27) has a tractable form, enabling it to be used in a constrained optimization formulation. Furthermore, for the two-model case, these properties mean that the criterion in (4.6) can be simplified to:

$$J' = -(\mu(1) - \mu(0))' \left[\Sigma(0) + \Sigma(1)\right]^{-1} (\mu(1) - \mu(0)). \tag{4.34}$$

Since both $\mu(1)$ and $\mu(0)$ are linear functions of $\mathbb{U}$, (4.34) is quadratic in the control inputs $\mathbb{U}$. This means that the two-model discrimination problem subject to linear constraints (Section 4.2.5) can be solved using Quadratic Programming.

Remark 4.1. *Since the covariance matrices $\Sigma(0)$ and $\Sigma(1)$ are positive definite, the cost function given by (4.34) is a concave function of $(\mu(1) - \mu(0))$. Both $\mu(1)$ and $\mu(0)$ are linear functions of $\mathbb{U}$, and hence (4.34) is also a concave function of the control inputs $\mathbb{U}$. This concavity makes the cost function particularly tractable for optimization and guarantees that a global optimum can be found in bounded time [102].*

Linear Constraints

A powerful aspect of the constrained finite horizon formulation is that optimal input sequences can be found subject to hard constraints on the control inputs. This can be used to model actuator saturation, for example, by constraining $u_{min} < u_{\tau+i} < u_{max}$. Constraints on the expected system state and inputs can be expressed as linear constraints on the control inputs. The expected system state conditioned on a model H_l is a linear function of the control inputs:

$$E[\mathbf{x}_{\tau+i}|H_l] = A(l)^i \hat{\mathbf{x}}_\tau + \sum_{j=0}^{i-1} A(l)^{i-j-1}(B(l)\mathbf{u}_{\tau+j}), \tag{4.35}$$

where $\hat{\mathbf{x}}_\tau$ is the mean of the initial system state. Hence constraints on the expected system state of the form $E[\mathbf{x}_{\tau+i}|H_l] = goal$ or $\mathbf{x}_{min} < E[\mathbf{x}_{\tau+i}|H_l] < \mathbf{x}_{max}$ are linear constraints in the control inputs. By imposing such constraints, we can:

1. Ensure that a certain task, defined in terms of the expected system state, is fulfilled
2. Ensure that the expected system state stays within a 'safe' operating region or within a valid linearization region
3. Ensure that the system state ends the experiment in the same region as it started.

Here, we have restricted our attention to linear constraints, since these are straightforward to encode, and are guaranteed to be convex; convexity simplifies the optimization problem greatly. The general formulation, however, applies to nonlinear constraints.

Summary

We have shown that the problem of designing a sequence of optimal control inputs to discriminate between an arbitrary number of models can be posed as a finite-horizon trajectory design problem. The resulting AE-MM algorithm, summarized in Table 4.2 works by minimizing a novel, closed form upper bound on the probability of model selection error, and imposing constraints on the expected system state and control inputs to ensure that a defined task is fulfilled and that actuator limits are not violated. This optimization can be solved using existing methods such as Sequential Quadratic Programmming. In the case of discrimination between two models, the optimization can be solved using Quadratic Programming; furthermore in this case the global optimum can be found in finite time.

Function ActiveMultipleModelMain($\mathcal{H}$,$p(\mathcal{H},h)$) **returns** $\mathbf{u}_{\tau:\tau+h-1}$

1) For each model H_i, calculate the mean $\mu(h)$ as a linear function of the control inputs, according to (4.31).
2) For each model H_i calculate the covariance $\Sigma(h)$ according to (4.32).
3) Using Sequential Quadratic Programming, minimize over $\mathbf{u}_{\tau:\tau+h-1}$:

$$\sum_i \sum_{j>i} P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \qquad (4.36)$$

where $k(i,j)$ is defined in (4.19), in terms of $\mu(h)$ and $\Sigma(h)$, subject to:

- Constraints on the expected state, for example $\mu_{\tau+i}(l) \leq \mu_{max}$ or $\mu_{\tau+j}(l) = \mu_{eq}$.
- Constraints on the control inputs, for example $\mathbf{u}_{\tau+i} \leq \mathbf{u}_{max}$.

Table 4.2. AE-MM Algorithm for Optimal Discrimination between Multiple Linear Models.

4.3 Simulation Results: Multiple-Model Discrimination

In this section we demonstrate the AE-MM algorithm using an aircraft fault detection scenario. We use a discrete time approximation to the longitudinal aircraft dynamics, linearized about the trim state, shown in Figure 4-5. Here, the state of the system consists of the horizontal velocity, the vertical velocity, the pitch angle, and the pitch rate. The observed output of the system is taken to be the pitch rate $\dot{\theta}$ and the vertical velocity V_y. The input is denoted u, and is taken to be the requested elevator angle. We assume that the elevator actuator saturates at $\pm 0.25 rad$. The terms w_t and v_t are the process noise and observation noise respectively. The noise at any time step is assumed to be independent of the noise at any other time step, and ω_t and ν_t are independent of each other with normal distributions $\mathcal{N}(0,Q)$ and $\mathcal{N}(0,R)$ respectively. The initial state of the system is also assumed to be normally distributed.

In the Multiple-Model selection task, we must determine which model $H(i)$ is most likely. Under

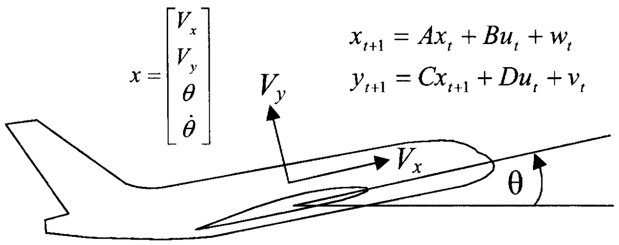

$x = \begin{bmatrix} V_x \\ V_y \\ \theta \\ \dot{\theta} \end{bmatrix}$
$x_{t+1} = Ax_t + Bu_t + w_t$
$y_{t+1} = Cx_{t+1} + Du_t + v_t$
V_y
V_x
θ

Figure 4-5. Discrete-time aircraft model linearized about the trim state.

model H_i, the system is described by $\{A(i),B(i),C(i),D(i),Q(i),R(i)\}$. For the aircraft example, we consider three single-point failures; the pitch rate sensor may fail, the vertical velocity sensor may fail, or the elevator actuator may fail. This gives the following alternative models:

- H_0: Nominal (no faults)
- H_1: Faulty pitch rate sensor
- H_2: Faulty vertical velocity sensor
- H_3: Faulty elevator actuator

In the case of sensor faults, we assume that the sensor reading is zero mean white noise, while in the case of the actuator fault we assume that the elevator exerts no control effort. Consistent with a Multiple-Model fault detection framework, we assume that the system matrices $\{A, B, C, D, Q, R\}$ are fully known for each of the possible faults, and that the true model persists indefinitely. In other words, in this section we do not allow switching between the different models; switching dynamics are considered in Section 4.4.

In each of the following discrimination tasks we control the system by constraining the expected state, conditioned on nominal operation. In Section 4.3.1 we constrain the aircraft to remain within an altitude envelope. For the sake of comparison, in Section 4.3.2 we show results from a manually generated identification sequence, and in Section 4.3.3 we show results from an altitude-hold sequence with an added auxiliary signal. In Section 4.3.4 the aircraft carries out an altitude change maneuver, while optimally detecting faults. We compare the resulting sequence with a fuel-optimal sequence.

We evaluate the efficacy of the AE-MM approach in terms of the reduction in the probability of model selection error achieved by using the discrimination-optimal sequence. There are two different error probabilities that we consider:

1. *Batch Error Probability.* This error probability is based on the assumption that Bayesian model selection is carried out based on all the observations over the entire horizon. This value is calculated in closed form.

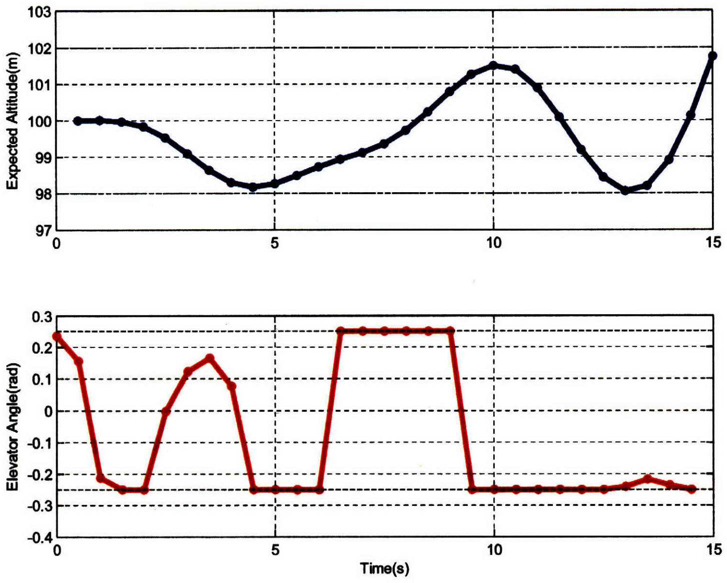


Figure 4-6. Discrimination-optimized input design for aircraft flight envelope scenario. Top: Expected altitude of aircraft given nominal operation. Bottom: Optimized sequence of control inputs. The optimal control discriminates between the different models while ensuring that in the nominal case, the aircraft altitude remains between 98 and $102m$. The optimized control input yields a batch error probability of 0.0003 and a sequential error probability of 0.0004.

2. *Sequential Error Probability.* This is the probability of a sequential Multiple-Model estimation scheme making a model selection error. This probability is estimated by carrying out a large number of simulations.

As discussed in Section 4.9 AE-MM minimizes an upper bound on the batch error probability although most practical Multiple-Model implementations are sequential. Nevertheless, the results provided here show that the new approach dramatically reduces the sequential error probability also.

4.3.1 Altitude Envelope

Figure 4-6 and Figure 4-7 show results from a fault detection scenario where the aircraft is constrained to remain within a flight envelope around an altitude of $100m$. The elevator angle is constrained to be at most $0.25rad$ in magnitude. The prior probabilities of models H_0 through H_3 are 0.65, 0.1, 0.05 and 0.2 respectively. We assume that these priors have been generated by a Multiple Model estimator running up until time t.

The optimized control input yields a batch error probability of 0.0003 and a sequential error probability of 0.0004. The *bang-bang* nature of the optimized control input highlights the fact that by optimizing up against hard constraints the discrimination power of the input signal can be much greater than a power-bounded auxiliary signal. This particular solution took $58.5s$ to generate, of which $30.7s$ was used to generate the necessary covariance matrices and of which $27.8s$ was used to perform the nonlinear optimization using Matlab's `fmincon` function.

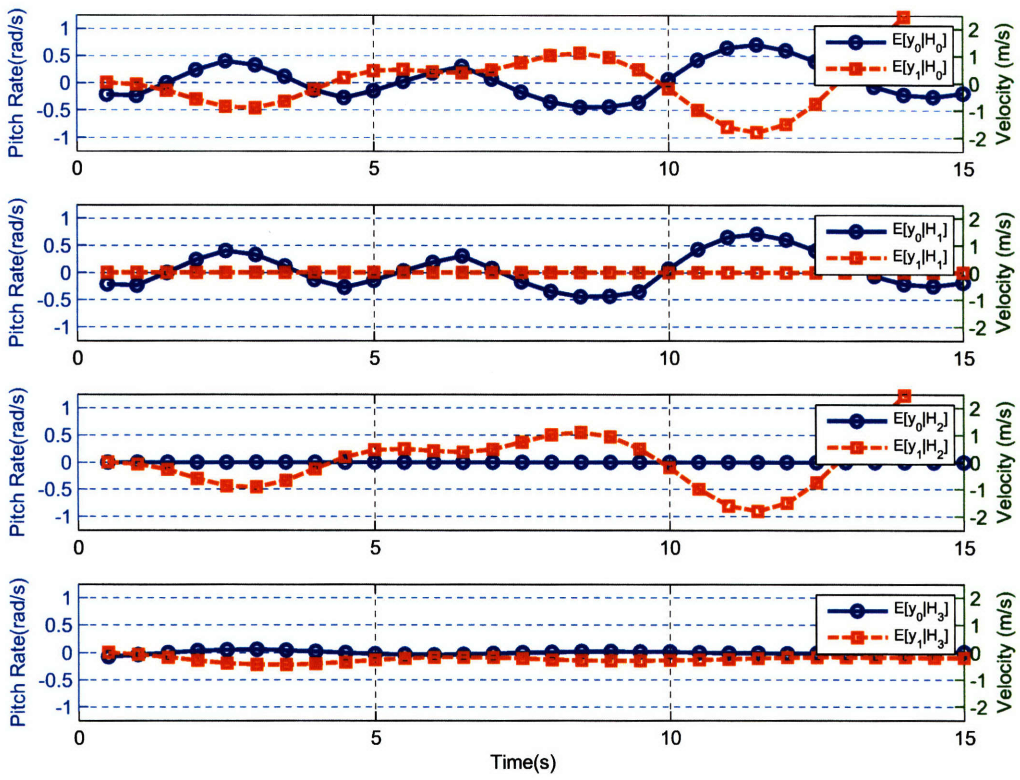


Figure 4-7. Expected observations for aircraft flight envelope scenario with optimized control input. The top plot shows the nominal case, where there are no faults. In the second and third plots, the pitch rate sensor and vertical velocity sensors are faulty, respectively. In the bottom plot, the elevator actuator is faulty. The optimized control input ensures that the observation sequences in each case are as different as possible, in order to minimize the probability of model selection error.

4.3.2 Manually Generated Sequence

In order to identify the longitudinal dynamics of an aircraft, pilots typically use a doublet control input[68]. Figure 4-8 shows such a control input sequence, with the same actuator limits as for the optimized control sequence. This sequence yields a batch error probability of 0.0269 and a sequential error probability of 0.0258. Hence the optimized sequence in Figure 4-6 has significantly greater discrimination power than the manually generated sequence.

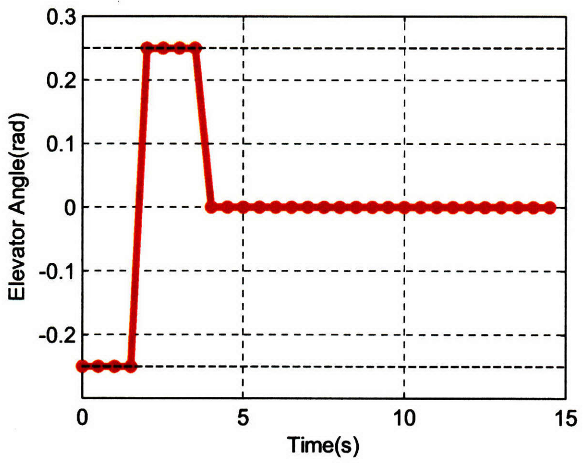


Figure 4-8. Typical manually generated identification sequence. This doublet form is used by pilots to perform aircraft system identification. This sequence yields a batch error probability of 0.0269 and a sequential error probability of 0.0258.

4.3.3 Auxiliary Signal

A number of existing approaches to control design for discrimination and model identification add a low power auxiliary signal to the nominal control sequence[98, 138, 71]. The power of the signal is small so that the effect on the system state is small. A typical approach to auxiliary signal design uses a Pseudo-Random Binary Signal (PRBS)[50].

Figure 4-9 shows an altitude-hold control signal with a typical PRBS auxiliary signal added. The PRBS signal was constrained to have a maximum elevator angle of 0.01 in order to ensure that the aircraft altitude did not deviate significantly from $100m$. Averaged over a number of randomly-generated signals, the resulting batch error probability was 0.3455, and the sequential error probability was 0.3510, which is far worse than the values obtained using the new discrimination approach.

4.3.4 Altitude Change Maneuver

The AE-MM algorithm can use constraints to ensure that a given control task is performed, while optimizing with respect to discrimination. This is demonstrated in Figure 4-10, where the aircraft carries out a maneuver that changes its altitude from $100m$ to $120m$, conditioned on the elevator

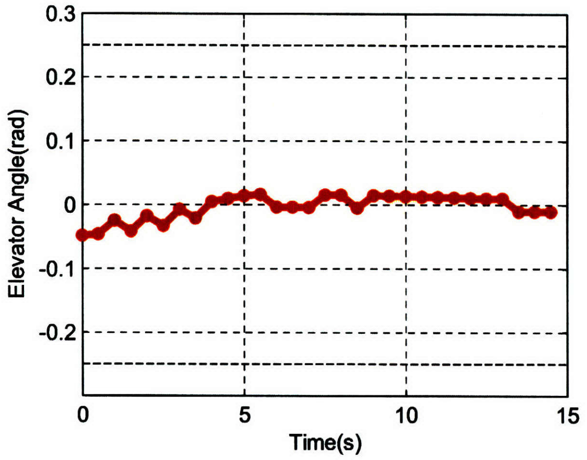


Figure 4-9. Typical control sequence with added Pseudo-Random Binary auxiliary signal. This sequence yields a batch error probability of 0.3360 and a sequential error probability of 0.3406.

actuator being functional. The discrimination-optimal control sequence is compared to the fuel-optimal one. Optimizing with respect to discrimination yields a batch error probability of 0.0006 and a sequential error probability of 0.0001. Optimizing with respect to fuel yields a batch error probability of 0.1914 and a sequential error probability of 0.1833. Hence a dramatic improvement in fault detection can be achieved by using control inputs designed for model discrimination, rather than those designed to optimize some other criterion and employing only passive model selection. This particular solution took $51.4s$ to generate, of which $30.9s$ was spent calculating the necessary covariances and $20.5s$ was spent performing the nonlinear optimization.

4.4 Active Estimation for Jump Markov Linear Systems

We now extend the multiple-model discrimination method to develop an active estimation capability for Jump Markov Linear Systems, which we call the AE-JMLS algorithm. By extending the error bound derived for discrimination between different models, to time-varying systems, we create a tractable upper bound on the probability of the true mode sequence being pruned. We then use a constrained finite horizon control design approach to ensure that a given control task is achieved, conditioned on nominal system operation.

4.4.1 Problem Statement

In this section we repeat the definition of a Jump Markov Linear System (JMLS) and describe how approximate state estimation can be carried out for such systems. A JMLS is a special case of a Probabilistic Hybrid Automaton (Def. 2) for which mode transitions do not depend on the continuous dynamics, and the continuous dynamics are linear in each mode. The Bayes Network representation of a JMLS is repeated in Figure 4-11. Furthermore, in this chapter we consider only

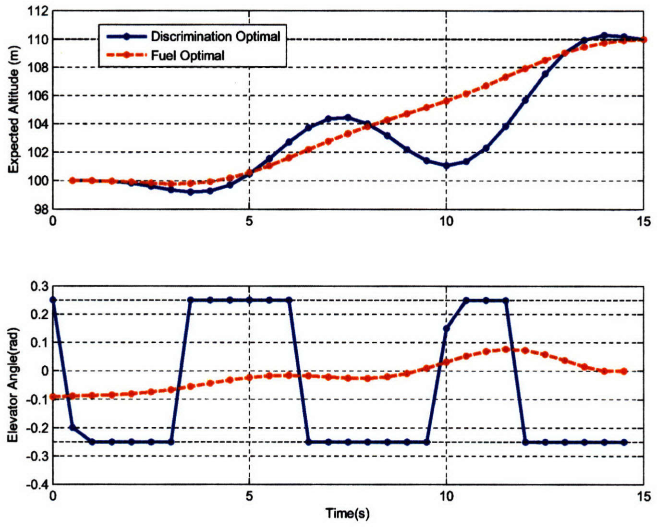


Figure 4-10. Discrimination-optimal and fuel-optimal control design for altitude change maneuver. The discrimination-optimal sequence gives a batch error probability of 0.0006 and a sequential error probability of 0.0001, while the fuel-optimal sequence gives a batch error probability of 0.1914 and a sequential error probability of 0.1833.

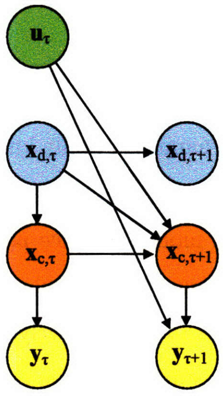


Figure 4-11. Bayes Network showing conditional dependencies between variables at time steps τ and $\tau + 1$ in a Jump Markov Linear System. Note that the discrete state does not depend on the control input or the continuous state, unlike in a general PHA.

Gaussian uncertainty; this is in contrast to Chapter 3 where we place no restrictions on the noise distribution. The continuous dynamics of a JMLS are defined by:

$$\begin{aligned}\mathbf{x}_{c,\tau+1} &= A(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau} + B(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \omega_\tau \\ \mathbf{y}_{\tau+1} &= C(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau+1} + D(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \nu_\tau,\end{aligned} \tag{4.37}$$

where $\mathbf{x}_{d,\tau}$ is a Markov chain that evolves according to a transition matrix $\mathcal{T}$ such that:

$$p(\mathbf{x}_{d,\tau+1} = j|\mathbf{x}_{d,\tau} = i) = [\mathcal{T}]_{ij}. \tag{4.38}$$

The variables ω and ν are zero-mean Gaussian white noise processes with covariance $Q(\mathbf{x}_{d,\tau})$ and $R(\mathbf{x}_{d,\tau})$, respectively. This system is a JMLS; the continuous dynamics depend on the discrete mode $\mathbf{x}_{d,\tau}$, which switches stochastically. There are $|\mathcal{X}_d|$ discrete modes, such that $\mathbf{x}_{d,\tau} \in \{1, \ldots, |\mathcal{X}_d|\}$. In the JMLS formulation, as opposed to the multiple-model formulation, switching between models is allowed; stochastic jumps represent component failures or recoveries from failure.

We define the problem of hybrid estimation in a JMLS as that of estimating the probability distribution $p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}|\mathbf{y}_{1:\tau})$ over the hybrid discrete-continuous state, conditioned on the sequence of all observations $\mathbf{y}_{1:\tau}$. This probability can be written as a sum over all possible mode sequences that end in the mode $\mathbf{x}_{d,\tau}$:

$$p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}|\mathbf{y}_{1:\tau}) = \sum_{\mathbf{x}_{d,1:\tau-1}} p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau}). \tag{4.39}$$

Each summand can be further expanded as a product of the posterior probability of the discrete mode sequence $m_{1:\tau}$ and the posterior distribution over the continuous state, conditioned on this mode sequence:

$$p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau}) = p(\mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau})p(\mathbf{x}_{c,\tau}|\mathbf{x}_{d,1:\tau}, \mathbf{y}_{1:\tau}). \tag{4.40}$$

For a given mode sequence, the system dynamics are fully known, although time-varying. This means that the probability distribution $p(\mathbf{x}_{c,\tau}|\mathbf{x}_{d,1:\tau}, \mathbf{y}_{1:\tau})$ can be calculated exactly using the Kalman Filter recursion[70]. The probability of a given mode sequence $p(\mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau})$ can also be calculated using the residuals in the Kalman filter equations. In principle, therefore, it is possible to calculate the distribution over the hybrid state $p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}|\mathbf{y}_{1:\tau})$ exactly, yielding a sum-of-Gaussians expression. In practice, however, this *exact* hybrid state estimation is infeasible since the number of mode sequences $\mathbf{x}_{d,1:\tau}$ grows exponentially with time and with the number of possible modes.

