Efficient Hierarchical Robot Motion Planning Under Uncertainty and Hybrid Dynamics

Ajinkya Jain
Department of Mechanical Engineering,
University of Texas at Austin,
Austin, TX 78712, USA
Email: ajinkya@utexas.edu

Scott Niekum
Department of Computer Science,
University of Texas at Austin,
Austin, TX 78712, USA
Email: sniekum@cs.utexas.edu

Abstract—Noisy observations coupled with nonlinear dynamics pose one of the biggest challenges in robot motion planning. By decomposing the nonlinear dynamics into a discrete set of local dynamics models, hybrid dynamics provide a natural way to model nonlinear dynamics, especially in systems with sudden “jumps” in the dynamics, due to factors such as contacts. We propose a hierarchical POMDP planner that develops locally optimal motion plans for hybrid dynamics models [1]. The hierarchical planner first develops a high-level motion plan to sequence the local dynamics models to be visited. The high-level plan is then converted into a detailed cost-optimized continuous state plan. This hierarchical planning approach results in a decomposition of the POMDP planning problem into smaller sub-parts that can be solved with significantly lower computational costs. The ability to sequence the visitation of local dynamics models also provides a powerful way to leverage the hybrid dynamics to reduce state uncertainty. We evaluate the proposed planner for two navigation and localization tasks in simulated domains, as well as an assembly task with a real robotic manipulator.

I. INTRODUCTION

One of the biggest challenges in robot motion planning is to develop feasible motion plans for systems having highly nonlinear dynamics in the presence of partial or noisy observations. Often, these nonlinearities are caused by sudden transitions or “jumps” in the dynamics (for example, due to contacts in a robot manipulation task). When the dynamics of a task can change suddenly in state space, even small state estimation errors can lead to large deviations and plan failure. Therefore, reasoning about the uncertainty over states becomes crucial in order to develop robust motion plans. Planning problems under uncertainty are often represented as a partially observable Markov decision process (POMDP) [2]. POMDP problems have been shown in literature to be PSPACE-complete [3], making exact planning intractable.

To make planning tractable, POMDP planners typically leverage various types of approximations [4, 5, 6, 7, 8] or structural assumptions [7, 9, 10, 11, 12] that simplify the problem. In this work, we propose to leverage a natural, simplifying assumption that the nonlinear dynamics of robot motion planning tasks can be decomposed into a discrete set of simpler local dynamics models, of which only one is active at any given time (e.g. a change in dynamics due to contact). Note that these local dynamics models may be approximate, especially when they are learned from data or are a simplification of a complex underlying model. A complete dynamics model can then be defined as a hybrid dynamics model having hybrid states comprised of the continuous states of the system along with a discrete state denoting the active local dynamics model.

The primary contribution of this work is a novel POMDP planner that plans in a hybrid belief space, allowing for efficient information gathering and planning under uncertainty with hybrid dynamics. We define the hybrid belief to be composed of a mixture of Gaussians and a discrete distribution, which represent the uncertainty over the state of the robot and the active dynamics model respectively. Using the hybrid belief representation, a hierarchical POMDP motion planner is presented that solves the POMDP problem by dividing it into two levels: at the higher level, discrete state plans are generated to find a sequence of local models that should be visited during the task, and at the lower level, these discrete state plans are converted into cost-optimized continuous state belief-space plans.

The biggest advantage of dividing the planning problem into two levels is that it breaks long-horizon planning problems into multiple smaller segments that can be sequenced to find a complete solution. Since POMDP planning becomes exponentially more difficult with longer horizons ([3]), a hierarchical approach breaks problem into chunks that can be solved with significantly less effort. Another major benefit of discrete state planning is that the planner can chose to leverage a specific local dynamics model in order to improve the effectiveness of the generated plans. For example, if it is known a priori that in the k-th local dynamics model, motion is allowed only along a particular vector (e.g. due to presence of a wall), it can be used to reduce the state uncertainty along the dimensions orthogonal to the allowed motion vector. This indirect feedback for uncertainty reduction is critical for tasks in which the observations are either extremely noisy or not available at all.

The proposed POMDP motion planner is evaluated on two simulated domains: autonomous robot navigation under uncertainty with spatially varying dynamics and planar navigation while localizing using walls. Finally, our algorithm is evaluated with a physical robotic manipulator that is tasked with partially assembling a toy airplane from the YCB dataset