4.4.2 Approximate Hybrid Estimation

A large number of approximate methods have been proposed that make the problem tractable by approximating the probability $p(\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}|\mathbf{y}_{1:\tau})$ [101, 130, 20]. One common approach is to discard mode sequences that have a low posterior probability $p(\mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau})$. Such *pruning* approaches typically ensure that a fixed number of mode trajectories are tracked. In this thesis, we assume that a pruning approach is used so that K individual mode sequences are tracked; this is called *K-best hybrid estimation.* While pruning is usually carried out at every time step, for the purposes of finite-horizon control design, we assume that pruning is carried out at the end of the control horizon. The reasons for, and implications of, this assumption are discussed in Section 4.9. Figure 4-12 shows the pruning process for a time horizon of h time steps and $K = 4$ tracked mode sequences.

It is possible for the true mode sequence to be discarded in this pruning process. If this occurs, the hybrid estimator typically diverges and the approximated state distribution no longer resembles the true distribution. The goal of active estimation for JMLS is to use control inputs to minimize the probability of the true mode sequence being pruned.

Definition 29. *At time step τ, given a JMLS $\mathcal{M}$ and an estimate for the hybrid state $p(\mathbf{x}_c, \mathbf{x}_d)$, the* Active Hybrid Estimation Problem *consists of designing a finite sequence of control inputs $\mathbf{u}_{\tau:\tau+h-1}$ that minimizes the probability of K-best hybrid estimation discarding the true mode sequence.*

We now demonstrate that calculating the probability of hybrid estimation pruning the true mode sequence cannot be calculated in closed form. Before the start of the horizon, at time step τ, the observations $\mathbf{y}_{\tau:\tau+h}$ are unknown. However the probability of pruning the true mode sequence, which we denote $\mathbf{x}_{d,1:\tau}{}^*$, can be expressed by marginalizing over all possible observations:

$$p(prune) = \int_{\mathbf{y}_{\tau:\tau+h}} p(E|\mathbf{y}_{1:\tau+h})p(\mathbf{y}_{\tau:\tau+h}|\mathbf{y}_{1:\tau})d\mathbf{y}_{\tau+1:\tau+h}, \tag{4.41}$$

where E is the event that the posterior probability of the true mode sequence $\mathbf{x}_{d,1:\tau}{}^*$ is not in the top K posteriors:

$$E \Longleftrightarrow \{p(\mathbf{x}_{d,1:\tau}{}^*|\mathbf{y}_{1:\tau+h}) < p(\mathbf{x}_{d,1:\tau}{}^{(i)}|\mathbf{y}_{1:\tau+h}) \; for \text{ K } or\ more \text{ i}\}. \tag{4.42}$$

The posterior probability $p(\mathbf{x}_{d,1:\tau}{}^{(i)}|\mathbf{y}_{1:\tau+h})$ can be calculated for a given mode sequence $\mathbf{x}_{d,1:\tau}{}^{(i)}$ and a given observation sequence $\mathbf{y}_{\tau:\tau+h}$, since the past observations $\mathbf{y}_{1:\tau-1}$ are known. The probability of a given observation sequence $p(\mathbf{y}_{\tau+1:\tau+h}|\mathbf{y}_{1:\tau-1})$ can be calculated as follows:

$$p(\mathbf{y}_{\tau+1:\tau+h}|\mathbf{y}_{1:\tau-1}) = \sum_i p(\mathbf{y}_{\tau+1:\tau+h}|\mathbf{x}_{d,1:\tau+h}{}^{(i)}, \mathbf{y}_{1:\tau})p(\mathbf{x}_{d,1:\tau+h}{}^{(i)}|\mathbf{y}_{1:\tau}). \tag{4.43}$$

In this equation, $p(\mathbf{x}_{d,1:\tau+h}{}^{(i)}|\mathbf{y}_{1:\tau})$ is the prior probability of the mode sequence $p(\mathbf{x}_{d,1:\tau+h}{}^{(i)})$. Calculation of this value is straightforward, as described in Section 4.4.4. Conditioned on a mode sequence, the distribution $p(\mathbf{y}_{\tau:\tau+h}|\mathbf{x}_{d,1:\tau+h}{}^{(i)}, \mathbf{y}_{1:\tau-1})$ over the observation sequences can be calculated using the Kalman filter update equations. This is described in detail in Section 4.4.3. The key point is that the terms in the integral (4.41) can be calculated in closed form; however the integral itself cannot be evaluated. Hence the probability of pruning the true mode sequence cannot be used as a criterion for optimization. In the same spirit as the AE-MM approach developed in Section 4.2, we therefore derive a tractable *upper bound* on the probability of pruning the true mode sequence. We then approximate the Active Hybrid Estimation Problem (Def. 29) by minimizing this bound instead of the true probability of pruning.

4.4.3 Bounding the Probability of Pruning

In Section 4.2.4 we introduced a new upper bound on the probability of error when selecting between an arbitrary number of stochastic linear dynamic systems. In this section we extend this work to create a bound on the probability of pruning the true mode sequence in hybrid state estimation for JMLS. The key insight behind this extension is that, conditioned on a mode sequence, the system has known, time-varying linear dynamics. We can enumerate each possible mode sequence over the over the planning horizon. Hence each mode sequence can be considered a *hypothesis* in the sense that only one mode sequence is the true one, and the approximate hybrid estimation algorithm described in Section 4.4.1 chooses the most *a posteriori* likely K hypotheses based on the available observations. The ideas for selection between known linear models developed in Section 4.2 therefore extend naturally to active hybrid estimation for JMLS.

Considering each mode sequence $m^{(i)}_{\tau:\tau+h}$ as a hypothesis H_i, the expression in (4.9) gives an upper bound on the probability that the true mode sequence is not selected as the *most likely* hypothesis. This is greater than or equal to the probability that the true mode sequence is not selected among the K most likely hypotheses, which is in turn the probability of pruning defined in (4.41). Hence (4.9) gives an upper bound on the probability of the true hypothesis being pruned, as required.

The bound in (4.9) applies for a general vector of observations $\mathbb{Y}$ with a multivariate Gaussian distribution. In order for this bound to be tractable for optimization, however, we must be able to calculate the mean and covariance of this distribution.

As before we define:

$$\begin{aligned}\mathbb{Y} &= \left[\begin{array}{cccc} \mathbf{y}'_{\tau+1} & \mathbf{y}'_{\tau+2} & \cdots & \mathbf{y}'_{\tau+h} \end{array}\right]' \\ \mathbb{U} &= \left[\begin{array}{cccc} \mathbf{u}'_{\tau} & \mathbf{u}'_{\tau+1} & \cdots & \mathbf{u}'_{\tau+h-1} \end{array}\right]'. \end{aligned} \tag{4.44}$$

We define $\mathbb{Y}(i)$ as the block vector of all observations over the time horizon, conditioned on mode sequence $\mathbf{x}_{d,1:\tau+h}(i)$. The distribution of $\mathbb{Y}(i)$ is given by:

$$\mu(i) = [\mu_{\tau+1}(i)', \ldots, \mu_{\tau+h}(i)']' \tag{4.45}$$

$$[\Sigma(i)]_{f,g} = E\Big[\big([\mathbb{Y}]_f - [\mu(i)]_f\big)\big([\mathbb{Y}]_g - [\mu(i)]_g\big)\big|\mathbf{x}_{d,1:\tau+h}(i), \mathbf{y}_{1:\tau+h}\Big], \tag{4.46}$$

where $[\cdot]_f$ denotes the (f)'th index into the array, and $[\cdot]_{f,g}$ denotes the (f,g)'th index into the matrix. The mean and covariance expressions can be calculated explicity in terms of the control inputs using repeated application of the Kalman Filter update equations. Given a distribution for the state at time step τ such that $p(\mathbf{x}_{c,\tau}|\mathbf{y}_{1:\tau}, \mathbf{x}_{d,1:\tau}(i)) = \mathcal{N}(\hat{x}(i), P(i))$, the mean $\mu(i)$ and covariance $\Sigma(i)$ as defined in (4.45) can be calculated. The results are given here for a system where $\mathbf{y}_\tau \in \Re^n$.

Define:

$$f = n(p-1) + q, \quad \text{where} \quad 1 \le q \le n \quad 1 \le p \le h \quad p, q \in \mathbb{Z}. \tag{4.47}$$

Then the mean is given by:

$$\begin{aligned}
[\mu(i)]_f &= [\mu_{\tau+p}(i)]_q \\
&= C(\mathbf{x}_{d,\tau+p})\Big(\prod_{l=1}^{p} A(\mathbf{x}_{d,\tau+l})\Big)\hat{\mathbf{x}}_0 + C(\mathbf{x}_{d,\tau+p})\sum_{l=0}^{p-1}\Big(\prod_{v=l}^{p-1} A(\mathbf{x}_{d,\tau+v})\Big)B(\mathbf{x}_{d,\tau+l})\mathbf{u}_{\tau+l} + D(\mathbf{x}_{d,\tau+p})\mathbf{u}_{\tau+p-1}.
\end{aligned} \tag{4.48}$$

Defining $g = n(r-1) + s$ in the same manner as (4.47), the expression for the covariance is:

$$\begin{aligned}
\left[\Sigma^{(i)}\right]_{f,g} =& R(\mathbf{x}_{d,\tau+p})'(p,r) + C(\mathbf{x}_{d,\tau+p})\Big(\prod_{v=1}^{p} A(\mathbf{x}_{d,\tau+v})\Big)P^{(i)}\Big(A(\mathbf{x}_{d,\tau+w})'\prod_{w=1}^{r}\Big)C(\mathbf{x}_{d,\tau+r})' \\
&+ \sum_{l=0}^{m-1} C(\mathbf{x}_{d,\tau+p})\Big(\prod_{v=l+2}^{p} A(\mathbf{x}_{d,\tau+v})\Big)Q(\mathbf{x}_{d,\tau+p})\Big(A(\mathbf{x}_{d,\tau+w})^T\prod_{w=l+2}^{r}\Big)C(\mathbf{x}_{d,\tau+r})',
\end{aligned} \tag{4.49}$$

where $m = min\{f, g\}$ and:

$$R(\mathbf{x}_{d,\tau})'(p,r) = \begin{cases} R(\mathbf{x}_{d,\tau}) & p = r \\ 0 & p \ne r. \end{cases} \tag{4.50}$$

We use the following notation regarding matrix products. Repeated right matrix products are denoted:

$$\Big(\prod_{i=1}^{p} A(\mathbf{x}_{d,i})\Big) = A(\mathbf{x}_{d,1})A(\mathbf{x}_{d,2}), \ldots, A(\mathbf{x}_{d,p-1})A(\mathbf{x}_{d,p}), \tag{4.51}$$

while repeated left matrix products are denoted:

$$\Big(A(\mathbf{x}_{d,i})\prod_{i=1}^{p}\Big) = A(\mathbf{x}_{d,p})A(\mathbf{x}_{d,p-1}), \ldots, A(\mathbf{x}_{d,2})A(\mathbf{x}_{d,1}). \tag{4.52}$$

In principle, therefore, we can use the bound in (4.9) as an optimization criterion for active hybrid estimation. This is in contrast to the exact value (4.41), which cannot be evaluated in closed form.

However the new criterion requires evaluating $\mathcal{O}(N_{seqs}^2)$ terms, where N_{seqs} is the number of mode sequences being considered. There are $|\mathcal{X}_d|^h$ possible mode sequences over a horizon of h time steps, which means that even for a relatively short time horizon and a modest number of possible modes, evaluating (4.9) is intractable. In Section 4.4.4 we overcome this problem using a principled bound relaxation approach.

4.4.4 Considering a Subset of Possible Mode Sequences

Since it is intractable to evaluate every term in (4.9), we cannot use this bound in an optimization technique such as Sequential Quadratic Programming. Instead, we find a looser bound that is tractable. In this section, we describe an approach for finding the tightest possible such bound while evaluating a fixed number of the terms in (4.9). The key idea is to replace the terms that come from mode sequences that are *a priori* unlikely to contribute to the probability of pruning with a looser upper bound that does not depend on the control inputs. Since these terms do not depend on the control inputs, they do not need to be evaluated in the optimization.

4.4.5 A Looser Bound on the Probability of Pruning

In deriving the looser bound, we make use of Theorem 4.2, which we prove at the end of this section:

$$\frac{\left|\frac{\Sigma_i+\Sigma_j}{2}\right|}{\sqrt{|\Sigma_i||\Sigma_j|}} \geq 1, \tag{4.53}$$

where Σ_i and Σ_j are symmetric and positive definite. We now define:

$$F_{ij}(\mathbb{U}) \triangleq P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} e^{-k(i,j)}, \tag{4.54}$$

and write the bound on the probability of pruning (4.41) using this definition:

$$P(pruning) \leq \sum_i \sum_{j>i} F_{ij}(\mathbb{U}). \tag{4.55}$$

Following on from the result in (4.53), and the definition in (4.10), it is clear that $k(i,j) \geq 0$, and hence $e^{-k(i,j)} \leq 1$. This yields the following bound on F_{ij}:

$$G_{ij} \triangleq P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}} \geq F_{ij}(\mathbb{U}). \tag{4.56}$$

Note that the bound G_{ij} does not depend on the control inputs $\mathbb{U}$. By replacing $F_{ij}(\mathbb{U})$ terms in (4.55) with G_{ij} terms, we obtain a looser upper bound on the probability of pruning the true mode sequence:

$$p(prune) \leq \sum_{i\in S} \sum_{j>i, j\in S} F_{ij}(\mathbb{U}) + \sum_{i\notin S} \sum_{j>i, j\notin S} G_{ij}. \tag{4.57}$$

Here S is the set of s mode sequences for which the full bound is calculated as a function of $\mathbb{U}$. We would like the tightest such bound for a given size of S. We achieve this as follows.

The difference between the bound G_{ij} and the bound $F_{ij}(\mathbb{U})$ is largest when the control inputs $\mathbb{U}$ drive the value of $F_{ij}(\mathbb{U})$ to zero, at which point the difference is $P(H_i)^{\frac{1}{2}} P(H_j)^{\frac{1}{2}}$. Since we do not have knowledge of $\mathbb{U}$ when choosing which mode sequences to include in S, we assume this *worst* *case* difference between G_{ij} and F_{ij}. In order to find the tightest bound of the form (4.57), we therefore include in S the mode sequences that give the largest $P(H_i)^{\frac{1}{2}}P(H_j)^{\frac{1}{2}}$. We would like to find S to maximize:

$$L = \sum_{i \in S} \sum_{j>i, j \in S} p(H_i)^{\frac{1}{2}} p(H_j)^{\frac{1}{2}}. \tag{4.58}$$

Intuitively, this means that optimization will concentrate on reducing terms where the control inputs can have the greatest effect on the probability of pruning the true mode sequence, and will ignore terms that do not contribute significantly to the probability of pruning the true mode sequence. It can be seen that in order to maximize L, the set S must be chosen to contain the s hypotheses with the greatest prior probability $p(H_i)$; replacing any $p(H_i)$ value with a lower one can only reduce, or have no effect on, terms in the summation (4.58).

We have therefore shown how to derive a tractable upper bound on the probability of pruning the true mode sequence that involves a fixed number s of mode sequences for which the observation statistics (4.48) and (4.49) need to be calculated. By choosing the s most likely mode sequences, we achieve the tightest such bound.

Arithmetic-Geometric Mean Theorem for Symmetric Matrices

In this section we prove Theorem 4.2, which was used in deriving a looser upper bound on the probability of pruning in Section 4.4.4.[1] Given some positive integer n, let us define

$$I_n = \{1, 2, \dots, n\}. \tag{4.59}$$

Let us also define $\varepsilon_{i_1 i_2 \dots i_n}$ (where $i_1, i_2, \dots, i_n \in I_n$) to be the components of the totally antisymmetric tensor in n dimensions, where $\varepsilon_{1,2,\dots,n} = 1$. The determinant of a $n \times n$ matrix M, with components m_{ij}, is then given by

$$|\mathrm{M}| = \sum_{i_1, i_2, \dots, i_n \in I_n} \epsilon_{i_1, i_2, \dots, i_n} m_{1,i_1} m_{2,i_2} \cdots m_{n,i_n}. \tag{4.60}$$

We shall now prove the following result.

Lemma 4.1. *If* A, B *are real, positive-definite, symmetric,* $n \times n$ *matrices, then*

$$|\mathrm{A} + \mathrm{B}| \geq |\mathrm{A}| + |\mathrm{B}|. \tag{4.61}$$

holds for all $n > 0$.

Proof: It is sufficient to prove this statement when A is diagonal (with positive components), as we can always find a basis where this is true.

[1]This proof is due to Senthooran Rajamanoharan.

Let us denote the diagonal components of A by λ_i, and the components of B by b_{ij}. Then, using (4.60)

$$|\mathrm{A}+\mathrm{B}| = \sum_{i_1,i_2,\ldots,i_n \in I_n} \epsilon_{i_1,i_2,\ldots,i_n}(\lambda_1\delta_{1,i_1}+b_{1,i_1})(\lambda_2\delta_{2,i_2}+b_{2,i_2})\cdots(\lambda_n\delta_{n,i_n}+b_{n,i_n}) \tag{4.62}$$

where δ_{ij} is the Kronecker delta. If we expand the brackets, we can rewrite this expression in the following manner:

$$\begin{aligned} |\mathrm{A}+\mathrm{B}| &= \lambda_1\lambda_2\ldots\lambda_n + \sum_{i_1,i_2,\ldots,i_n\in I_n} \epsilon_{i_1,i_2,\ldots,i_n} b_{1,i_1}b_{2,i_2}\cdots b_{n,i_n} + \\ &\sum_{i_2,\ldots,i_n\in I_n-\{1\}} \lambda_1\epsilon_{1,i_2,\ldots,i_n} b_{2,i_2}b_{3,i_3}\cdots b_{n,i_n} + \\ &\sum_{i_1,i_3,\ldots,i_n\in I_n-\{2\}} \lambda_2\epsilon_{i_1,2,i_3,\ldots,i_n} b_{1,i_1}b_{3,i_3}\cdots b_{n,i_n} + \cdots + \\ &\sum_{i_1,\ldots,i_{n-1}\in I_n-\{n\}} \lambda_n\epsilon_{i_1,i_2,\ldots,i_{n-1},n} b_{1,i_1}b_{2,i_2}\cdots b_{n-1,i_{n-1}} + \\ &\sum_{i_3,\ldots,i_n\in I_n-\{1,2\}} \lambda_1\lambda_2\epsilon_{1,2,i_3,\ldots,i_n} b_{3,i_3}b_{4,i_4}\cdots b_{n,i_n} + \\ &\sum_{i_2,i_4,\ldots,i_n\in I_n-\{1,3\}} \lambda_1\lambda_3\epsilon_{1,i_2,3,i_4,\ldots,i_n} b_{2,i_2}b_{4,i_4}\cdots b_{n,i_n} + \cdots \end{aligned}$$

We can identify the first term with $|\mathrm{A}|$, and the second term with $|\mathrm{B}|$. After some thought, it becomes apparent that each of the remaining terms takes the form of a product of various combination of λ_i and some cofactor of B. Specifically, if we define the cofactor M_J, where $J \equiv \{j_1, j_2, \ldots\} \subset I_n$, to be the determinant of the matrix obtained by removing the $j_1, j_2, \ldots$-numbered columns and rows from B, then

$$\begin{aligned} |\mathrm{A}+\mathrm{B}| &= |\mathrm{A}|+|\mathrm{B}|+ \\ &\lambda_1\mathrm{M}_{\{1\}}+\lambda_2\mathrm{M}_{\{2\}}+\cdots+\lambda_n\mathrm{M}_{\{n\}}+ \\ &\lambda_1\lambda_2\mathrm{M}_{\{1,2\}}+\lambda_1\lambda_3\mathrm{M}_{\{1,3\}}+\cdots+ \\ &\lambda_1\lambda_2\lambda_3\mathrm{M}_{\{1,2,3\}}+\cdots \end{aligned} \tag{4.63}$$

As we have formed M_J by removing rows and columns symmetrically from B, each M_J must itself be the determinant of a positive-definite symmetric matrix, and hence

$$\mathrm{M}_J > 0 \;\; \forall \;\; J \subset I_n. \tag{4.64}$$

This, along with the fact that all λ_i are positive, means that (4.61) follows directly from (4.63). □

Theorem 4.2. *The arithmetic-geometric mean theorem. Any two symmetric positive-definite $n \times n$*

real matrices Σ_0, Σ_1 satisfy the following inequality:

$$\frac{\left|\frac{\Sigma_0+\Sigma_1}{2}\right|}{\sqrt{|\Sigma_0|\,|\Sigma_1|}} \geq 1. \tag{4.65}$$

Proof: Using Lemma 4.1 in the last line,

$$\begin{aligned} |\Sigma_0|\,|\Sigma_1| &= \left(\frac{|\Sigma_0|+|\Sigma_1|}{2}\right)^2 - \left(\frac{|\Sigma_0|-|\Sigma_1|}{2}\right)^2 \\ &\leq \left(\frac{|\Sigma_0|+|\Sigma_1|}{2}\right)^2 \\ &\leq \left|\frac{\Sigma_0+\Sigma_1}{2}\right|^2. \end{aligned} \tag{4.66}$$

□

4.4.6 Mode Sequence Enumeration as Best-First Search

Choosing the s most likely mode sequences is a challenging problem in itself given an exponential number of possible sequences. Prior work has, however, shown that this problem can be posed as a tree search problem[61]. This enables the best s mode sequences to be found efficiently using a best-first informed search approach[117]. We now describe how this can be applied to our problem. Figure 4-13 shows part of the search tree for the most likely mode sequence enumeration problem. The tree has K initial nodes that correspond to each of the mode sequences stored by the hybrid estimator at time step τ. In the diagram, $K = 2$. The graph search approach aims to find the s mode sequences $\mathbf{x}_{d,1:\tau+h}$ with the highest prior probability $p(\mathbf{x}_{d,1:\tau+h}|\mathbf{y}_{1:\tau})$. Making use of the Markov assumption inherent in JMLS, we can write this as:

$$p(\mathbf{x}_{d,1:\tau+h}|\mathbf{y}_{1:\tau}) = p(\mathbf{x}_{d,1:\tau}) \prod_{i=\tau+1}^{\tau+h} p(\mathbf{x}_{d,i}|\mathbf{x}_{d,i-1}) = p(\mathbf{x}_{d,1:\tau}) \prod_{i=\tau+1}^{\tau+h} [\mathcal{T}]_{\mathbf{x}_{d,i}\mathbf{x}_{d,i-1}}, \tag{4.67}$$

where $[\mathcal{T}]_{ab}$ is defined in (4.38). We can equivalently maximize the logarithm of this probability:

$$\log p(\mathbf{x}_{d,1:\tau+h}|\mathbf{y}_{1:\tau}) = \log p(\mathbf{x}_{d,1:\tau}|\mathbf{y}_{1:\tau}) + \sum_{i=\tau+1}^{\tau+h} \log[\mathcal{T}]_{\mathbf{x}_{d,i}\mathbf{x}_{d,i-1}}. \tag{4.68}$$

The cost of each initial node in the search tree is the probability of the corresponding mode sequence $\mathbf{x}_{d,1:\tau}(i)$ given the observations up to time step τ. This probability is provided by the hybrid estimator running up to time step τ. Each intermediate node in the graph corresponds to a partial assignment to the modes $\mathbf{x}_{d,1:\tau+h}$, and a goal node corresponds to a full assignment. The cost of a node is the sum of the arcs from the initial node to that node. We can therefore explore the graph using best-first search [117] to find the set of s goal nodes with the largest log-probability given in

(4.68). In this manner, the most likely mode sequences can be enumerated without evaluating the prior likelihood of each mode sequence, which is intractable.

4.4.7 Summary

The AE-JMLS algorithm is summarized in Table 4.3. The algorithm works in parallel with approximate hybrid estimation, which calculates the probability of each of the K tracked mode sequences. Starting from these sequences, AE-JMLS then enumerates the s most likely future mode sequences over the horizon $\tau+1,\ldots,\tau+h$ using best-first search. AE-JMLS forms an upper bound on the probability of approximate hybrid estimation losing the true mode sequence, which involves only the s most likely future sequences. AE-JMLS minimizes this upper bound subject to constraints on the expected system state using Sequential Quadratic Programming. This yields an optimized sequence of control inputs that is applied to the system, while hybrid estimation continues to estimate the hybrid state. Since control inputs are designed subject to hard constraints, we can ensure that:

1. A control task, defined in terms of the expected state, is achieved.
2. The expected system state remains within a linearization region.
3. Actuator saturation limits are not violated.

In this sense, the active hybrid estimation approach uses constraints to perform control, while optimizing with regard to estimation.

Function ACTIVEHYBRIDMAIN($\mathcal{M}$,$\mathbf{y}_{1:\tau}$,K,h) **returns** ($\mathbf{u}_{\tau:\tau+h-1}{}^*$)

1) Perform K-best hybrid estimation given the observation sequence $\mathbf{y}_{1:\tau}$. Calculate the probabilities of the K tracked mode sequences, as well as the distribution over the continuous state conditioned on each mode sequence.
2) Starting from the K tracked mode sequences, enumerate the s most likely future mode sequences over the horizon $\tau+1,\ldots,\tau+h$ using best-first search.
3) Forms the upper bound on the probability of pruning the true mode sequence involving only terms corresponding to the s most likely future mode sequences:

$$J = \sum_{i\in S}\sum_{j>i,j\in S} P(H_i)^{\frac{1}{2}}P(H_j)^{\frac{1}{2}}e^{-k(i,j)}. \qquad (4.69)$$

4) Using Sequential Quadratic Programming, minimize (4.69) subject to constraints on the expected system state and control inputs to find the optimal control sequence $\mathbf{u}_{\tau:\tau+h-1}{}^*$.
5) Execute optimal control sequence $\mathbf{u}_{\tau:\tau+h-1}{}^*$ while estimating hybrid state.

Table 4.3. AE-JMLS for Active Hybrid Estimation algorithm with JMLS.

4.5 Simulations Results: JMLS Discrimination

In this section we demonstrate the AE-JMLS algorithm in simulation. We consider the aircraft described in Section 4.3, except we now model the system as a JMLS with the following modes:

- Mode 1: Nominal (no faults)
- Mode 2: Faulty pitch rate sensor
- Mode 3: Faulty vertical velocity sensor
- Mode 4: Faulty elevator actuator

These modes are the same as the models described in Section 4.3. The key difference is that the JMLS model explicitly models stochastic jumps between the modes; stochastic jumps represent component failures or recoveries from failure. The transition probability matrix is:

$$\mathcal{T} = \begin{bmatrix} 0.97 & 0.01 & 0.01 & 0.01 \\ 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}. \tag{4.70}$$

Notice that once a fault occurs, it persists indefinitely. The initial belief state is uniform across the four discrete modes. As in Section 4.3.4 we constrain the expected system state, conditioned on nominal operation, to make the aircraft carry out an altitude change maneuver.

Given the designed control sequence we simulate the JMLS aircraft model and carry out approximate hybrid estimation as described in Section 4.4.1. Hybrid estimation is sequential, and at every time step mode sequences that are not in the 20 most likely are discarded. While in designing the control sequence we assume that sequences are discarded at the end of the planning horizon, we evaluate the new control design approach in terms of the benefits afforded to *sequential* hybrid estimation, since this is the most common implementation. The two metrics used are:

1. *Probability of discarding the true mode sequence.* This is estimated by carrying out a large number of simulations and recording occurrences of true mode sequence loss.
2. *Probability of Maximum A Posteriori (MAP) mode sequence estimation error.* Correctly estimating the MAP mode sequence is important for control, in particular. This value is again estimated through a large number of simulations.

The control design algorithm considered the *a priori* most likely 10 sequences in the simplification step described in Section 4.4.4. The elevator angle was constrained to have a maximum magnitude of $0.25rad$.

Figure 4-14 shows the designed active hybrid estimation control input, as well as the fuel-optimal sequence, for comparison. Active hybrid estimation yields a probability of losing the correct mode sequence of 0.0570 and a probability of MAP mode sequence error of 0.1540. Optimizing with respect to fuel yields a probability of losing the correct mode sequence of 0.0970 and a MAP mode sequence error probability of 0.3040. Again, a significant improvement in fault detection can be achieved by using control inputs designed for model discrimination, rather than those designed to optimize some other criterion and employing only passive hybrid estimation. This solution took $269.7s$ to generate, of which $1.02s$ was spent finding the *a priori* most likely 10 sequences, $76.3s$ was spent calculating the necessary covariances, and $192.4s$ was spent carrying out the nonlinear optimization.

4.6 Qualitative State Plan Constrained Active Estimation

In this section we extend the active estimation approach described in Section 4.4 to address the problem of execution of Qualitative State Plans defined in Section 2. We describe a novel approach, called AE-QSP , that enables the latitude in the Qualitative State Plan specification to be used for the purposes of active estimation.

In Chapter 3 we described how chance-constrained execution of QSPs can be approximated by sampling from the distributions of all uncertain variables and constraining the trajectories of the resulting samples, or 'particles'. The resulting constraints are highly non-convex for a number of reasons. First, constraints on the number of particles for which an activity fails are combinatorial in nature. Second, temporally-flexible constraints lead to integer constraints on the time of each event. Third, the feasible region for each activity may be nonconvex. For resource-optimal execution, we can use MILP to deal with this non-convexity since cost functions such as fuel and time can be expressed as piecewise-linear functions of the decision variables. In the case of active estimation, however, the cost function derived in Section 4.4 is highly nonlinear and multimodal. Piecewise linearization of this function is not a viable solution since it is defined over many dimensions. QSP-constrained active estimation is therefore a very challenging problem.

Our approach to addressing this problem is as follows. First we generate a feasible solution using the PC-QSP approach described in Section 3. Then we perform a convex tightening of the constraints imposed by the Qualitative State Plan around the feasible solution. Any solution within the new, convex constraints satisfies the original chance-constrained Qualitative State Plan, up to the particle approximation; this approximation becomes exact as the number of particle used tends to infinity. We perform a search within the convex constraints for a feasible solution that is optimal with regard to active estimation. The convexity of the feasible region means that Sequential Quadratic Programming can be used to solve this problem to local optimality. Furthermore, by starting the search at a feasible inital guess, we guarantee that the approach will return a solution that satisfies the Qualitative State Plan.

In Sections 4.6.1 and 4.6.2 we describe the approach in detail.

4.6.1 Convex Tightening of Mixed-Integer Linear Constraints

In this section we describe a simple approach for approximating Mixed-Integer Linear constraints as a set of convex linear constraints. We approximate a non-convex feasible set $\mathcal{F}$, described by the Mixed-Integer Linear constraints, as a convex feasible set $\mathcal{G}$, described by a conjunction of linear constraints. This convex constraint tightening must be performed so that the following two conditions hold:

1. *Satisfaction of convex constraints implies satisfaction of nonconvex constraints.* This is equivalent to the condition $\mathcal{G} \subseteq \mathcal{F}$.

2. *Feasibility of nonconvex constraints implies feasibility of convex constraints.* This is equivalent to $\mathcal{F} \neq \emptyset \implies \mathcal{G} \neq \emptyset$.

The importance of these conditions will be highlighted in Section 4.6.2. The following approach performs the necessary approximation:

1. *Find initial feasible solution.* Define a linear, or piecewise linear cost function f. Minimize f subject to the nonconvex feasible set $\mathcal{F}$ using MILP.[2] Any resulting feasible solution is a feasible assignment $\langle \mathbf{C}^{\text{feas}}, \mathbf{D}^{\text{feas}} \rangle$ to all continuous variables $\mathbf{C}$ and all binary variables $\mathbf{D}$. If no solution exists, stop.

2. *Fix all binary variables to* $\mathbf{D}^{\text{feas}}$. The mixed-integer linear constraints become a set of linear constraints. Choose $\mathcal{G}$ to be the convex feasible set defined by these constraints. Since a solution exists for $\mathbf{D} = \mathbf{D}^{\text{feas}}$, we have $\mathcal{G} \neq \emptyset$. Furthermore $\mathcal{G} \subseteq \mathcal{F}$ as required.

4.6.2 Convex Tightening Approach to QSP-Constrained Active Estimation

We now apply the convex constraint tightening approach described in Section 4.6.1 to the problem of QSP-constrained active estimation. This gives a novel algorithm for active estimation with QSPs, described in full in Table 4.4. The new approach is illustrated in Figure 4-17. Due to the two convex tightening properties described in Section 4.6.1, this approach guarantees that active estimation will return a solution that satisfies the chance-constrained Qualitative State Plan if one exists, subject to the particle approximation. In practical terms, the convex constraint tightening corresponds to fixing the following parameters in the solution:

[2]Minimization of f is not strictly necessary, since we are only concerned with generating a feasible solution.

1. The times of every event
2. The particles that succeed for each activity
3. The edge constraints satisfied for each successful particle at each time step

The convex constraint tightening means that we are not guaranteed to find the global optimum with regard to active estimation, since we have reduced the feasible region. Intuitively, by fixing variables such as the times of events, we are not allowing active estimation to make full use of the latitude in the Qualitative State Plan. However the nonlinearity of the active estimation cost function means that global optimality cannot be guaranteed in any case, and in Section 4.7 we show empirically, that a locally optimal solution can be highly effective.

```
   Function ACTIVEQSPMAIN($P$,$\mathcal{M}$,$N$) returns $\mathbf{u}_{\tau:\tau+h-1}$
1) Find initial feasible solution to execution of QSP. Define a linear or piecewise-linear cost function $F_{lin}$, which could be the same as the cost function $F$ defined in QSP $P$. Use the Particle Control approach (Table 3.1) with $N$ particles, to solve the single-stage, limited horizon robust execution problem (Def. 26) for QSP $P$. This approach is guaranteed to find a feasible solution $\langle \mathbf{C}^{\text{feas}}, \mathbf{D}^{\text{feas}} \rangle$ if one exists, where $\mathbf{C}$ is the set of all continuous decision variables and $\mathbf{D}$ is the set of all binary decision variables. If a feasible solution does not exist, stop.
2) Fix all binary variables to $\mathbf{D}^{\text{feas}}$. Denote the resulting set of linear constraints $\mathcal{L}$.
3) Optimize for active estimation within $\mathcal{L}$. Using Sequential Quadratic Programming, minimize $F_{ae}$ over $\mathbf{u}_{\tau:\tau+h-1}$, subject to $\mathbf{C} \in \mathcal{L}$. Here $F_{ae}$ is the cost function for active hybrid estimation defined in (4.69). Start the optimization at $\mathbf{C}^{\text{feas}}$ to ensure that the initial guess is feasible.
```

Table 4.4. Active Hybrid Estimation with Qualitative State Plan constraints

4.6.3 Summary

We have presented a novel algorithm for active hybrid estimation with chance-constrained qualitative state plans. This algorithm uses nonlinear optimization to minimize an upper bound on the probability of approximate hybrid state estimation pruning the true mode sequence. By performing a convex tightening of the constraints around a resource-optimal solution generated using particle control, we are able to make the optimization tractable and guarantee that the algorithm will return a feasible solution, if one exists.

4.7 Simulation Results: QSP-constrained Active Estimation

In this section we demonstrate the AE-QSP algorithm in simulation using an AUV scenario. We model the same MBARI AUV as in Section 3.9, again considering the longitudinal dynamics only.

Noisy observations of depth and pitch rate are available through the depth sensor and Inertial Navigation System respectively. In this section we augment the model derived in [86] to include stochastic failures. We model the AUV as a JMLS with 4 modes:

- *Mode 1: Nominal operation.* The dynamics are identical to those in [86].
- *Mode 2: Pitch rate sensor failure.* Pitch rate readings are zero-mean Gaussian white noise.
- *Mode 3: Depth sensor failure.* Depth readings are zero-mean Gaussian white noise.
- *Mode 4: Tailcone actuator failure.* The tailcone actuator exerts no control influence.

The transition probability matrix for the AUV is:

$$T = \begin{bmatrix} 0.97 & 0.01 & 0.01 & 0.01 \\ 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}. \tag{4.71}$$

Notice that once a fault occurs, it persists indefinitely. The initial belief state is uniform across the four discrete modes. We model depth disturbances as zero-mean Gaussian white noise with standard deviation of $0.5m$. We assume that the noise on the depth observations and pitch rate observations are uncorrelated and are both zero-mean Gaussian white noise with standard deviation 0.1.

The AUV must execute the mission described by the Qualitative State Plan in Figure 4-18, while performing Active Hybrid Estimation.[3] The AE-QSP algorithm works by first generating a fuel-optimal solution to the robust execution problem using the PC-QSP algorithm. This solution is shown in Figure 4-19. AE-QSP takes this solution and fixes all binary variables to yield a convex feasible region. This means that the schedule is fixed, and the particles that succeed for each chance constraint are fixed. Then AE-QSP searches for a solution within this convex feasible region that minimizes the upper bound on the probability of approximate hybrid estimation losing the true mode sequence. The resulting plan is shown in Figure 4-20 for a typical case. This particular solution took $47.8s$ to generate, of which $0.12s$ was spent finding the most likely 5 sequences, $13.5s$ was spent calculating the necessary covariances and $34.1s$ was spent performing the nonlinear optimization. Note that the AE-QSP solution satisfies the chance-constrained Qualitative State Plan, but performs doublet maneuvers at $t = 100s$ and $t = 170s$ in order to disambiguate the hybrid state. The AE-QSP solution yields a probability of losing the true mode sequence of 0.042, whereas the fuel-optimal sequence has a probability of losing the true mode sequence of 0.10.

[3]Note that in this case, we only require the AUV to satisfy the state constraints if its actuator is functional; in this scenario satisfying the state constraints with a broken actuator is not possible.

4.8 Conclusion

This chapter introduced a novel method for active state estimation in Jump Markov Linear Systems. The method designs finite sequences of control inputs that reduce the probability of pruning the true mode sequence while ensuring that a given control task is achieved. We first presented a novel method for constrained optimal discrimination between a finite number of linear dynamics systems, and give simulation results that show that the probability of model selection error is dramatically decreased by the new method. By extending this approach we then derived a tractable active hybrid estimation method for chance-constrained Qualitative State Plans. Simulation results showed that the new method significantly reduces the probability of hybrid estimation losing the true mode sequence.

4.9 Discussion

One of the key assumptions used in creating the new approaches to model discrimination and active hybrid estimation is that model selection and mode sequence pruning are carried out at the end of the finite horizon control sequence, rather than at every time step. However in most Multiple Model and hybrid estimation schemes, selection occurs at every time step. While it would be possible, and in fact simpler, to formulate the control design problem for one time step consistent with such a scheme, we did not do so for two reasons. First, in the discrete-time formulation, in many systems the effect of control inputs at time step τ is not manifested in the observations until some time step after $\tau + 1$. Hence a design approach that only takes into account how the inputs at time step τ will affect the observations at $\tau+1$ will be severely limited. Second, by constraining the system state over a long horizon, the optimization has much greater latitude in designing a powerful control sequence for the purposes of estimation; the system state can be driven far from its initial value, while being brought back to its goal value by the end of the time horizon. Hence by considering the probability of pruning over a horizon, rather than over a single time step, the approach yields sequences that are more powerful with regard to discrimination. Furthermore, in Sections 4.3 and 4.5 we provided empirical results that show that the new methods do indeed reduce the probability of error when a sequential estimation method is employed.

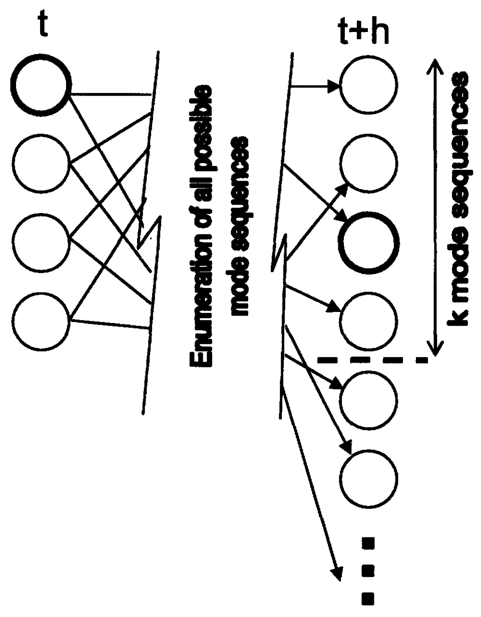


Figure 4-12. Pruning approach to approximate hybrid estimation. At time step τ, the estimator is tracking $K = 4$ distinct mode sequences. We assume that at time step $\tau+h$, the posterior probabilities of all possible mode sequences are calculated. The top K sequences are retained, while the remaining sequences are pruned. The true mode sequence is shown in bold; in this case it is not pruned.

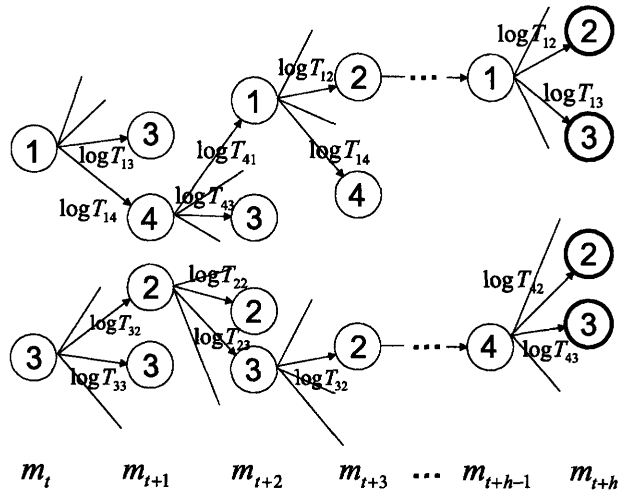


Figure 4-13. Search tree for enumeration of *a priori* most likely mode sequences. At the far left there are K nodes corresponding to the mode sequences tracked by the hybrid estimator at time step τ. Nodes at the far right correspond to a full assignment to the modes $\mathbf{x}_{d,1,\tau+h}$. Each arc value is the logarithm of the transition probability, and the value of a node is found by summing the value of the arcs leading to the node. Nodes are expanded in best-first order until s goal nodes are expanded. In this manner, the most likely mode sequences can be enumerated without evaluating the prior likelihood of each mode sequence, which is intractable. In this diagram $K = 2$ and $s = 4$.

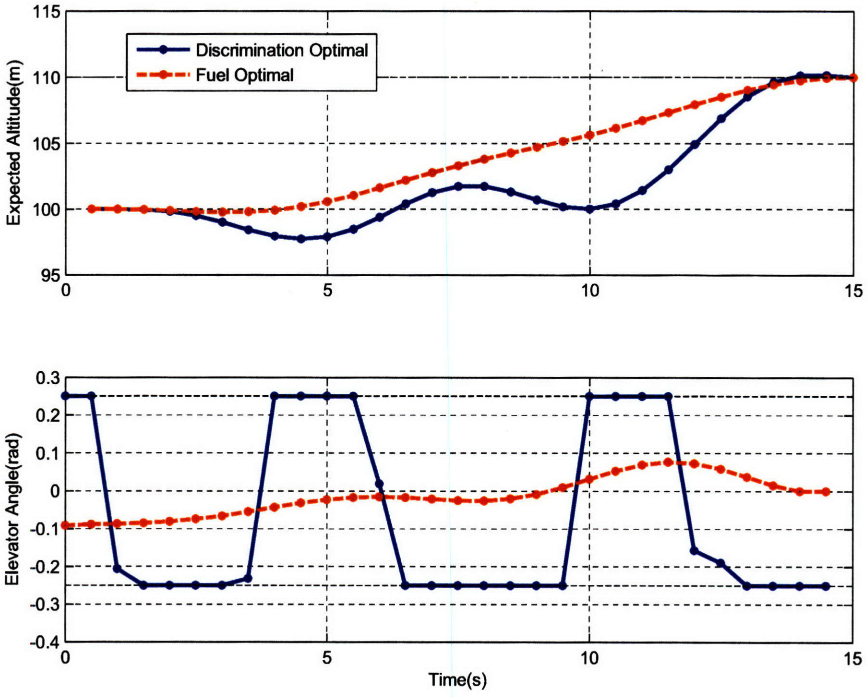


Figure 4-14. Active hybrid estimation for altitude change maneuver with JMLS aircraft model. The active estimation sequence yields a probability of losing the correct mode sequence of 0.0570 and a probability of MAP mode sequence error of 0.1540. Optimizing with respect to fuel yields a probability of losing the correct mode sequence of 0.0970 and a MAP mode sequence error probability of 0.3040.

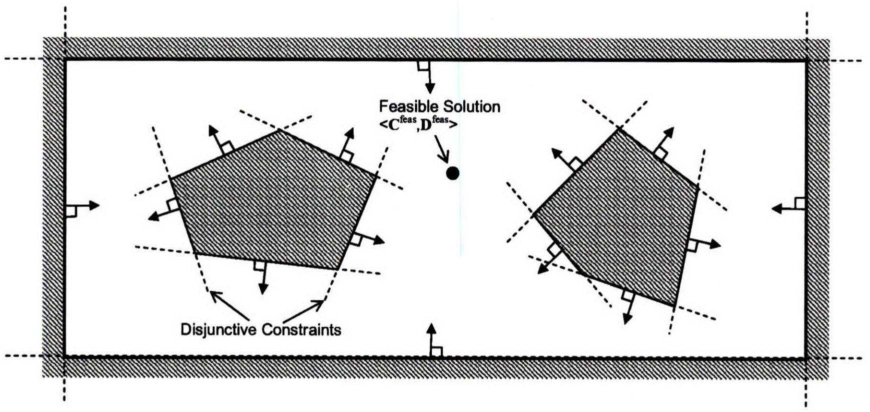


Figure 4-15. Nonconvex polygonal feasible region. Nonconvexity of the feasible region leads to disjunctive edge constraints. For each obstacle, at least one of the edge constraints must be satisfied. In a MILP, this disjunction is encoded using binary variables. Each binary indicates whether a given edge constraint is satified. Any feasible solution, such as $\langle \mathbf{C}^{\text{feas}}, \mathbf{D}^{\text{feas}} \rangle$ has a full assignment to the binary variables.

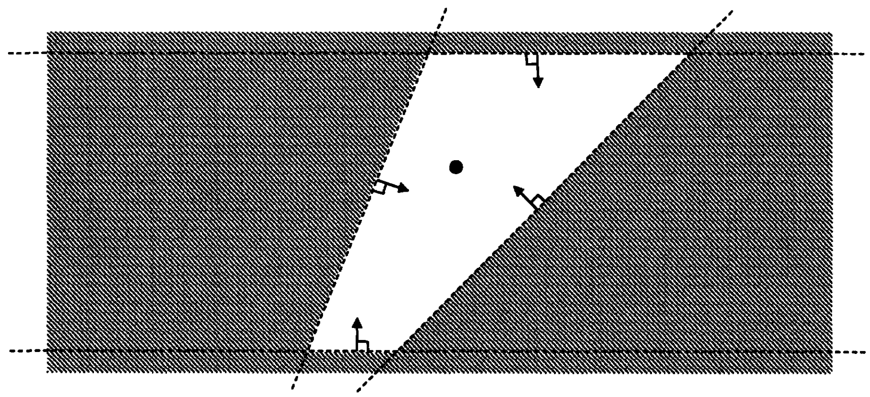


Figure 4-16. Convex tightening of nonconvex feasible region. Assigning all of the binary variables yields a convex feasible region that is a subset of the original nonconvex region. Using the binary variables from the feasible solution $\langle C^{feas}, D^{feas} \rangle$ ensures that the convex region is not empty.

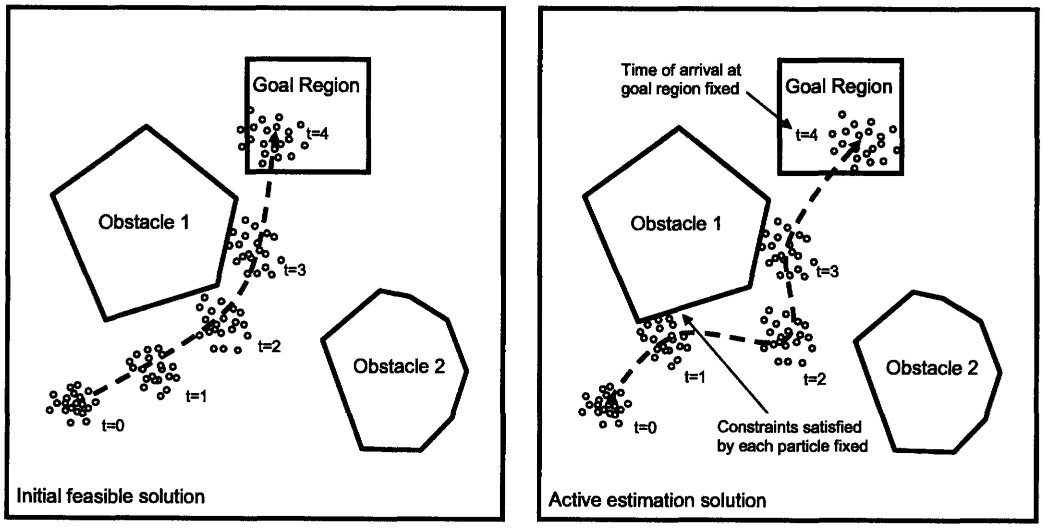
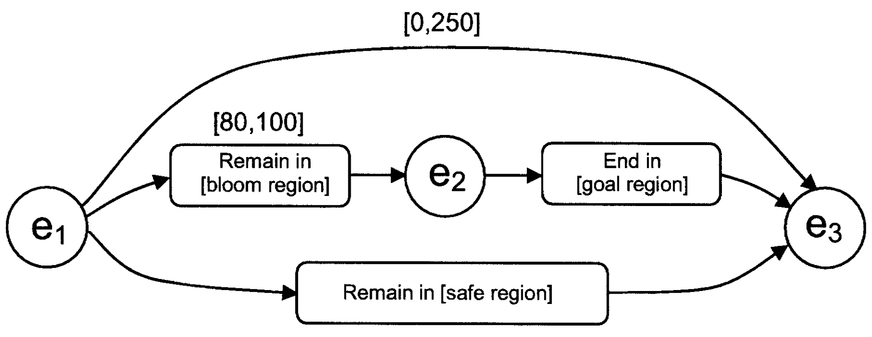

Goal Region
t=4
Obstacle 1
t=3
t=2
Obstacle 2
t=1
t=0
Initial feasible solution
Time of arrival at goal region fixed
Goal Region
t=4
Obstacle 1
t=3
t=2
t=1
Obstacle 2
t=0
Constraints satisfied by each particle fixed
Active estimation solution

Figure 4-17. Illustration of active estimation with chance-constrained qualitative state plans. **Left:** Initial feasible chance-constrained solution to QSP execution. In this case particle control has been used to find the fuel-optimal solution, in which exactly 10% of the particles fail. **Right:** Active estimation solution, optimized with regard to the probability of estimator error. The approximated chance constraints are satisfied, since fewer than 10% of the particles fail. The times of events, particles that fail, and the constraints satisfied by each successful particle are the same as for the initial solution. This introduces suboptimality in the active estimation solution.



$$p(\text{End in [goal region]}\,\mathit{fails}\ \text{OR}\ \text{Remain in [safe region]}\,\mathit{fails}) < 0.01$$

$$p(\text{Remain in [bloom region]}\,\mathit{fails}) < 0.1$$

Figure 4-18. Qualitative State Plan for AUV science mission. The AUV must remain in the bloom region before reaching the goal region, in order to rendezvous with the surface vessel.

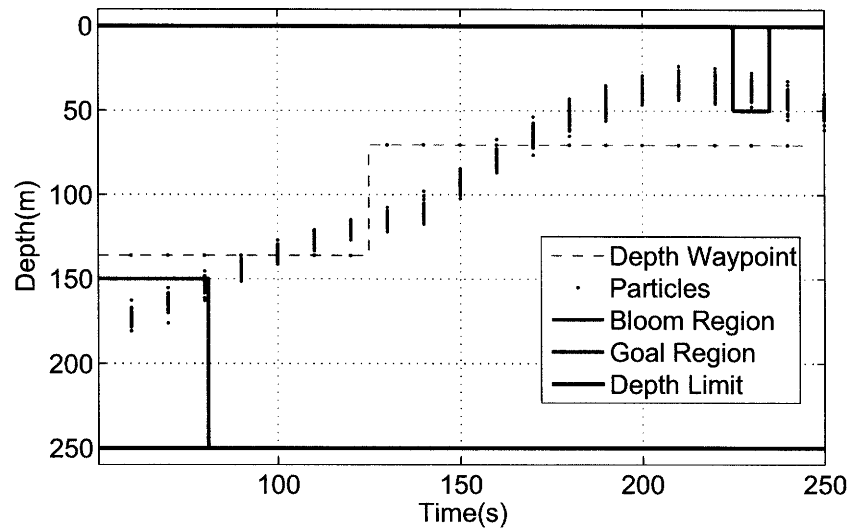


Figure 4-19. Typical fuel-optimal solution generated by PC-QSP algorithm using 50 particles and a horizon of 20 time steps. This solution robustly satisfies the Qualitative State Plan and minimizes fuel use. The AE-QSP algorithm uses this solution as a starting point, searching for an estimation-optimal solution in a convex region around the fuel-optimal solution.

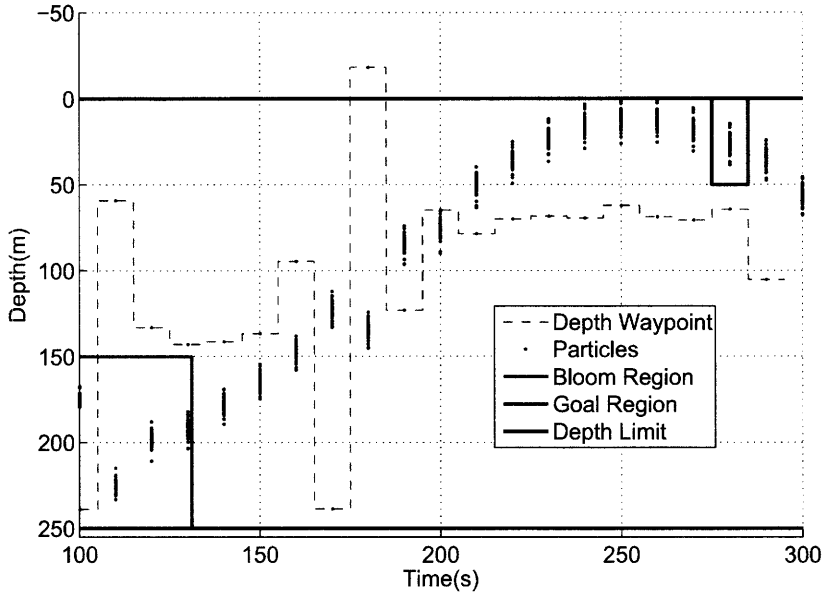


Figure 4-20. Typical estimation-optimal solution generated by AE-QSP algorithm using 50 particles and a horizon of 20 time steps. The solution robustly satisfies the Qualitative State Plan while minimizing an upper bound on the probability of approximate hybrid estimation losing the true mode sequence.

Chapter 5

Hybrid Model Learning

In this chapter we introduce a novel approach for model learning in hybrid discrete-continuous systems[1]. The hybrid model learning capability enables the hybrid executive to adapt its system model according to observed data to account for changing system characteristics and inaccurate model specifications. Since the control and estimation capabilities of the hybrid executive rely on accurate system models, a model learning capability is essential in ensuring that the executive operates as intended in a safe and optimal manner.

Model learning for hybrid discrete-continuous systems is challenging because of the coupling between the discrete state and the continuous dynamics. Both the discrete state and the continuous state are unobservable, however the continuous dynamics that we are trying to learn depend on the discrete state. Hence learning probabilistic hybrid models is significantly more challenging than learning purely discrete or purely continuous dynamic models.

In order to overcome these challenges we use an approximate Expectation-Maximization approach to find the hybrid model estimate. We estimate the hidden state given existing hybrid estimation capabilities, then learn the hybrid system dynamics using this estimate. We show empirically that iterating this process yields convergence to a model that is locally optimal in a Maximum Likelihood (ML) sense.

The contributions described in this chapter apply to a restricted class of Probabilistic Hybrid Automata that have linear continuous dynamics in each mode. In Section 5.1 we review related work. In Section 5.2 we define a Linear Probabilistic Hybrid Automaton and state the hybrid model learning problem. In Section 5.3 we review general Expectation-Maximization [34], and in Section 5.4 we give an overview of the new hybrid model learning approach before describing its key components in Sections 5.5 and 5.6. Finally, in Section 5.7 we provide simulation results.

[1]The theoretical developments in this chapter were carried out in conjunction with Seung Chung, while the experimental validation was carried out in conjunction with Stephanie Gil.

5.1 Related Work

In the case of linear systems with only continuous dynamics, previous work ([47, 27]) developed methods to determine the Maximum-Likelihood (ML) model parameters using an Expectation-Maximization(EM)[34] approach. This approach guarantees convergence to a local maximum of the likelihood function. More recent work extended this approach to Jump Markov Linear Systems(JMLS)[48, 9]. These systems have linear continuous dynamics and Markovian discrete transitions; in this case the discrete dynamics are independent of the continuous state. [48] used an 'approximate' EM approach to learn the parameters of JMLS. Due to the approximation introduced in the Expectation Step (E-Step), this approach does *not* have guaranteed convergence. [9] used an approach inspired by, but not the same as, Expectation Maximization; this approach iterates between calculating the maximum likelihood discrete model and the maximum likelihood continuous model. This is analogous to 'hard' EM, where instead of using distributions over the unknown variables, only the most likely values are used. This approach does have a guarantee of convergence to a local maximum of the likelihood function.

In this chapter we present a new approach to hybrid model learning that extends the prior work in three ways. First, we generalize hybrid model learning to LPHA, where transitions in the discrete state *do* depend on the continuous state. Second, we extend the work of [48, 9] to learn explicitly the dependence of control inputs on the system dynamics. Finally, we consider 'soft' EM, where a distribution over the hidden variables is used, rather than just the most likely values. The new approach presented here is inspired by the work of [59], which suggested the application of EM of to PHA, but did not provide the necessary mathematical basis for doing so.

5.2 Problem Statement

In this section we are concerned with a restricted type of Probabilistic Hybrid Automaton, which we refer to as a Linear Probabilistic Hybrid Automaton (LPHA). The continuous dynamics for a LPHA are given by:

$$\begin{aligned} \mathbf{x}_{c,\tau+1} &= A(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau} + B(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \omega_\tau \\ \mathbf{y}_{\tau+1} &= C(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau+1} + D(\mathbf{x}_{d,\tau})\mathbf{u}_\tau + \nu_\tau, \end{aligned} \tag{5.1}$$

where $\mathbf{x}_c \in \Re^{n_x}$ is the continuous state and $\mathbf{x}_d \in \mathcal{X}_d$ is the discrete state, or *mode*. We use the subscript notation $\mathbf{x}_{c,\tau}$ to denote the value of variable $\mathbf{x}_c$ at time t_τ, and use $\mathbf{x}_{c,\tau_1:\tau_2}$ to denote the sequence $\mathbf{x}_{c,\tau_1}, \ldots, \mathbf{x}_{c,\tau_2}$. The variables ω and ν are process and observation noise respectively, which we restrict to be zero-mean, Gaussian white noise with covariance $Q(\mathbf{x}_{d,\tau})$ and $R(\mathbf{x}_{d,\tau})$, respectively. The initial distribution $p(\mathbf{x}_{c,0}, \mathbf{x}_{d,0})$ is a sum-of-Gaussians, where $p(\mathbf{x}_{c,0}|\mathbf{x}_{d,0})$ is a Gaussian with

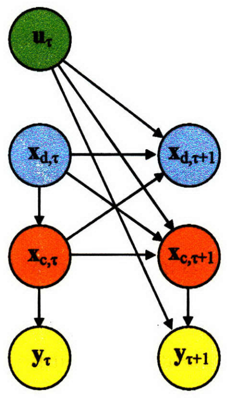


Figure 5-1. Bayes Network showing conditional dependencies between variables at time t and $t+1$ in a Linear Probabilistic Hybrid Automaton. Note that the discrete state *does* depend on the input and the continuous state, as in a general PHA.

mean $\mu(\mathbf{x}_{d,0})$ and covariance $V(\mathbf{x}_{d,0})$.

The evolution of the discrete state $\mathbf{x}_{d,\tau}$ is the same as defined for PHA in Definition 2, which we repeat here. The probabilistic discrete evolution of the automaton is specified through a finite set of *transition specifications* $\mathcal{C}_i \triangleq \langle g_i, \mathcal{T}_i \rangle$. Each *transition specification* $\mathcal{C}_i$ has an associated Boolean *guard* $g_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathcal{U}_d \to \{true, false\}$ and a *transition probability matrix* $\mathcal{T}_i \in \mathbb{R}^{|\mathcal{X}_d| \times |\mathcal{X}_d|}$. Each matrix $\mathcal{T}_i$ defines the probability of a transition from any mode to any other mode given that guard condition i is satisfied, such that $[\mathcal{T}_i]_{m,n} = p(\mathbf{x}_{d,\tau+1} = m | \mathbf{x}_{d,\tau} = n, g_i = \text{true})$. Here $[\mathcal{T}_i]_{m,n}$ denotes the (m, n)'th element of the matrix $\mathcal{T}_i$. The elements of each column in $\mathcal{T}_i$ must sum to 1. For clarity we consider guard regions over the continuous state only, however dealing with guards defined over control inputs or output variables in the learning problem is straightforward since these are fully observable. The Bayes Network representation of a LPHA is shown in Figure 5-1.

Definition 30. *The* continuous parameters $\theta_c(\mathbf{x}_d)$ *are defined for each mode* $\mathbf{x}_d \in \mathcal{X}_d$ *of a LPHA* $\mathcal{P}$ *as the set* $\langle A(\mathbf{x}_d), B(\mathbf{x}_d), C(\mathbf{x}_d), D(\mathbf{x}_d), Q(\mathbf{x}_d), R(\mathbf{x}_d), V(\mathbf{x}_d), \mu(\mathbf{x}_d) \rangle$. *Given a known input sequence* $\mathbf{u}_{c,1:f}$, *a known discrete mode sequence* $\mathbf{x}_{d,1:f}$ *and parameters* $\theta_c(x_d)$ *for each* $\mathbf{x}_{d,t}$ *that appears in the sequence* $\mathbf{x}_{d,1:f}$, *the stochastic dynamics of the continuous state* $\mathbf{x}_c$ *and observation* $\mathbf{y}$ *are completely defined.*

Definition 31. *The* discrete parameters θ_d *are defined as the set of transition probability matrices* $\mathcal{C}_i$ *and the initial probability distribution* $p(\mathbf{x}_{d,0} = \mathbf{x}_d)$.

Note that the discrete parameters do not include the guard conditions c_i themselves; learning of guard regions is a topic for future research.

Definition 32. *The* hybrid model parameters *for a LPHA* $\mathcal{P}$ *are defined as the union of the continuous parameters for all modes and the discrete parameters, i.e.* $\theta = \langle \bigcup_{\mathbf{x}_d \in \mathcal{X}_d} \theta_c(\mathbf{x}_d), \theta_d \rangle$, *where* $\theta_c(\mathbf{x}_d)$ *is given in Definition 30 and* θ_d *is given in Definition 31. The hybrid model parameters uniquely define the hybrid discrete-continuous stochastic dynamics of* $\mathcal{P}$.

We now define the problem of hybrid model learning for LPHA.

Definition 33. *Given a finite sequence of observations* $\mathbf{y}_{1:f+1}$ *and a finite sequence of control inputs* $\mathbf{u}_{c,0:f}$*, the* hybrid model learning problem for LPHA *is to determine the hybrid model parameters* θ *that maximize the likelihood* $p(\mathbf{y}_{1:f+1}|\theta)$.

5.3 Review of Expectation-Maximization

In this section we review the general Expectation-Maximization (EM) algorithm[34]. More thorough reviews have been carried out by [91] and [29].

EM is an iterative approach for finding a local maximum to a function, which is often the Maximum Likelihood probability of some general vector of observations $\mathbb{Y}$ given a vector of parameters θ:

$$f(\theta) = p(\mathbb{Y}|\theta). \tag{5.2}$$

The maximizer of this function is also the maximizer of the function:

$$g(\theta) = \log p(\mathbb{Y}|\theta). \tag{5.3}$$

EM addresses the situation when the likelihood value $p(\mathbb{Y}|\theta)$ is not readily evaluated. An example of this is when there are *hidden* variables so that only the distribution $p(\mathbb{Y}, \mathbb{X}|\theta)$ is known explicitly. In this case we can express the likelihood using a marginalization over the hidden variables and rewrite (5.3) as:

$$g(\theta) = \log \int_{\mathbb{X}} p(\mathbb{Y}, \mathbb{X}|\theta) d\mathbb{X} \tag{5.4}$$

In this case $\mathbb{X}$ takes continuous values; in the case of discrete-valued $\mathbb{X}$, the integral is replaced by a summation. The key difficulty in maximizing $g(\theta)$ is that it involves a logarithm over an integral (or a large summation), which is difficult to deal with in many cases[29]. It is, however, possible to create a lower bound to $g(\theta)$ that instead involves an integral or sum of logarithms, which *is* tractable. In EM, Jensen's inequality is used to give the lower bound:

$$g(\theta) = \log \int_{\mathbb{X}} p(\mathbb{Y}, \mathbb{X}|\theta) d\mathbb{X} \geq \int_{\mathbb{X}} p(\mathbb{X}|\mathbb{Y}, \theta^k) \log \frac{p(\mathbb{Y}, \mathbb{X}|\theta)}{p(\mathbb{X}|\mathbb{Y}, \theta^k)} d\mathbb{X} \triangleq h(\theta|\theta^k), \tag{5.5}$$

where θ^k is a guess for the value for the parameters θ at iteration k of the EM algorithm. This bound can be written in terms of an expectation over the hidden state X, and an 'entropy' term denoted $\mathcal{H}$ that does not depend on θ.

$$\begin{aligned} h(\theta|\theta^k) &= \int_{\mathbb{X}} p(\mathbb{X}|\mathbb{Y},\theta^k)\log\frac{p(\mathbb{Y},\mathbb{X}|\theta)}{p(\mathbb{X}|\mathbb{Y},\theta^k)}d\mathbb{X} \\ &= \int_{\mathbb{X}} p(\mathbb{X}|\mathbb{Y},\theta^k)\log p(\mathbb{Y},\mathbb{X}|\theta)d\mathbb{X} - \int_{\mathbb{X}} p(\mathbb{X}|\mathbb{Y},\theta^k)\log p(\mathbb{X}|\mathbb{Y},\theta^k)d\mathbb{X} \\ &= E_{\mathbb{X}|\mathbb{Y},\theta^k}\left[\log p(\mathbb{Y},\mathbb{X}|\theta)\right] + \mathcal{H}. \end{aligned} \tag{5.6}$$

The lower bound $h(\theta|\theta^k)$ is the tightest possible bound[91], and in particular at the current guess θ^k, the bound touches the objective function $g(\theta)$.

The key idea behind the EM algorithm is to construct the tractable bound $h(\theta|\theta^k)$ given the current guess θ^k and then to maximize the bound with respect to θ to give θ^{k+1}. The bound $h(\theta|\theta^k)$ involves an integral over logarithms rather than the logarithm of an integral, which makes it tractable for optimization in many cases. Because the bound $h(\theta|\theta^k)$ touches the objective function $g(\theta)$ at the current guess, it is guaranteed that maximizing the bound with regard to θ finds a value for θ that increases the true objective function $g(\theta)$ unless θ^k is a local maximizer of $g(\theta)$; in this case the local optimum has been found[91]. EM therefore proceeds as follows:

1. **Initialization:** Set $k = 1$. Initialize θ^k to be an initial guess for θ.

2. **Expectation Step:** Given θ^k, calculate the bound $h(\theta|\theta^k)$.

3. **Maximization Step:** Set θ^{k+1} to be the value of θ that maximizes the bound $h(\theta|\theta^k)$.

4. **Convergence Check:** Evaluate $g(\theta^{k+1})$. If the value of $g(\theta)$ has converged, stop. Otherwise set $k = k + 1$ and go to Step 2.

For each k we have $g(\theta^{k+1}) \geq g(\theta^k)$, with equality only if θ^k is a local maximizer. Hence EM guarantees that θ^k will converge to a local maximizer of $g(\theta)$, and does so by iteratively maximizing a *tractable* lower bound on $g(\theta)$. This is illustrated in Figure 5-2. In order for this convergence property to apply, it is necessary that the bound $h(\theta|\theta^k)$ touches the objective function. Previous work has shown that in special cases, this requirement can be relaxed[95], however in the general case the bound must touch the objective function. This is illustrated in Figure 5-3.

5.4 Overview of Approach

In this section we outline the new approach to hybrid model learning for LPHA. The new approach uses EM as described in Section 5.3, except for the key difference that the Expectation Step is approximated in order to ensure tractability.

In order to solve the hybrid model learning problem defined in Section 5.2, we must maximize the value $f(\theta) = p(\mathbf{y}_{1:f+1}|\theta)$. Using EM to do so as described in Section 5.3 we first calculate

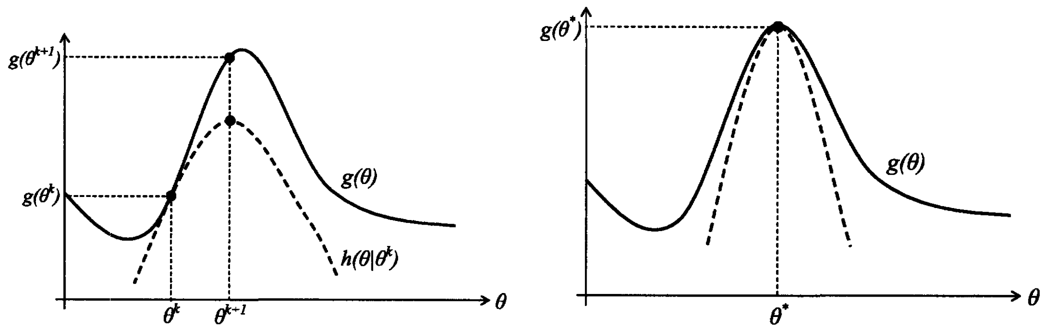


Figure 5-2. Expectation-Maximization finds a local maximum of the function $g(\theta)$ through iterative lower bound maximization. **Left:** At each iteration a lower bound $h(\theta|\theta^k)$ is constructed, which touches $g(\theta)$ at the current guess θ^k. Any value of θ^{k+1} that increases $h(\theta|\theta^k)$ must also increase $g(\theta)$, as shown. **Right:** EM converges when θ^k is at a local maximum of $h(\theta|\theta^k)$ denoted θ^*, which, under smoothness assumptions, is guaranteed to be a local maximum of $g(\theta)$.

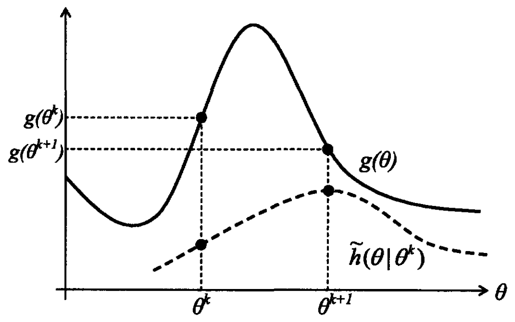


Figure 5-3. Convergence is not guaranteed if the lower bound does not touch the objective function. In this figure the 'approximate' lower bound $\tilde{h}(\theta|\theta^k)$ is used instead of $h(\theta|\theta^k)$. Since the approximate bound does not touch $g(\theta)$, we cannot guarantee that successive iterations will increase $g(\theta)$.

the bound $h(\theta|\theta^k)$ (Expectation Step) and then maximize the bound (Maximization Step). In the case of hybrid model learning, the hidden data X is comprised of both the hidden continuous state sequence $\mathbf{x}_{c,0:f+1}$ and the hidden discrete mode sequence $\mathbf{x}_{d,0:f}$. The observed data consists of the observation sequence $\mathbf{y}_{1:f+1}$. The bound is therefore:

$$h(\theta|\theta^k) = E\big[\log p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta)\big] + \mathcal{H}, \tag{5.7}$$

where the expectation is calculated over the hybrid state distribution $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$. Calculating this expectation is not, in the general case, possible in closed form. However in Section 5.5 we show how the structure of LPHA can be exploited to make this possible. In doing so we enable existing results for Linear Time Invariant(LTI) systems to be used in the Maximization step, as described in Section 5.6. These results make it possible for the maximum of the bound (5.7) to be found analytically. Both the Expectation Step and the Maximization step are intractable in practice, however, because the number of mode sequences $\mathbf{x}_{d,0:f}$ is exponential in the number of time steps f. In Section 5.5 we therefore introduce an approximation to the bound $h(\theta|\theta^k)$ that makes

```
    Function HybridLearningMain($\theta^1$,$\mathbf{y}_{1:f+1}$) returns $\theta$
1)  Initialization. Set $k = 1$ and initialize $\theta^k$ to be the initial guess for $\theta$.
2)  Approximate Expectation Step. Given $\theta^k$ and the sequence of observations $\mathbf{y}_{1:f+1}$ determine the approximate bound $\tilde{h}(\theta|\theta^k)$ using the approach in Section 5.5.
3)  Maximization Step. Set $\theta^{k+1}$ to be the value of $\theta$ that maximizes the bound $\tilde{h}(\theta|\theta^k)$ using the approach in Section 5.6.
4)  Convergence Check. If the value of $\theta$ has converged, stop. Otherwise set $k = k + 1$ and go to Step 2.
```

Table 5.1. Tractable Expectation-Maximization approach to hybrid model learning with LPHA.

the Expectation and Maximization Steps tractable. The tractable approximate EM algorithm for LPHA is shown in Table 5.1.

5.5 Expectation Step for Hybrid Model Learning

In order to calculate the bound (5.7) analytically we first use the law of iterated expectations to write the bound in terms of an expectation over the continuous state, conditioned on the discrete mode sequence, and an expectation over discrete mode sequences.

$$h(\theta|\theta^k) = E_{\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k}\left[E_{\mathbf{x}_{c,0:f+1}|\mathbf{x}_{d,0:f},\mathbf{y}_{1:f+1},\theta^k}\left[\log p(\mathbf{y}_{1:f+1},\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\theta)\right]\right] + \mathcal{H}. \quad (5.8)$$

Writing the expectations out in full gives:

$$h(\theta|\theta^k) = \mathcal{H} + \sum_{\mathbf{x}_{d,0:f}}\Bigg(p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)* \\ \int p(\mathbf{x}_{c,0:f+1}|\mathbf{x}_{d,0:f},\mathbf{y}_{1:f+1},\theta^k)\log p(\mathbf{y}_{1:f+1},\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\theta)d\mathbf{x}_{c,0:f+1}\Bigg). \quad (5.9)$$

In order to construct this bound we require two key values. The first value is $p(\mathbf{y}_{1:f+1},\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\theta)$, also known as the 'completed-data' probability. This describes the joint distribution over observation sequences and state sequences given the model parameters. The second value is $p(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)$, which is the distribution over the hidden hybrid state *given* both the observed observation sequence and the *current guess* for the model parameters. The calculation of these distributions is described in Sections 5.5.1 and 5.5.2.

5.5.1 Completed-Data Probability

In order to calculate the complete-data probability $p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta)$, we first use the Markov properties of LPHA to write the probability as:

$$p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta) =$$
$$p(\mathbf{x}_{c,0}, \mathbf{x}_{d,0}|\theta) \prod_{\tau=0}^{f-1} p(\mathbf{x}_{d,\tau+1}|\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}, \theta) \prod_{\tau=0}^{f} p(\mathbf{y}_{\tau+1}|\mathbf{x}_{c,\tau+1}, \mathbf{x}_{d,\tau}, \theta) p(\mathbf{x}_{c,\tau+1}|\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}, \theta). \quad (5.10)$$

We can calculate the individual terms in (5.10) by extending standard results from LTI systems[27] to switching systems. Given the definition of the process noise ω, we have:

$$p(\mathbf{x}_{c,\tau+1}|\mathbf{x}_{c,\tau}, \mathbf{x}_{d,\tau}, \theta) = (2\pi)^{-\frac{n_x}{2}} |Q(\mathbf{x}_{d,\tau})|^{-\frac{1}{2}} e^{-\frac{1}{2}\delta_p' Q^{-1}(\mathbf{x}_{d,\tau})\delta_p}$$
$$\delta_p = \mathbf{x}_{c,\tau+1} - A(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau} - B(\mathbf{x}_{d,\tau})\mathbf{u}_\tau. \quad (5.11)$$

Similarly, given the definition of the observation noise ν, we have:

$$p(\mathbf{y}_{\tau+1}|\mathbf{x}_{c,\tau+1}, \mathbf{x}_{d,\tau}, \theta) = (2\pi)^{-\frac{n_y}{2}} |R(\mathbf{x}_{d,\tau})|^{-\frac{1}{2}} e^{-\frac{1}{2}\delta_o' R^{-1}(\mathbf{x}_{d,\tau})\delta_o}$$
$$\delta_o = \mathbf{y}_{\tau+1} - C(\mathbf{x}_{d,\tau})\mathbf{x}_{c,\tau+1} - D(\mathbf{x}_{d,\tau})\mathbf{u}_\tau. \quad (5.12)$$

The initial probability distribution $p(\mathbf{x}_{c,0}, \mathbf{x}_{d,0})$ is given by:

$$p(\mathbf{x}_{c,0}, \mathbf{x}_{d,0}|\theta) = p(\mathbf{x}_{d,0})(2\pi)^{-\frac{n_x}{2}} |V(\mathbf{x}_{d,0})|^{-\frac{1}{2}} e^{-\frac{1}{2}\left[\mathbf{x}_{c,0}-\mu(\mathbf{x}_{d,0})\right]' V^{-1}(\mathbf{x}_{d,0})\left[\mathbf{x}_{c,0}-\mu(\mathbf{x}_{d,0})\right]}. \quad (5.13)$$

The hybrid model parameters θ define the distribution of all the necessary values in (5.10) through (5.13). Hence the completed-data probability $p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta)$ can be evaluated in closed form, as required.

5.5.2 Hybrid State Distribution

Exact Hybrid State Estimation

Calculation of the distribution $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$ is a problem of hidden state estimation for hybrid systems. Different variations of this problem have received a great deal of attention in recent years[19, 61, 39, 80]. The state estimation problem addressed by [19, 61] among others, is to estimate the *current hybrid state* given all observations up to the current time step. We extend the work of [61] in order to determine $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$, the hybrid state distribution over the *entire sequence.* The key idea is to perform a forward-backward, or 'smoothing', Kalman Filter recursion for each possible mode sequence[121]. This recursion determines $p(\mathbf{x}_{c,0:f+1}|\mathbf{y}_{1:f+1}, \theta^k, \mathbf{x}_{d,0:f})$, the probability distribution over continuous state sequences conditioned on a particular mode sequence. In addition the forward Kalman Filter residuals are used to determine the observation likelihood conditioned on a mode sequence, $p(\mathbf{y}_{1:f+1}|\mathbf{x}_{d,0:f},\theta)$. Using this likelihood we evaluate $p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta)$, the posterior probability of the discrete mode sequence using Bayes' Rule:

$$p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta) = \frac{p(\mathbf{y}_{1:f+1}|\mathbf{x}_{d,0:f},\theta)p(\mathbf{x}_{d,0:f}|\theta)}{\sum_{\mathbf{x}_{d,0:f}} p(\mathbf{y}_{1:f+1},\mathbf{x}_{d,0:f}|\theta)p(\mathbf{x}_{d,0:f}|\theta)}. \tag{5.14}$$

The desired joint distribution $p(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)$ over mode sequences and continuous state *sequences* is then given by the expression:

$$p(\mathbf{x}_{c,0:f+1}|\mathbf{y}_{1:f+1},\mathbf{x}_{d,0:f},\theta^k)p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k). \tag{5.15}$$

Approximate Hybrid State Estimation

The calculation of the distribution (5.15) is intractable in practice, since it requires performing a forward-backward Kalman Filter recursion for every possible discrete mode sequence $\mathbf{x}_{d,0:f}$; the number of such mode sequences is exponential in f. In a similar spirit to [61], we therefore approximate the distribution (5.15) by performing the Kalman Filter recursion only for mode sequences in the restricted set $\mathcal{S}$. This introduces approximation in (5.15) in two distinct, but important ways. First, the probability of mode sequences that are not in $\mathcal{S}$ are assigned to zero. Second, the posterior probability of each mode sequence in $\mathcal{S}$ cannot be evaluated exactly; the sum in (5.14) over all possible mode sequences is no longer possible. This means that $p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta)$ can only be calculated accurate to a factor that is constant over the mode sequences. This is a well-known problem in approximate inference; a standard approach is to choose the factor so that the approximated posteriors, denoted $\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta)$, sum to one:

$$\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta) = \frac{1}{c}p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta), \tag{5.16}$$

where c is a normalization constant given by:

$$c = \sum_{\mathbf{x}_{d,0:f}\in\mathcal{S}} p(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k). \tag{5.17}$$

Denoting the approximate *joint* distribution as $\tilde{p}(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)$, we have:

$$\tilde{p}(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k) = \begin{cases} p(\mathbf{x}_{c,0:f+1}|\mathbf{y}_{1:f+1},\theta^k,\mathbf{x}_{d,0:f})\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k) & \mathbf{x}_{d,0:f}\in\mathcal{S} \\ 0 & \mathbf{x}_{d,0:f}\notin\mathcal{S}. \end{cases} \tag{5.18}$$

Given this approximation of $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$, we can write the approximation of the bound $h(\theta|\theta^k)$ as:

$$\tilde{h}(\theta|\theta^k) = \sum_{\mathbf{x}_{d,0:f}} \Bigg(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \int p(\mathbf{x}_{c,0:f+1}|\mathbf{x}_{d,0:f}, \mathbf{y}_{1:f+1}, \theta^k) \\ * \log p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta) d\mathbf{x}_{c,0:f+1} \Bigg) + \tilde{\mathcal{H}}. \tag{5.19}$$

Note that $h(\theta|\theta^k)$ is a special case of $\tilde{h}(\theta|\theta^k)$ for which the set $\mathcal{S}$ comprises all possible mode sequences $\mathbf{x}_{d,0:f}$.

The technical challenge in performing the approximation (5.18) is to choose the set of mode sequences $\mathcal{S}$. Previous work in hybrid state estimation[61] introduced the idea of capturing as much of the probability mass of the true distribution $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$ as possible in the approximate distribution, by choosing the *a posteriori most likely* mode sequences. In the Expectation-Maximization approach for hybrid learning we successively maximize a lower bound on the log likelihood $g(\theta)$. The exact bound $h(\theta|\theta^k)$ is the tightest bound possible[91], hence we choose the approximated bound $\tilde{h}(\theta|\theta^k)$ to be as close as possible to the exact bound $h(\theta|\theta^k)$. In order to do so we use the same heuristic as for approximate hybrid state estimation, namely to capture as much of the probability mass of the true distribution $p(\mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)$ as possible in the approximate Expectation Step. We therefore aim to include in $\mathcal{S}$ the K most likely mode sequences.

Calculation of the K most likely mode sequences is a challenging problem in itself. We use the iterative approach introduced by [61]. This approach calculates for each time step $\tau = 1, \ldots, f$ the K partial sequences $\mathbf{x}_{d,0:\tau}$ that maximize $\tilde{p}(\mathbf{x}_{d,0:\tau}|\mathbf{y}_{1:\tau+1}, \theta^k)$, restricting the search to sequences that are successors of the K partial sequences $\mathbf{x}_{d,0:\tau-1}$ stored at time step $\tau - 1$. The algorithm is illustrated in Fig. 5-4. Although this algorithm does not guarantee finding the most likely K sequences, in most cases it is sufficient and, most importantly, the algorithm is linear in the number of time steps.

In summary, we have introduced an approach for performing an approximate Expectation Step of the EM algorithm for hybrid model learning. The approach, described in full in Table 5.2, makes calculation of an approximation to the bound $h(\theta|\theta^k)$ tractable by restricting the number of discrete mode sequences considered to the set $\mathcal{S}$. In Section 5.7 we show how the number of sequences K affects the performance of the learning algorithm.

5.6 Maximization Step for Hybrid Model Learning

In the Maximization Step the bound $\tilde{h}(\theta|\theta^k)$ must be maximized over θ. It is only strictly necessary for $\tilde{h}(\theta|\theta^k)$ to be increased at each iteration, rather than maximized, for convergence of the EM

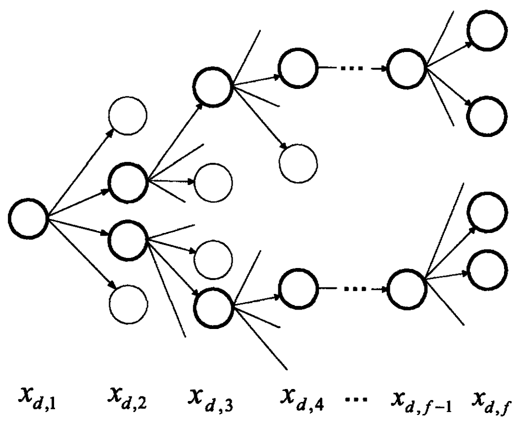


Figure 5-4. K-best enumeration. At each time step τ in the sequence, K sequences are stored. The successors of these sequences are enumerated, and the K partial sequences $\mathbf{x}_{d,0:\tau+1}$ with the greatest posterior probability $p(\mathbf{x}_{d,0:\tau+1}|\mathbf{y}_{1:\tau+2},\theta^k)$ are retained. In this example $K=4$ and the sequences retained at time f are shown in bold.

```
    Function ApproxEStep($\mathbf{y}_{1:f+1}$,$K$,$\theta^k$) returns $\tilde{p}(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)$
1)  Initialization. Initialize the set $\mathcal{S}$ of $K$-best partial sequences to contain the
    initial mode $\mathbf{x}_{d,0}$ and assign $\tau = 0$.
2)  K-best Enumeration. Find the $K$ partial sequences $\mathbf{x}_{d,0:\tau+1}$ that maximize
    the probability $\tilde{p}(\mathbf{x}_{d,0:\tau+1}|\mathbf{y}_{1:\tau+2},\theta^k)$. Evaluation of this probability requires
    performing a forward Kalman Filter step as described in [61]. Assign the set
    $\mathcal{S}$ to contain only these sequences.
3)  If $\tau < f$, set $\tau = \tau + 1$ and go to 2). Otherwise go to 4).
4)  Backward Kalman Filter Recursion. For each complete mode sequence
    $\mathbf{x}_{d,0:f}$ in $\mathcal{S}$, perform a backward Kalman Filter recursion to determine the
    probability $\tilde{p}(\mathbf{x}_{c,0:f+1},\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1},\theta^k)$. The forward Kalman Filter part of
    this process has already been performed in 2).
```

Table 5.2. Approximate Expectation Step for LPHA

algorithm; however in this section we show that for hybrid model learning in LPHA, the maximum of $\tilde{h}(\theta|\theta^k)$ can be found in closed form. In Section 5.6.1 we find the optimal continuous parameters θ_c, while in Section 5.6.2 we find the optimal discrete parameters θ_d.

5.6.1 Maximization Step for Continuous Model Parameters

The key insight in estimating the continuous model parameters is that for a given mode sequence, existing results for LTI systems[27] can be used to find the maximum. We express the full bound (5.9) as a summation over mode sequences. To find the maximum of $\tilde{h}(\theta|\theta^k)$ we set its derivative with respect to θ_c to zero. We can write this derivative as a summation over the derivative for each mode sequence:

$$\frac{\partial \tilde{h}(\theta|\theta^k)}{\partial \theta_c} = \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \frac{\partial}{\partial \theta_c} \int_{\mathbf{x}_{c,0:f+1}} p(\mathbf{x}_{c,0:f+1}|\mathbf{x}_{d,0:f}, \mathbf{y}_{1:f+1}, \theta^k) \right.$$
$$\left. * \log p(\mathbf{y}_{1:f+1}, \mathbf{x}_{c,0:f+1}, \mathbf{x}_{d,0:f}|\theta) d\mathbf{x}_{c,0:f+1} \right) = 0. \tag{5.20}$$

The optimal values for $A(\mathbf{x}_d)$ and $B(\mathbf{x}_d)$ are found by summing the LTI results from [27] over the mode sequences in $\mathcal{S}$ to give the following equations:

$$\begin{aligned}
&\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} P_{\tau+1,\tau}(\mathbf{x}_{d,0:f}) \right) = \\
&A^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} P_{\tau}(\mathbf{x}_{d,0:f}) \right) \\
&+ B^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{u}_\tau \hat{\mathbf{x}}_\tau{}'(\mathbf{x}_{d,0:f}) \right)
\end{aligned} \tag{5.21}$$

$$\begin{aligned}
&\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \hat{\mathbf{x}}_{\tau+1} \mathbf{u}_\tau{}' \right) = \\
&A^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \hat{\mathbf{x}}_\tau(\mathbf{x}_{d,0:f}) \mathbf{u}_\tau{}' \right) \\
&+ B^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{u}_\tau \mathbf{u}_\tau{}' \right),
\end{aligned} \tag{5.22}$$

where $\mathcal{F}(\mathbf{x}_{d,0:f})$ is the set of time steps in the sequence $\mathbf{x}_{d,0:f}$ for which the mode is $\mathbf{x}_d$. Members of $\mathcal{F}(\mathbf{x}_{d,0:f})$ are integers in the range $[0, f]$. Solving the set of linear equations (5.21), (5.22) yields the optimal values for $A(\mathbf{x}_d)$ and $B(\mathbf{x}_d)$. Similarly the optimal values for $C(\mathbf{x}_d)$ and $D(\mathbf{x}_d)$ are found by performing a weighted sum over the LTI results from [27] to give the system of linear equations:

$$\begin{aligned}
&\sum_{\mathbf{x}_{d,0:f}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{y}_{\tau+1} \hat{\mathbf{x}}_{\tau+1}{}'(\mathbf{x}_{d,0:f}) \right) = \\
&C^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} P_{\tau+1}(\mathbf{x}_{d,0:f}) \right) \\
&+ D^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{u}_\tau \hat{\mathbf{x}}_{\tau+1}{}'(\mathbf{x}_{d,0:f}) \right) \qquad (5.23)
\end{aligned}$$

$$\begin{aligned}
&\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,1:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{y}_{\tau+1} \mathbf{u}_\tau{}' \right) = \\
&C^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \hat{\mathbf{x}}_{\tau+1}(\mathbf{x}_{d,0:f}) \mathbf{u}_\tau{}' \right) \\
&+ D^*(\mathbf{x}_d) \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \mathbf{u}_\tau \mathbf{u}_\tau{}' \right). \qquad (5.24)
\end{aligned}$$

Using the optimal values for $A(\mathbf{x}_d)$, $B(\mathbf{x}_d)$, $C(\mathbf{x}_d)$ and $D(\mathbf{x}_d)$ we obtain the optimal covariance matrices for the noise processes:

$$\begin{aligned}
Q^*(\mathbf{x}_d) = &\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\frac{\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)}{|\mathcal{F}(\mathbf{x}_{d,0:f})|} \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \left(P_{\tau+1}(\mathbf{x}_{d,0:f}) \right.\right. \\
&\left.\left. - A^*(\mathbf{x}_d) P_{\tau,\tau+1}(\mathbf{x}_{d,0:f}) - B^*(\mathbf{x}_d) \mathbf{u}_\tau \hat{\mathbf{x}}_{\tau+1}{}'(\mathbf{x}_{d,0:f}) \right) \right) \qquad (5.25)
\end{aligned}$$

$$\begin{aligned}
R^*(\mathbf{x}_d) = &\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\frac{\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k)}{|\mathcal{F}(\mathbf{x}_{d,0:f})|} \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \left(\mathbf{y}_{\tau+1} \right.\right. \\
&\left.\left. - C^*(\mathbf{x}_d) \hat{\mathbf{x}}_{\tau+1}(\mathbf{x}_{d,0:f}) - D^*(\mathbf{x}_d) \mathbf{u}_\tau \right) \mathbf{y}_{\tau+1}{}' \right). \qquad (5.26)
\end{aligned}$$

Finally, the optimal parameters for the initial continuous distribution are given by:

$$\begin{aligned}
\mu^*(\mathbf{x}_d) = &\sum_{\{\mathbf{x}_{d,0:f}|\mathbf{x}_{d,0}=\mathbf{x}_d\}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \hat{\mathbf{x}}_0{}'(\mathbf{x}_{d,0:f}) \\
V^*(\mathbf{x}_d) = &\sum_{\{\mathbf{x}_{d,0:f}|\mathbf{x}_{d,0}=\mathbf{x}_d} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) P_{0,0}(\mathbf{x}_{d,0:f}), \qquad (5.27)
\end{aligned}$$

where the summation is over the discrete mode sequences $\mathbf{x}_{d,0:f}$ for which the initial mode $\mathbf{x}_{d,0}$ is the same as $\mathbf{x}_d$.

In (5.21) through (5.27) we use the following definitions:

$$\begin{aligned}
\hat{\mathbf{x}}_\tau(\mathbf{x}_{d,0:f}) = &E\left[\mathbf{x}_{c,\tau}|\mathbf{x}_{d,0:f}, \mathbf{y}_{1:f+1}, \theta^k\right] \\
P_{\tau_1,\tau_2}(\mathbf{x}_{d,0:f}) = &E\left[\mathbf{x}_{c,\tau_1} \mathbf{x}_{c,\tau_2}{}'|\mathbf{x}_{d,0:f}, \mathbf{y}_{1:f+1}, \theta^k\right]. \qquad (5.28)
\end{aligned}$$

We have therefore shown that for LPHA, calculation of the optimal continuous parameters in the maximization step can be carried out in closed form. By choosing the set $\mathcal{S}$ to contain a subset of the possible discrete mode sequences we make this calculation tractable despite the exponential number of such sequences.

5.6.2 Maximization Step for Discrete Model Parameters

We now maximize the bound $\tilde{h}(\theta|\theta^k)$ with respect to the discrete parameters θ_d. For each guard condition c_i we must find the optimal value of the transition matrix $\mathcal{T}_i$. It is not sufficient simply to set the derivative of $\tilde{h}(\theta|\theta^k)$ with respect to each element of $\mathcal{T}_i$ to zero; in order to ensure that $\mathcal{T}_i$ describes a valid transition probability matrix, we must impose the constraint that each column in $\mathcal{T}_i$ has elements that sum to one. We therefore perform the following constrained optimization for every mode $\mathbf{x}_d \in \mathcal{X}_d$ and guard $i \in \mathcal{C}$:

$$\text{Maximize over } [\mathcal{T}_i]_{j,\mathbf{x}_d},\ j = 1, \dots, |\mathcal{X}_d| : \tag{5.29}$$

$$\tilde{h}(\theta|\theta^k)$$

$$\text{Subject to:}$$

$$\sum_{j=1}^{|\mathcal{X}_d|} [\mathcal{T}_i]_{j,\mathbf{x}_d} = 1, \tag{5.30}$$

where we use the notation $[\mathcal{T}_i]_{a,b}$ to denote the (a, b)'th element of $\mathcal{T}_i$.

We use a Lagrange multiplier approach to solve this optimization. The Lagrangian L is given by:

$$L = \tilde{h}(\theta|\theta^k) + \lambda\left(\sum_{j=1}^{|\mathcal{X}_d|} [\mathcal{T}_i]_{j,\mathbf{x}_d} - 1\right), \tag{5.31}$$

where λ is a Lagrange multiplier. We must set the derivative of L with respect to each term $\mathcal{T}_i(j, \mathbf{x}_d)$ as well as λ to zero. In order to do so we must evaluate the expression $\frac{\partial \tilde{h}(\theta|\theta^k)}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}}$:

$$\frac{\partial \tilde{h}(\theta|\theta^k)}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}} = \frac{\partial}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}} \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) *$$
$$\sum_{\tau=1}^{f} \int p(\mathbf{x}_{c,\tau-1}|\mathbf{y}_{1:f+1}, \mathbf{x}_{d,0:f}, \theta^k) \log p(\mathbf{x}_{d,\tau}|\mathbf{x}_{d,\tau-1}, \mathbf{x}_{c,\tau-1}, \theta) d\mathbf{x}_{c,\tau-1}. \tag{5.32}$$

Evaluating the derivative in this expression is made challenging by the dependence of mode transitions on the continuous state. However since the distribution $p(\mathbf{x}_{d,\tau}|\mathbf{x}_{d,\tau-1}, \mathbf{x}_{c,\tau-1}, \theta)$ has a constant value for each guard condition c_i, we can rewrite the integral over $\mathbf{x}_{c,\tau-1}$ as a sum over guard conditions:

$$\frac{\partial \tilde{h}(\theta|\theta^k)}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}} = \frac{\partial}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}} \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau=1}^{f} \sum_{c_i \in \mathcal{G}} p_{c_i}(\mathbf{x}_{d,0:f}) \log \mathcal{T}_i(\mathbf{x}_{d,\tau}, \mathbf{x}_{d,\tau-1}), \tag{5.33}$$

where $p_{c_i}(\mathbf{x}_{d,0:f})$ is the probability that guard condition c_i is satisfied given the discrete mode sequence $\mathbf{x}_{d,0:f}$, the observation sequence $\mathbf{y}_{1:f+1}$ and the current guess of the parameters θ^k. This probability can be written as:

$$p_{c_i}(\mathbf{x}_{d,0:f}) = \int_{\mathcal{C}_i} p(\mathbf{x}_{c,\tau-1}|\mathbf{y}_{1:f+1}, \mathbf{x}_{d,0:f}, \theta^k) d\mathbf{x}_{c,\tau-1}, \tag{5.34}$$

where $\mathcal{C}_i$ is the region of $\mathbf{x}_{c,\tau-1}$ for which the guard c_i is satisfied. Evaluating (5.34) requires integrating a Gaussian over $\mathcal{C}_i$. Prior work has shown that for special classes of guard conditions c_i such as rectangular and linear guards, the integral can be evaluated efficiently using a lookup of Gaussian cumulative distribution functions[61]. Using this approach we can evaluate the derivative in (5.33) with respect to $[\mathcal{T}_i]_{j,\mathbf{x}_d}$:

$$\frac{\partial \tilde{h}(\theta|\theta^k)}{\partial [\mathcal{T}_i]_{j,\mathbf{x}_d}} = \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \left(\tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} \frac{p_{c_i}(\mathbf{x}_{d,0:f})}{[\mathcal{T}_i]_{j,\mathbf{x}_d}} \right), \tag{5.35}$$

where $\mathcal{F}(\mathbf{x}_{d,0:f})$ contains the time steps in $\mathbf{x}_{d,0:f}$ for which $\mathbf{x}_{d,\tau-1} = \mathbf{x}_d$ and $\mathbf{x}_{d,\tau} = j$. Using the expression (5.35) we now set to zero the derivative of the Lagrangian (5.31) with respect to $[\mathcal{T}_i]_{j,\mathbf{x}_d}$ to yield:

$$\lambda \cdot [\mathcal{T}_i]_{j,\mathbf{x}_d} = - \sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} p_{c_i}(\mathbf{x}_{d,0:f}). \tag{5.36}$$

We now sum over each target mode j to yield:

$$\lambda \sum_{j=1}^{|\mathcal{X}_d|} [\mathcal{T}_i]_{j,\mathbf{x}_d} = - \sum_{j=1}^{|\mathcal{X}_d|} \left(\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} p_{c_i}(\mathbf{x}_{d,0:f}) \right). \tag{5.37}$$

Setting the derivative of L with respect to λ to zero yields the original constraint (5.30). Substituting this into (5.37) gives the optimal value for λ. We substitute this back into (5.36) to give the following expression for the optimal value of $[\mathcal{T}_i]_{j,\mathbf{x}_d}$:

$$[\mathcal{T}_i]^*_{j,\mathbf{x}_d} = \frac{\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} p_{c_i}(\mathbf{x}_{d,0:f})}{\sum_{\mathbf{x}_d \in \mathcal{X}_d} \left(\sum_{\mathbf{x}_{d,0:f} \in \mathcal{S}} \tilde{p}(\mathbf{x}_{d,0:f}|\mathbf{y}_{1:f+1}, \theta^k) \sum_{\tau \in \mathcal{F}(\mathbf{x}_{d,0:f})} p_{c_i}(\mathbf{x}_{d,0:f}) \right)}. \tag{5.38}$$

Using a similar Lagrange multiplier method to determine the optimal initial mode distribution, we arrive at the following optimal parameters:

$$p^*(\mathbf{x}_{d,0} = \mathbf{x}_d) = \frac{\sum_{\mathbf{x}_{d,0:f} \in \mathcal{Q}} \tilde{p}(\mathbf{x}_{d,0:f} | \mathbf{y}_{1:f+1}, \theta^k)}{\sum_{\mathbf{x}_d \in \mathcal{X}_d} \left(\sum_{\mathbf{x}_{d,0:f} \in \mathcal{Q}} \tilde{p}(\mathbf{x}_{d,0:f} | \mathbf{y}_{1:f+1}, \theta^k) \right)}, \tag{5.39}$$

where $\mathcal{Q}$ is the set of mode sequences $\mathbf{x}_{d,0:f}$ for which the initial mode $\mathbf{x}_{d,0} = \mathbf{x}_d$. In other words, the optimal initial discrete mode distribution is found by taking the *weighted fraction of initial modes* in the sequence set $\mathcal{S}$; the weighting is the posterior probability of each mode sequence given the current guess for the parameters.

In summary, we have shown that the Maximization Step for LPHA can be performed analytically, by providing a closed form expression for the hybrid parameters θ that maximize the bound $\tilde{h}(\theta|\theta^k)$. In the case where $\tilde{h}(\theta|\theta^k) = h(\theta|\theta^k)$, i.e. a complete Expectation Step has been carried out, the Maximization step is typically intractable due to the exponential number of mode sequences in the set $\mathcal{S}$. However, by restricting the size of the set $\mathcal{S}$, the Maximization Step is made tractable. Note that the Maximization Step is still guaranteed to find the maximum of the bound $\tilde{h}(\theta|\theta^k)$; in other words restricting the size of the set $\mathcal{S}$ in the Expectation Step does not introduce suboptimality in the Maximization Step.

5.7 Simulation Results

In this section we demonstrate the new model learning approach using a simulated planetary rover example.[2] We consider the subsystem consisting of a motor and a wheel. An intermittent fault causes the wheel to 'stick' at random, and the probability of the wheel sticking is different depending on whether the wheel is being driven forwards or backwards. When stuck, the wheel experiences increased friction.

The wheel subsystem is modeled as a LPHA with two modes. In Mode 1 the wheel operates normally, while Mode 2 the wheel is stuck. The hidden continuous state $\mathbf{x}_c$ is $\left[i \ \dot{\theta}\right]'$ where i is the current in the motor and $\dot{\theta}$ is the angular velocity of the wheel. Noisy observations $\mathbf{y}$ of the wheel velocity are available through an encoder. The input $\mathbf{u}$ is the voltage applied to the driver circuit.

[2]The empirical results presented here were generated in conjunction with Stephanie Gil.

The true continuous parameters are given by:

$$\begin{aligned}
A(1) &= \begin{bmatrix} -0.0044 & -0.0203 \\ 0.0366 & 0.1665 \end{bmatrix} \; B(1) = \begin{bmatrix} 0.92 \\ 0.81 \end{bmatrix} \\
A(2) &= \begin{bmatrix} -0.0032 & -0.0142 \\ 0.0256 & 0.1106 \end{bmatrix} \; B(2) = \begin{bmatrix} 0.93 \\ 0.71 \end{bmatrix} \\
C(1) = C(2) &= \begin{bmatrix} 0 & 1 \end{bmatrix} \; C(1) = D(2) = 0.
\end{aligned} \tag{5.40}$$

The true guard conditions are given by:

$$\begin{aligned}
\mathcal{C}_1 = [-\infty\ 0] \;\; \mathcal{T}_1 &= \begin{bmatrix} 0.9 & 0.2 \\ 0.1 & 0.8 \end{bmatrix} \\
\mathcal{C}_2 = [0\ \infty] \;\; \mathcal{T}_2 &= \begin{bmatrix} 0.5 & 0.1 \\ 0.5 & 0.9 \end{bmatrix},
\end{aligned} \tag{5.41}$$

where the guard regions $\mathcal{C}_1$ and $\mathcal{C}_2$ are defined over $\dot{\theta}$.

While we would like to evaluate the performance of the new algorithm against the likelihood $p(\mathbf{y}_{1:f+1}|\theta)$, evaluation of this is intractable. The change in the lower bound $\tilde{h}(\theta|\theta^k)$ can, however, be evaluated. We therefore analyze the convergence of the algorithm in terms of this change. Fig. 5-5 shows the change in the lower bound over the course of a typical learning run. The change in $\tilde{h}(\theta|\theta^k)$ is always positive, and decreases almost monotonically as learning proceeds, indicating convergence of the learned parameters. As has been previously demonstrated for LTI systems[27] the continuous model parameters typically do not converge to the true parameters. This is because, first, there are an infinite number of equivalent LTI systems, and second, because the EM algorithm is a local optimization approach. In many cases, however, the *discrete* parameters *do* converge to values close to the true ones. A typical case is shown in Fig 5-6.

Fig. 5-7 shows the effect of the number of tracked mode sequences on the performance of the algorithm. We cannot use the likelihood $p(\mathbf{y}_{1:f+1}|\theta)$ to evaluate performance, but since we are often interested in estimating the hidden mode, we use the fraction of Maximum A Posteriori (MAP) mode estimate errors at convergence as a performance criterion. Counterintuitively, the number of tracked mode sequences K has essentially no impact on the performance of the learning algorithm. Extensive testing also showed that increasing K does not, on average, increase the number of iterations to convergence. These results motivate the use of 'hard' EM, where the single most likely trajectory is tracked, in order to minimize computation time.

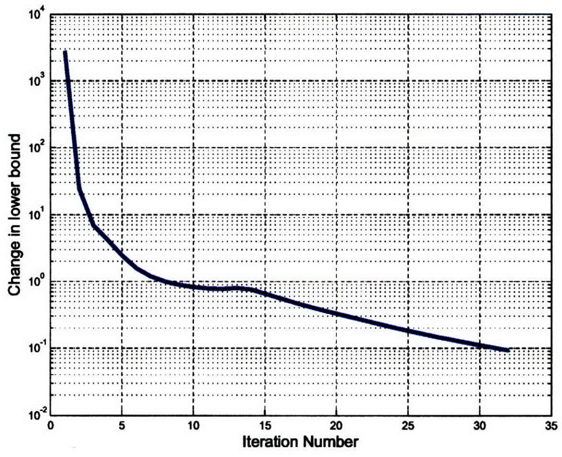


Figure 5-5. Convergence of the approximate EM algorithm for hybrid model learning for a typical run, with $K = 10$ and $f = 100$. The change in the bound $\tilde{h}(\theta|\theta^k)$ decreases almost monotonically as EM proceeds. This shows convergence of the learned parameters.

5.8 Conclusion

We have presented a new approximate Expectation-Maximization approach for learning the parameters of dynamic systems with hybrid discrete-continuous state. The new method can handle transitions in the discrete state that are conditioned on the continuous state, and can learn the effects of control inputs on the system. Approximation is introduced by tracking a subset of the discrete mode sequences in the Expectation step. Simulation results showed that the approach converges in practice, and that the number of tracked mode sequences does not affect the performance of the algorithm.

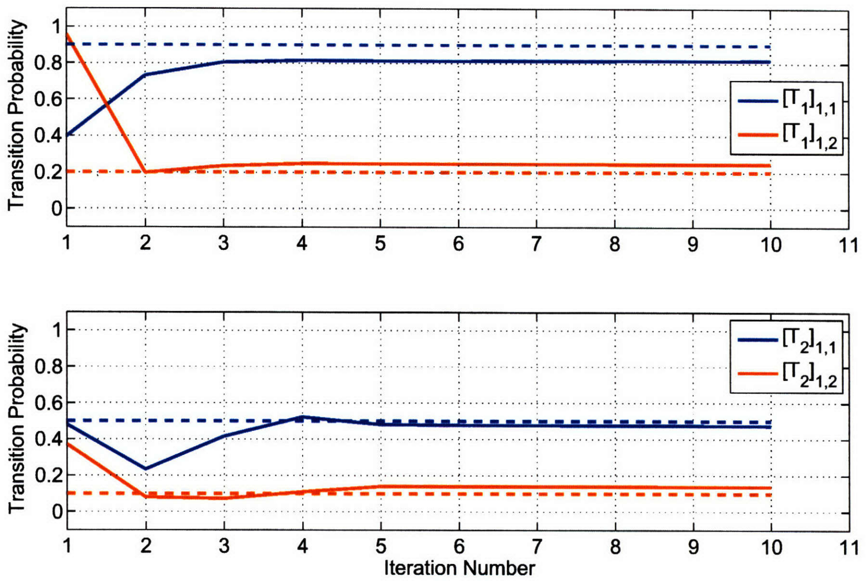


Figure 5-6. Convergence of the discrete parameters for a typical run, with $K = 10$ and $f = 100$. The transition probabilities for specification 1 (top) and specification 2 (bottom) converge to values close to the true ones (shown dashed).

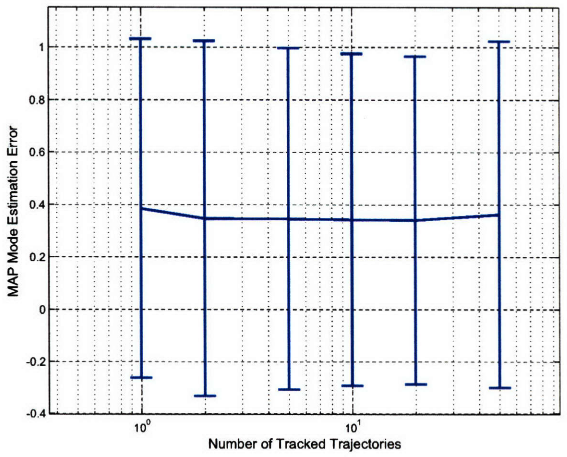


Figure 5-7. Fraction of MAP mode estimation errors at convergence against number of mode sequences tracked (K) averaged over 20 random runs. Convergence was defined as a change of less than 0.1 in $\tilde{h}(\theta|\theta^k)$. For each run, the initial continuous parameters were chosen randomly by perturbing their true values by up to 50%, and the discrete transition probabilities were chosen from a uniform distribution between 0 and 1. The error bars represent two standard deviations. K has no almost effect on the MAP error.

Chapter 6

Future Work

There are a number of interesting avenues for future research on the topic of execution of probabilistic systems. We elaborate on some of these here.

Uncertain obstacle locations and models. In this thesis we did not demonstrate the Particle Control algorithm in the case where the feasible region itself is uncertain, or the system model is uncertain. The algorithm in its current form can handle such forms of uncertainty, simply by sampling from the uncertain distributions. Uncertainty in the feasible region, in particular, has a number of important applications; for example in the case of an AUV, the topology of the sea floor is known only approximately. In the case of UAV path planning, the locations of the obstacles may be uncertain. Future work will investigate the performance of the Particle Control algorithm with obstacle uncertainty and model uncertainty.

Particle Control for more general systems. The appeal of the Particle Control formulation for solving chance-constrained problems is its generality. In principle this formulation can be applied to a very general class of system including nonlinear systems, systems with discrete jumps, and systems with mixed discrete-continuous control inputs. However the optimization that results from the Particle Control formulation is not necessarily tractable. In this thesis we showed that in the case of linear dynamic systems and Jump Markov Linear Systems, MILP can be used to solve the optimization to global optimality. Future work will aim to find ways to make optimization tractable for Particle Control with more general system classes.

Bias and variance reduction. The Particle Control approach is stochastic, in the sense that the solution returned by the algorithm is stochastic; running the algorithm with the same parameters will not necessarily produce the same solution. This stochasticity arises from the fact that Particle Control approximates the original stochastic problem, each time it is invoked, by sampling from random variables. In Section 3.6 we showed that the Particle Control approach is biased; that is, on average the solution gives a probability of failure that is greater than the specified maximum. In addition, the variance is relatively high for small particle sets; that is, the variability of the optimal solution is high. Future work will aim to reduce the bias and variance for a given size of particle set. One possible approach is to use quasi-random sampling[58]. The key idea is, instead of drawing samples completely at random, to ensure that at least one sample is drawn from each region of the state space. This preserves convergence properties, while ensuring a more even distribution of particles across the state space. In many applications this has been shown to reduce variance, and there is a strong possibility that it would reduce bias and variance in the Particle Control case.

Approximation error results. The Particle Control approach solves an approximation to a full stochastic control problem. The approximation is generated through sampling, and we have shown that as the number of particles tends to infinity, the approximation error goes to zero. An open question, however, is the number of particles required to achieve a given level of accuracy. One way of expressing this question is to ask *'what is the probability, that the true probability of failure is below the specified maximum, as a function of the number of particles used?'*. Since we specify the maximum probability of failure as a hard constraint, we would like to know how many particles are necessary to have high confidence that it is satisfied by any solution returned by the Particle Control approach. Future work will derive analytic results to answer such questions.

Maximum probability of safety. This thesis demonstrated that in many situations, a chance-constrained formulation is a convenient way of posing the robust execution problem. It has a dual problem formulation, however, which is to minimize the probability of failure subject to constraints on the objective function. While it was not demonstrated in this thesis, this alternative formulation is readily handled within the Particle Control approach. Future work will investigate this alternative formulation.

Analytic results for Particle Control in closed loop. In this thesis we presented a receding-horizon approach to execution and proposed a number of algorithms for solving the single-stage, finite-horizon execution problem. We did not, however, present any analytical results pertaining to the use of Particle Control in receding-horizon. Two such results of prime importance are *robust stability* and *robust feasibility*. The first states that, when applied in receding horizon, the overall control strategy is stable even when disturbances are applied to the system. The second states that the constrained optimization problem carried out at each time step is feasible; in other words, the controller will not drive itself into a situation where, at the planning iteration, it becomes stuck. Both of these properties have been researched extensively in the case of receding-horizon control under set-bounded uncertainty. However to the author's knowledge the literature on this topic is very sparse for chance-constrained control under stochastic uncertainty. Future work will look for analytic proofs of robust stability and robust feasibility for the Particle Control approach.

Empirical comparison of Particle Control with set-bounded approaches. As mentioned in Section 3.2, an alternative approach to Particle Control is to convert the probabilistic uncertainty representation into a set-bounded one, and to employ approaches designed for set-bounded uncertainty. Such approaches find a *conservative* solution to the stochastic planning problem, while Particle Control find an *approximate* solution to the problem. Set-bounded control approaches are typically less computationally intensive than Particle Control , but the generation of the equivalent set-bounded uncertainty representation requires additional computational resources. An empirical investigation is therefore necessary to compare the two approaches rigorously; this is a topic for future work.

Active hybrid estimation with non-Gaussian noise. The Active Hybrid Estimation approach described in this thesis is restricted to the case where all forms of uncertainty are Gaussian. While the AE-QSP algorithm used sampling to approximate chance constraints, the cost function minimized in the approach still relies on the Gaussian assumption. The Particle Control approach is able to deal with arbitrary forms of uncertainty by using a sampling approach. Future work will investigate the use of sampling to approximate the probability of pruning the true mode sequence in approximate hybrid estimation. The key difficulty in doing so is that, for a distribution approximated as a set of samples, we lose the partitioning of the state space into decision regions that occurs for full distributions; for the Gaussian case, we used this partitioning to derive analytic results bounding the probability of pruning. It is possible that this problem would be avoided by assuming a stochastic hybrid estimation scheme, such as a particle filter, rather than K-best enumeration.

Model learning using stochastic approaches. This thesis highlighted a number of problems involved in the application of Expectation-Maximization (EM) to learning of Probabilistic Hybrid Automata. In particular, applying a local optimization algorithm such as EM to maximize a value function that (we believe) has a very large number of local maximum is inherently limited. In a large number of domains, similar problems have been overcome by using stochastic optimization approaches. A wealth of such approaches now exist, see [125] for an overview. Future work will look to apply these approachs to the hybrid model learning problem.

Appendix A

Alternatives to Monte-Carlo Approach

The Particle Control approach uses a sampling, or 'Monte-Carlo' approach, to approximate the stochastic optimization problem as a tractable deterministic one. In this section we consider alternatives to the Monte-Carlo approach and motivate the use of a Monte-Carlo approach.

A.1 Discrete-Time Propagation of Non-Gaussian Distributions

In this section we show how general distributions can be propagated through linear discrete-time dynamics.

Consider the linear equation:

$$\mathbf{y} = F\mathbf{x} + G, \tag{A.1}$$

where $\mathbf{y}$ and $\mathbf{x}$ are random variables in $\mathbb{R}^{n_x}$. The matrices F and G are constant. We are given the probability density function $p_\mathbf{x}(\mathbf{x})$, which need not be Gaussian, defined such that:

$$P\Big((x_1, \dots, x_{n_x}) \in C\Big) = \int_C p_\mathbf{x}(\mathbf{x}) dx_1 \dots dx_{n_x}, \tag{A.2}$$

where x_i is the i'th element of $\mathbf{x}$ and C is a subset of $\mathbb{R}^{n_x}$. We would like to calculate the probability density function $p_\mathbf{y}(\mathbf{y})$. The following theorem allows us to do this:

Theorem A.1. *If* $\mathbf{y} = F\mathbf{x} + G$ *where* F *is invertible, then:*

$$p_{\mathbf{y}}(\mathbf{y}) = \frac{p_{\mathbf{x}}(F^{-1}(\mathbf{y} - G))}{|F|} \tag{A.3}$$

Proof: This proof follows the development in [55]. Assume there is a one-to-one mapping T between $\mathbf{x}$ and $\mathbf{y}$ such that $T : (x_1, \dots, x_{n_x}) \mapsto (y_1, \dots, y_{n_x})$, We denote the inverse transformations as $x_1(y_1, \dots, y_{n_x})$ through $x_{n_x}(y_1, \dots, y_{n_x})$. Consider the integral:

$$I = \int_A g(x_1, \dots, x_{n_x}) dx_1 \dots dx_{n_x}, \tag{A.4}$$

where A is an arbitrary region such that $A \subseteq \mathbb{R}^{n_x}$ and g is a function $g : \mathbb{R}^{n_x} \longrightarrow \mathbb{R}$. We now perform a change of variables to express I in terms of an integration over the space of $\mathbf{y}$. Define a region B such that:

$$\mathbf{x} \in A \Longleftrightarrow \mathbf{y} \in B, \tag{A.5}$$

then:

$$I = \int_B g\Big(x_1(y_1, \dots, y_{n_x}), \dots, x_{n_x}(y_1, \dots, y_{n_x})\Big) |J(y_1, \dots, y_{n_x})| dy_1 \dots dy_{n_x}, \tag{A.6}$$

where J is the Jacobian of g such that:

$$J = \begin{bmatrix} \frac{\partial x_1}{\partial y_1} & \cdots & \frac{\partial x_{n_x}}{\partial y_1} \\ & \ddots & \\ \frac{\partial x_1}{\partial y_{n_x}} & \cdots & \frac{\partial x_{n_x}}{\partial y_{n_x}} \end{bmatrix} \tag{A.7}$$

We assume that the partial derivatives are continuous and J is finite on the range of the mapping T. We now set the function g to be the probability density function $p_{\mathbf{x}}(\mathbf{x})$. The equations (A.4) and (A.6) give us:

$$\int_A p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = \int_B p_{\mathbf{x}}\Big(x_1(\mathbf{y}), \dots, x_{n_x}(\mathbf{y})\Big) |J(\mathbf{y})| d\mathbf{y}. \tag{A.8}$$

Also notice that:

$$P(\mathbf{x} \in A) = \int_A p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = P(\mathbf{y} \in B) = \int_B p_{\mathbf{y}}(\mathbf{y}) d\mathbf{y}. \tag{A.9}$$

Comparing the result (A.8) and the equality (A.9) we see that:

$$p_{\mathbf{y}}(\mathbf{y}) = p_{\mathbf{x}}(x_1(\mathbf{y}), \dots, x_{n_x}(\mathbf{y})) |J(\mathbf{y})|. \tag{A.10}$$

We now apply this to the linear equation (A.1). Since F is invertible we have a one-to-one mapping between $\mathbf{x}$ and $\mathbf{y}$, with the inverse transformation given by:

$$\mathbf{x} = F^{-1}(\mathbf{y} - G). \tag{A.11}$$

For this transformation the Jacobian is given by $|F|^{-1}$, which is constant. Using (A.10) we have the result:

$$p_{\mathbf{y}}(\mathbf{y}) = \frac{p_{\mathbf{x}}(F^{-1}(\mathbf{y} - G))}{|F|}. \tag{A.12}$$

□

We now use Theorem A.1 to propagate the probability distributions in Linear Time Invariant (LTI) discrete-time systems. Consider the LTI system defined by:

$$\mathbf{x}_{\tau+1} = A\mathbf{x}_\tau + B\mathbf{u}_\tau + \omega_\tau, \tag{A.13}$$

where $\mathbf{x}$ and ω_τ are in $\mathbb{R}^{n_x}$ and $\mathbf{u}_\tau$ is in $\mathbb{R}^{n_u}$. We are given the distribution of the initial state $p_{\mathbf{x}_0}(\mathbf{x}_0)$ and of the disturbance process $p_{\omega_\tau}(\omega_\tau)$ for all τ. The variable $\mathbf{u}_\tau$ is the control input at time step τ, which is known (nonrandom). The matrices A and B are constant. We now define:

$$\mathbb{X} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_{N_p} \end{bmatrix} \quad \mathbb{U} = \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_p-1} \end{bmatrix} \quad \mathbb{W} = \begin{bmatrix} \mathbf{x}_0 \\ \omega_0 \\ \omega_1 \\ \vdots \\ \omega_{N_p-1} \end{bmatrix} \tag{A.14}$$

The linear relationship between these block matrices can be written as follows:

$$\mathbb{X} = F\mathbb{W} + G, \tag{A.15}$$

where F is calculated by repeated application of the LTI dynamics (A.13):

$$F = \begin{bmatrix} I & 0 & 0 & \dots & 0 \\ A & I & 0 & \dots & 0 \\ A^2 & A & I & \dots & 0 \\ & & & \ddots & \\ A^{N_p} & & & \dots & I \end{bmatrix} \tag{A.16}$$

and G is given by:

$$G = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ A^2B & AB & B & \dots & 0 \\ & & & \ddots & \\ A^{N_p-1} & & & \dots & B \end{bmatrix} \mathbb{U} \tag{A.17}$$

In order to apply Theorem A.1 it remains to show that F is invertible. This is the case if F is full rank, in other words F has $n_x \times (N_P + 1)$ linearly independent rows. Notice first that each block of n_x rows is linearly independent due to the lower triangular structure of F. In other words $[I\ 0\ \dots\ 0]$ is linearly independent of $[A\ I\ 0\ \dots\ 0]$, which is linearly independent of $[A^2\ A\ I\ \dots\ 0]$ and so on. It remains then to show that each block of n_x rows has n_x linearly independent rows. This is clearly true if A has n_x linearly independent rows. Therefore F is invertible as long as A is of full rank. In the LTI system (A.13), if A is not of full rank, then one or more elements of the state vector $\mathbf{x}$ are linearly equivalent; hence the system can be expressed as an equivalent, smaller, LTI system in which A *is* of full rank. In general, therefore, Theorem A.1 can be applied to LTI discrete-time systems, and we have:

$$p_{\mathbb{X}}(\mathbb{X}) = \frac{p_{\mathbb{W}}(F^{-1}(\mathbb{X} - \mathbb{G}))}{|F|}. \tag{A.18}$$

In this thesis, we are concerned with hybrid discrete-continuous systems, in particular Jump Markov Linear Systems (JMLS). Given the relation in (A.18) is is straightforward in principle to calculate the state distribution for a JMLS. First we enumerate each possible discrete mode sequence, and calculate the probability of the mode sequence, denoted $p(\mathbf{x}_{d,0:N_p})$. Each mode sequence describes a Linear Time Varying discrete-time system, for which we can calculate the matrices F and G. We can therefore calculate the distribution of the continuous state $\mathbf{x}_{c,0:N_p}$, *conditioned* on a discrete mode sequence, denoted $p(\mathbf{x}_{c,0:N_p}|\mathbf{x}_{d,0:N_p})$. By performing a marginalization over the discrete mode sequences, we obtain the distribution of the continuous state sequence $\mathbf{x}_{c,0:N_p}$:

$$p(\mathbf{x}_{c,0:N_p}) = \sum_{\mathbf{x}_{d,0:N_p}} p(\mathbf{x}_{c,0:N_p}|\mathbf{x}_{d,0:N_p})p(\mathbf{x}_{d,0:N_p}). \tag{A.19}$$

Furthermore, we can do this for *arbitrary* distributions for the disturbances ω_τ and the initial state $\langle \mathbf{x}_{c,0}, \mathbf{x}_{d,0} \rangle$.

A.2 Continuous-Time Propagation of Non-Gaussian Distributions

In this thesis we restrict our attention to control and estimation of discrete-time system models. This is convenient for a number of reasons. First, it simplifies the definition of the hybrid discrete-continuous system dynamics; the discrete dynamics, for instance, can be represented in a compact transition probability matrix form. Second, it parameterizes the optimal control problem, reducing the problem from an infinite-dimensional one to a finite-dimensional one. Third, the hybrid estimation problem is simplified, since there is a finite number of possible mode sequences for a given time horizon. This simplification applies to the hybrid learning problem, since also there we calculate a distribution over discrete mode sequences. In each of the three fields of control, estimation, and learning, a large number of prior authors have used discrete-time systems for precisely these reasons, for example [106, 50, 48, 118, 43, 61, 110].

The real-world plants that we model, however, operate in continuous time. In applying the methods we develop in this thesis to such plants, the discrete-time assumption is an approximation. Standard methods exist for approximating continuous-time systems as discrete-time systems[43]. For the sake of completeness, in this section we consider the propagation of general probability distributions through *continuous-time* linear dynamics.

A continuous-time system driven by a random process can be modeled as:

$$d\mathbf{x}(t) = f(\mathbf{x}(t), t)dt + G(\mathbf{x}(t), t)d\mathbf{w}(t), \tag{A.20}$$

where $\mathbf{x}(t)$ is the system state at time t. The variable $\mathbf{w}(t)$ is a Brownian motion process, which is defined as follows:

1. $\mathbf{w}(0) = 0$ with probability one.

2. $\mathbf{w}(t) - \mathbf{w}(s)$ is Gaussian with distribution $\mathcal{N}(0, t-s)$ for all $t \geq s \geq 0$.

3. For all times $0 < t_1 < t_2 < \cdots < t_n$, the random variables $\mathbf{w}(t_1), \mathbf{w}(t_2) - \mathbf{w}(t_1), \ldots, \mathbf{w}(t_n) - \mathbf{w}(t_{n-1})$ are independent.

White noise is defined as the derivative of Brownian motion, so that (A.20) can, heuristically, be written:

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), t) + G(\mathbf{x}(t), t)\omega(t), \tag{A.21}$$

where $\omega(t)$ is continuous-time white noise. This equation now has a very similar form to discrete-time nonlinear stochastic dynamics:

$$\mathbf{x}_{\tau+1} = f(\mathbf{x}_\tau, t) + G(\mathbf{x}_\tau, t)\omega_\tau, \tag{A.22}$$

where ω_τ is a discrete-time noise process. The reason that (A.21) is a heuristic expression of (A.20) is that Brownian motion $\mathbf{w}(t)$ is nowhere differentiable[136].

The equation (A.20) is an example of a Stochastic Differential Equation, for which the solution $\mathbf{x}(t)$ satisfies:

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t f(\mathbf{x}(\tau), \tau) d\tau + \int_0^t G(\mathbf{x}(t), t) d\mathbf{w}(\tau). \tag{A.23}$$

The first integral in (A.23) is an ordinary time integral, while the second is an *Itô integral*, and the stochastic differential equation (A.20) is refered to as the *Itô equation.* The 'normal' rules of calculus do not apply in evaluating these integrals. A review of Itô calculus is not necessary for the present discussion, for a detailed discussion of Itô integrals, refer to [100].

In our case we are interested in determining the distribution of $\mathbf{x}(t)$. We assume that $E[d\mathbf{w}(t)] = 0$ and $E[d\mathbf{w}d\mathbf{w}(t)'] = Q(t)dt$. Then the *Fokker-Planck* equation (also known as the Kolmogorov Forward Equation) provides a partial differential equation for the evolution of $p(\mathbf{x}(t))$:

$$\frac{\partial p(\mathbf{x}(t))}{\partial t} = -\sum_{i=1}^{n} \frac{\partial p(\mathbf{x}(t))}{\partial \mathbf{x}_i} + \frac{1}{2} \sum_{i,j=1}^{n} \frac{p(\mathbf{x}(t)) \left[G(\mathbf{x}(t)) Q(t) G(\mathbf{x}(t))' \right]_{ij}}{\partial \mathbf{x}_i \partial \mathbf{x}_j}, \tag{A.24}$$

where $[\cdot]_{ij}$ denotes the (i,j)'th element in the matrix, and n is the size of the vector $\mathbf{x}$. While this provides a partial differential equation for $p(\mathbf{x}(t))$, solution of the partial differential equation is by no means straightforward in the general case. [46] provides a detailed handling of special cases that can be solved exactly. The case of linear system dynamics and Gaussian initial conditions is one such special case:

$$\begin{aligned} d\mathbf{x}(t) &= A(t)\mathbf{x}(t)dt + G(t)d\mathbf{w}(t) \\ p(\mathbf{x}(t), 0) &= \mathcal{N}(\hat{\mathbf{x}}(0), P(0)). \end{aligned} \tag{A.25}$$

Here the solution is given by:

$$p(\mathbf{x}(t)) = \mathcal{N}(\hat{\mathbf{x}}(t), P(t)), \tag{A.26}$$

where $\hat{\mathbf{x}}(t)$ satisfies:

$$\dot{\hat{\mathbf{x}}}(t) = A(t)\hat{\mathbf{x}}(t) \tag{A.27}$$

and $P(t)$ satisfies:

$$\dot{P}(t) = A(t)P(t) + P(t)A'(t) + G(t)Q(t)G'(t). \tag{A.28}$$

These are linear differential equations that can be solved explicitly to find $\hat{\mathbf{x}}(t)$ and $P(t)$ given the initial conditions $\hat{\mathbf{x}}(0)$ and $P(0)$.

In the literature attention has been devoted to determining the distribution of $\mathbf{x}(t)$ explicitly using the Fokker-Planck equation, or its extensions, for systems that are more general than (A.25). In this thesis we are interested in switching systems with linear dynamics and non-Gaussian uncertainty:

$$d\mathbf{x}(t) = A(q,t)\mathbf{x}(t)dt + G(q,t)d\mathbf{w}(t). \tag{A.29}$$

Here $q \in \mathcal{Q}$ represents the discrete state, which switches at continuous time intervals according to a Markov transition distribution. The term $d\mathbf{w}(t)/dt$ is a noise process which *need not be Gaussian.* In addition, the initial state distribution $p(\mathbf{x}(0))$ need not be Gaussian. There are three key difficulties in applying the Fokker-Planck equation in this more general case. First, we may not be able to write down an explicit solution for $p(\mathbf{x}(t))$. In this case we have to resort to numerical solution of the resulting partial differential equation; however this is often intractable except when $\mathbf{x}(t)$ is of very low dimension. Finally, the Fokker-Planck equation is only defined for a class of random processes called *independent increment processes*, and we are concerned with general random processes. We now summarize the literature relating to these problems.

In the case of Gaussian noise, [13] describe the *generalized Fokker-Planck equation*, which applies to systems with both switching and continuous dynamics. For the case of a nonlinear, switching wind turbine model they are able to provide an approximate numerical solution for the stationary state distribution (rather than the full time-varying distribution), which they claim is the distribution of interest. Their approach is limited to models of dimension $n < 5$ by memory requirements and complexity issues; they note that Monte-Carlo approaches, on the other hand, can be used for $n > 5$ and are in fact more efficient for $n > 3$. In this thesis, we consider problems where n is between 40 and 160; hence this approach is unlikely to be of use.

For non-Gaussian noise processes that are independent increment processes, the Fokker-Planck equation is defined; examples of independent increment processes include Gaussian processes, Poisson processes and Lévy processes. In this case the Fokker-Planck equation is sometimes known as the *Kolmogorov-Feller* equation[49]. Direct numerical solution of this equation is challenging even for two-dimensional problems or smaller; for example [54] showed that the Fokker-Planck equation is impractical for finding the state distribution in the case of a single-variate linear system driven by Poisson noise or Lévy noise. Approximate methods have been derived by, for example, [74, 66, 116, 132], however these are restricted to small n.

We must also consider general non-Gaussian noise processes that are not independent increment processes. For the purposes of a rigorous mathematical definition, such processes may be considered the output of a nonlinear stochastic differential equation driven by Gaussian white noise $\nu(t)$:

$$\dot{\omega}(t) = f(\omega(t), t) + \nu(t). \tag{A.30}$$

Here $\omega(t)$ is a non-Gaussian noise process, which we now pass through the linear dynamics:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + \omega(t). \tag{A.31}$$

We can concatenate the linear and nonlinear systems to give a stochastic differential equation driven by Gaussian white noise. The important point is that the resulting differential equation is *nonlinear*, even though the system dynamics (A.31) are linear. Many authors have asserted that, in the nonlinear case, explicit solutions of the Fokker-Planck equation are not known, for example [136, 100, 46]. In fact, even determining statistics such as the mean and covariance is intractable in the general case, due to the *moment closure problem*; this means that, in order to calculate any statistic exactly we need the entire distribution. Again, approximate solutions exist, but these are limited to lower-order problems.

In summary, continuous-time propagation of probability distributions through dynamic systems is a very challenging problem, except in special cases. In this thesis we are concerned with hybrid discrete-continuous systems with arbitrary uncertainty distributions, and a literature search revealed that, despite recent activity in the area, the problem of distribution propagation has not been solved for this case. This, along with the reasons given at the beginning of Section A.2 motivates the use of a discrete-time system approximation in the thesis.

A.3 Evaluating and Constraining Probabilities

In this section we consider the evaluation of probabilities of events of interest given a probability distribution for the system state sequence. In this thesis, the events of interest consist of whether a set of episodes fails or not. The probability of a set of activities failing is given by:

$$p\Big(\mathcal{A}(c_c)\text{ fails}\Big) = \int_{\mathbf{x}_{1:N_p}} s\Big(\mathcal{A}(c_c), \mathbf{x}_{1:N_p}, T\Big) p(\mathbf{x}_{1:N_p}) d\mathbf{x}_{1:N_p}, \tag{A.32}$$

where $s(\cdot)$ is the indicator function defined in (2.4); we repeat the definition here:

$$s(\mathcal{A}, X_{0:f}, T) = \begin{cases} 0 & \text{All episodes } a \in \mathcal{A} \text{ succeed for state sequence } X_{0:f} \text{ and schedule } T \\ 1 & \text{There exists an episode } a \in \mathcal{A} \text{ that fails for state sequence } X_{0:f} \text{ and schedule } T. \end{cases} \tag{A.33}$$

In general, the function $s(\cdot)$ is unity for certain regions of the state space, and zero for all others. Let us define $\mathbb{D}$ as the region where $s(\mathcal{A}, X_{0:f}, T) = 1$. Then the probability of failure (A.32) can be expressed equivalently as:

$$p\Big(\mathcal{A}(c_c)\text{ fails}\Big) = \int_{\mathbb{D}} p(\mathbf{x}_{1:N_p}) d\mathbf{x}_{1:N_p}. \tag{A.34}$$

This is now a finite integral over the distribution $p(\mathbf{x}_{1:N_p})$, whereas (A.32) is an integral over the entire state space (ignoring the special case when $s(\mathcal{A}, X_{0:f}, T) = 1$ for all $\mathbf{x}_{1:N_p}$). This integral cannot be evaluated in closed form, except when $p(\mathbf{x}_{1:N_p})$ takes on very special forms, such as being uniform. For example, it is well known that the following integral does not admit a closed-form solution:

$$I = \int_a^b \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx. \tag{A.35}$$

In other words, even in the simple case of a univariate Gaussian distribution, which for most other purposes has a convenient form, the finite integral cannot be performed.

The robust execution problem considered in this thesis (Def. 26) performs an optimization over a control input sequence with constraints on the probability (A.32). We must approximate this value, since for almost all distributions it cannot be evaluated in closed form. There are two common approaches to approximating the integral in (A.34); first, numerical integration and second, Monte-Carlo approaches. Numerical integration typically works by splitting the integration region into finite subregions, and approximating the integrand in each region as a polynomial; in the case of the *trapezoidal rule* this polynomial is affine. The subregions are chosen to be of a form that makes evaluation of the approximated integral for the subregion straightforward; by performing this approximate integration for each subregion we obtain the approximation of the entire integral.

Monte-Carlo approaches work by sampling the uncertain distribution and evaluating the function $s(\cdot)$ at each sampled value. Since (A.32) is an expectation of $s(\cdot)$ taken over the distribution $p(\mathbf{x}_{1:N_p})$, we have convergence of the sample mean of $s(\cdot)$ to the true expectation as the number of samples approaches infinity. We described this convergence in detail in Section 3.3.

In Section A.4 we provide a number of reasons for choosing to use Monte-Carlo approaches, rather than numerical integration approaches, in this thesis.

A.4 Motivation for Monte-Carlo Methods

In this section we motivate the use of Monte-Carlo approaches in approximating the robust execution problem. First, consider approximation of the robust execution problem *without* using Monte-Carlo methods. We showed in Section A.1 that, for linear discrete-time system dynamics of the form (A.13) and arbitrary noise distributions, the system state sequence distribution can be calculated in closed form. Given this distribution, we can approximate the probability of episode failure (A.32) using numerical integration. Note, however, that our goal is to *constrain* the probability (A.32), rather than simply to evaluate it. In principle it would be possible to use a numerical evaluation of (A.32) in a constrained nonlinear optimization routine. The resulting constraints would have a particularly inconvenient form since they can only be evaluated numerically, and are are not, in general, convex. This means we cannot use any particular structure to simplify the optimization problem. Constrained nonlinear optimization approaches typically deal with such constraint using *barrier methods.* These methods penalize violation of constraints in the cost function, thereby approximating the constrained optimization problem as an unconstrained one. Without convexity of the constraints and the cost function, we cannot guarantee convergence to the global optimum of the approximated problem; we also cannot guarantee feasibility of the returned solution. Furthermore, numerical integration must be performed at each iteration of the optimization process, and each one of the numerical integrations is highly computationally intensive.

Monte-Carlo methods, on the other hand, are promising for three reasons. First, propagating the approximate distribution through the system dynamics is trivial; we simply propagate each sample in a deterministic manner. This is in constrast to propagation of the exact distribution, which requires the inversion of a $n_x \cdot (N_p + 1) \times n_x \cdot (N_p + 1)$ matrix. Second, a Monte-Carlo approximation of the robust execution problem leads to an optimization that can be encoded as a Mixed Integer Linear Program, as we show in the thesis. This ensures that the global optimum of the approximated problem can be found in bounded time. We show empirically in this thesis that this optimum can be found in seconds for several autonomous vehicle scenarios for a reasonable number of particles. Finally, Monte-Carlo methods can deal with a more general forms of uncertainty than the additive noise in (A.13). For example, multiplicative uncertainty, or uncertainty in the system matrices A and B can be handled trivially; we simply sample from all of these distributions. For these reasons, we choose Monte-Carlo approximation of the probability of episode failure in approximating the robust execution problem; this leads to the Particle Control approach described in this thesis.

Bibliography

[1] US military handbook MIL-HDBK-1797, 1997.

[2] F. Ababsa, M. Mallem, and D. Roussel. Comparison between particle filter approach and kalman filter-based technique for head tracking in augmented reality systems. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1021–1026, 2004.

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

[4] J. F. Allen. A general model of action and time. *Artificial Intelligence*, 23(2), 1984.

[5] C. Andrieu, A. Doucet, S. S. Singh, and V. B. Tadic. Particle methods for change detection, system identification, and control. *Proceedings of the IEEE*, 92, 2004.

[6] M. Aoki. *Optimization of Stochastic Systems.* Academic Press, 1967.

[7] A. C. Atkinson and D. R. Cox. Planning experiments for discriminating between models. *Journal of the Royal Statistical Society, Series B*, 36(3):321–348, 1974.

[8] A. Aurnhammer and K. Marti. *Dynamic Stochastic Optimization*, pages 133–154. Springer, Berlin, 2003.

[9] H. Balakrishnan, I. Hwang, J. Jang, and C. Tomlin. Inference methods for autonomous stochastic linear hybrid systems. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, *Hybrid Systems: Computation and Control, HSCC 2007*, volume 2993 of *Lecture Notes in Computer Science*, pages 64–79. Springer Verlag, 2004.

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

[11] N. M. Barr, D. Gangsaas, and D. R. Schaeffer. Wind models for flight simulator certification of landing and approach guidance and control systems. Paper FAA-RD-74-206, 1974.

[12] I. Batina. *Model Predictive Control for Stochastic Systems by Randomized Algorithms.* PhD thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2004.

[13] J. Bect, Y. Phulpin, H. Baili, and G. Fleury. On the fokker-planck equation for stochastic hybrid systems: Application to a wind turbine model. In *Proceedings of the 9th International Conference on Probabilistic Methods Applied to Power Systems*, 2006.

[14] John Bellingham, Arthur Richards, and Jonathan How. Receding horizon control of autonomous aerial vehicles. In *Proceedings of the American Control Conference*, 2002.

[15] A. Bemporad and M. Morari. Robust model predictive control: A survey. In A. Tesi A. Garulli and A. Vechino, editors, *In Robustness in Identification and Control*, pages 207–226. Springer-Verlag, 1999.

[16] D. P. Bertsekas. *Dynamic Programming and Stochastic Control.* Academic Press, 1976.

[17] D. P. Bertsekas and J. N. Tsitsiklis. *Introduction to Probability.* Athena Scientific, Belmont, MA, 2002.

[18] L. Blackmore, A. Bektassov, M. Ono, and B. C. Williams. Robust, optimal predictive control of jump markov linear systems using particles. In A. Bemporad, A. Bicchi, and G. Buttazzo, editors, *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science, 4416, pages 104–117. Springer-Verlag, 2007.

[19] L. Blackmore, S. Funiak, and B. C. Williams. Combining stochastic and greedy search in hybrid estimation. In *Proc. of 20th National Conference on Artificial Intelligence (AAAI05)*, 2005.

[20] 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.

[21] A. Blum and M. Furst. Fast planning through planning graph analysis. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1995.

[22] E. K. Boukas, P. Shi, and K. Benjelloun. Robust stochastic stabilization of discrete-time linear systems with markovian jumping parameters. *Journal of Dynamic Systems, Measurement, and Control*, 121:331–334, 1999.

[23] M. Campbell and J. Ousingsawat. On-line estimation and path planning for multiple vehicles in an uncertain environment. In *Proc. AIAA GNC Conference*, 2002.

[24] D. W. Caress and D. N. Chayes. Mapping the seafloor: Software for the processing and display of swath sonar data. http://www.ldeo.columbia.edu/res/pi/MB-System/, 2007.

[25] C. B. Chang and M. Athans. State estimation for discrete systems with switching parameters. *IEEE Transactions on Aerospace and Electronic Systems*, AES-14:418–424, May 1978.

[26] J. Chen and R. Patton. *Robust Model-Based Fault Diagnosis for Dynamic Systems.* Kluwer Academic, Boston, 1999.

[27] S. Cheng and P. N. Sabes. Modeling sensorimotor learning with linear dynamical systems. *Neural Computation*, 18(4):760–793, 2006.

[28] O. L. V. Costa, M. D. Fragoso, and R. P. Marques. *Discrete-Time Markovian Jump Linear Systems.* Springer-Verlag, 2005.

[29] F. Dallaert. The expectation maximization algorithm. Technical Report GIT-GVU-02-20, College of Computing, Georgia Insitute of Technology, 2002.

[30] J. de Kleer and B. C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.

[31] 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.

[32] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artificial Intelligence Journal*, 1991.

[33] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artif. Intell.*, 49(1-3):61–95, 1991.

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

[35] Carmel Domshlak and Jörg Hoffmann. Fast probabilistic planning through weighted model counting. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS-06)*, 2006.

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

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

[38] 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.

[39] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000.

[40] R. Duda, P. Hart, and D. Stork. *Pattern Classification (2nd ed.).* Wiley Interscience, 2000.

[41] V. Fedorov. *Theory of Optimal Experiments.* Academic Press, New York, 1972.

[42] K. Felsenstein. Optimal bayesian design for discrimination among rival models. *Computational Statistics and Data Analysis*, 14:427–436, 1992.

[43] G. F. Franklin, J. D. Powell, and M. L. Workman. *Digital Control of Dynamic Systems.* Addison-Wesley, 1990.

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

[45] 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.

[46] C. W. Gardiner. *Handbook of Stochatic Methods.* Springer, 2004.

[47] Zoubin Ghahramani and Geoffrey E. Hinton. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2, Department of Computer Science, University of Toronto, Feb 1996.

[48] Zoubin Ghahramani and Geoffrey E. Hinton. Variational learning for switching state-space models. *Neur. Comp.*, 12(4):831–864, 2000.

[49] I. I. Gihman and A. V. Skorohod. *Stochastic Differential Equations.* Springer Verlag, 1972.

[50] G. C. Goodwin and R. L. Payne. *Dynamic System Identification: Experiment Design and Data Analysis.* Academic Press, New York, 1977.

[51] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings - F*, 140(2):107–113, April 1993.

[52] J. Gossner, B. Kouvaritakis, and J. Rossiter. Stable generalized predictive control with constraints and bounded disturbances. *Automatica*, 33:551, 1997.

[53] A. Greenfield and A. Brockwell. Adaptive control of nonlinear stochastic systems by particle filtering. In *Proceedings of the Conference on Control and Automation*, 2003.

[54] M. Grigoriu. Characteristic function equations for the state of dynamic systems with gaussian, poisson, and lévy white noise. *Probabilistic Engineering Mechanics*, pages 449–461, 2004.

[55] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes.* Oxford University Press, 1982.

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

[57] Patrick J. Hayes. The second naive physics manifesto. *Computation & intelligence: collected readings*, pages 567–585, 1995.

[58] P. Hellekalek and G. Larcher (Eds.). *Random and Quasi-Random Point Sets.* Lecture Notes in Statistics. Springer, 1998.

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

[60] D. H. Van Hessem. *Stochastic Inequality Constrained Closed-loop Model Predictive Control.* PhD thesis, TU Delft, Delft, The Netherlands, 2004.

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

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

[63] A. Hofmann. *Robust Execution of Bipedal Walking Tasks from Biomechanical Principles.* PhD thesis, Computer Science and Artificial Intelligence Laboratory, MIT, 2006.

[64] X. Hu and T. Ersson. Active state estimation of nonlinear systems. *automatica*, 40:2075–2082, 2004.

[65] 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.

[66] R. Iwankievicz and S. R. K. Nielsen. Dynamic response of non-linear systems to poisson distributed random impulses. *Journal of Sound and Vibration*, pages 407–423, 1992.

[67] Ali A. Jalali and Vahid Nadimi. A survey on robust model predictive control from 1999-2006, 2006.

[68] J. Jang and C. Tomlin. Longitudinal stability augmentation system design of the stanford dragonfly uav using a single gps receiver. In *Proceedings of the AIAA GNC conference*, Austin, Texas, August 2003.

[69] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. Technical Report CS-96-08, Brown University, 1996.

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

[71] F. Kerestecioglu. *Change Detection and Input Design in Dynamical Systems.* Wiley, New York, 1993.

[72] S.-J. Kim and R. A. Iltis. Performance comparison of particle and extended kalman filter algorithms for gps c/a code tracking and interference rejection. In *Proceedings of Conference Information Systems Sciences*, 2002.

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

[74] H. U. Koyluoglu, R. K. Nielsen, and R. Iwankievicz. Response and reliability of poisson-driven systems by path integration. *Journal of Engineering Mechanics*, pages 117–130, 1995.

[75] Nicholas Kushmerick, Steve Hanks, and Daniel S. Weld. An algorithm for probabilistic planning. *Artificial Intelligence*, 76(1-2):239–286, 1995.

[76] T. Léauté. Coordinating agile systems through the model-based execution of temporal plans. SM thesis, Massachusetts Institute of Technology, 2005.

[77] A. Lecchini, W. Glover, J. Lygeros, and J. Maciejowski. Monte carlo optimisation for conflict resultion in air traffic control. *IEEE Transactions on Intelligent Transportation Systems*, 2006.

[78] A. Lecchini, W. Glover, J. Lygeros, and J. Maciejowski. Monte carlo optimization for conflict resolution in air traffic control. *IEEE Transactions on Intelligent Transportation Systems*, 2006. To appear.

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

[80] 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.

[81] 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.

[82] P. Li, M. Wendt, and G. Wozny. Robust model predictive control under chance constraints. *Computers and Chemical Engineering*, 24:829–834, 2000.

[83] P. Li, M. Wendt, and G. Wozny. A probabilistically constrained model predictive controller. *Automatica*, 38:1171–1176, 2002.

[84] S. M. Majercik and M. L. Littman. Contingent planning under uncertainty via stochastic satisfiability. *Artificial Intelligence*, 147(1-2):119–162, 2003.

[85] Stephen M. Majercik and Michael L. Littman. MAXPLAN: A new approach to probabilistic planning. In *Artificial Intelligence Planning Systems*, pages 86–93, 1998.

[86] R. McEwen and K. Streitlien. Modeling and control of a variable-length auv. In *Proc. Unmanned Untethered Submersible Technology Conference*, Durham, NH, August 2001.

[87] S. McIlraith and R. Reiter. On tests for hypothetical reasoning, 1992.

[88] D. McRuer, I. Ashkenas, and D. Graham. *Aircraft Dynamics and Automatic Control.* Princeton University Press, 1990.

[89] N. Megiddo. On the complexity of linear programming. In T. Bewley, editor, *Advances in Economic Theory: Fifth World Congress*, pages 225–268. Cambridge University Press, 1987.

[90] N. Metropolis and S. Ulam. The monte carlo method. *American Statistical Association*, 44:335–341, 1949.

[91] T. Minka. Expectation-maximization as lower bound maximization, 1998.

[92] Ruben Morales-Menendez, 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.

[93] 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.

[94] S. Narasimhan, G. Biswas, G. Karasai, and F. Zhao. Building observers to address fault isolation and control problems in hybrid dynamic systems. In *Proceedings of the IEEE Internation Conference on Systems, Man, and Cybernetics (SMC 2000)*, pages 2393–2398, 2000.

[95] R. Neal and G. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models.* Kluwer, 1998.

[96] A. Y. Ng and M. Jordan. Pegasus: A policy search method for large mdps and pomdps. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000.

[97] R. Nikoukhah, S. L. Campbell, and F. Delebecque. Detection signal design for failure detection: a robust approach. *Int. J Adaptive control and Signal Processing*, 14:701–724, 2000.

[98] R. Nikoukhah, S. L. Campbell, K. Horton, and F. Delebecque. Auxiliary signal design for robust multi-model identification. *IEEE Trans. Automat. Contr.*, 47:158–163, 2002.

[99] J. Nocedal and S. Wright. *Numerical Optimization.* Springer, 1999.

[100] B. Œksendal. *Stochastic Differential Equations.* Springer, 2003.

[101] S. M. Oh, J. M. Rehg, T. Balch, and F. Dallaert. Learning and inference and parametric switching linear dynamic systems. In *Proceedings of IEEE International Conference on Computer Vision*, 2005.

[102] P. Pardalos and J. Rosen. Methods for global concave minimization: A bibliographic survey. Siam review, Society for Industrial and Applied Mathematics, September 1986.

[103] R. J. Patton, P. M. Frank, and R. N. Clark, editors. *Issues of Fault Diagnosis for Dynamic Systems.* Springer, New York, 2002.

[104] V. Pavlovic, J. Rehg, T.-J. Cham, and K. Murphy. A dynamic Bayesian networ approach to figure tracking using learned dynamic models. In *Proceedings of ICCV*, 1999.

[105] H. Pham. On some recent aspects of stochastic control and their applications. *Probability Surveys*, 2:506–549, 2005.

[106] A. Pouliezos and G. Stavrakakis. *Real Time Fault Monitoring of Industrial Processes.* Kluwer Academic, Boston, 1994.

[107] A. Prekopa. *Stochastic Programming.* Kluwer, 1995.

[108] J. G. Proakis. *Digital Communications.* New York, 1995.

[109] M. L. Puterman. *Markov Decision Processes.* Wiley, 1994.

[110] A. Richards. *Robust Constrained Model Predictive Control.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2005.

[111] A. Richards, J. How, T. Schouwenaars, and E. Feron. Plume avoidance maneuver planning using mixed integer linear programming. In *Proc. AIAA Guidance, Navigational and Control Conference*, 2001.

[112] A. Richards and J. P. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the American Control Conference*, pages 1936–1941, 2002.

[113] A. Richards, T. Schouwenaars, J. P. How, and E. Feron. Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming. *Journal of Guidance, Control and Dynamics*, 25(4), 2002.

[114] H. Robbins and S. Munro. A stochastic approximation method. *Ann. Math. State*, 22:400–507, 1951.

[115] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods.* Springer-Verlag, 1999.

[116] J. B. Roberts. System response to random impulses. *Journal of Sound and Vibration*, pages 23–34, 1972.

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

[118] T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *Proc. European Control Conference*, 2001.

[119] A. Schwarm and M. Nikolaou. Chance-constrained model predictive control. *AlChE Journal*, 45:1743–1752, 1999.

[120] L. G. Shapiro and G. C. Stockman. *Computer Vision.* Prentice-Hall, 2001.

[121] R. H. Shumway and D. S. Stoffer. An approach to time series smoothing and forecasting using the em algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[122] M. Simandl, I. Puncochar, and J. Kralovec. Rolling horizon for active fault detection. In *Proc. Control and Decision Conference*, Seville, Spain, 2005.

[123] S. S. Singh, N. Kantas, B. Vo, A. Doucet, and R. Evans. Simulation-based optimal sensor scheduling with application to observer trajectory planning. *Automatica*, 43:817–830, 2007.

[124] D. E. Smith and D. S. Weld. Conformant graphplan. In *Proc. 15th National Conf. AI*, 1998.

[125] J. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control.* Wiley, 2003.

[126] Peter Struss. Testing physical systems. In *National Conference on Artificial Intelligence*, pages 251–256, 1994.

[127] S. A. Tarim, S. Manandhar, and T. Walsh. Stochastic constraint programming: A scenario-based approach. *Constraints*, 11:53–80, 2006.

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

[129] Simon Tong and Daphne Koller. Active learning for parameter estimation in bayesian networks. In *NIPS*, pages 647–653, 2000.

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

[131] A. N. Vargas, W. Furloni, and J. B. R. do Val. Constrained model predictive control of jump linear systems with noise and non-observed markov state. In *Proceedings of the American Control Conference*, 2006.

[132] M. Vasta and A. Luongo. Dynamic analysis of linear and nonlinear oscillations of a beam under axial and transversal random poission pulses. *Nonlinear Dynamics*, pages 421–435, 2004.

[133] B. C. Williams, M. Ingham, S. H. Chung, and P. H. Elliott. Model-based programming of intelligent embedded systems and robotic space explorers. *Proceedings of the IEEE*, 91(1):212–237, 2003.

[134] B. C. Williams and P. Pandurang. A reactive planner for a model-based executive. In *Proceedings of the National Conference on Artificial Intelligence*, pages 971–978, 1996.

[135] A. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12:601–611, 1976.

[136] A. S. Willsky. 6.433 recursive estimation lecture notes, spring 1994.

[137] L. Yu, X. Ji, and S. Wang. Stochastic programming models in financial optimization: A survey. *Advanced Modeling and Optimization*, 5(1), 2003.

[138] X. J. Zhang. Auxiliary signal design in fault detection and diagnosis. *NASA STI/Recon Technical Report A*, 90:24257–+, 1989.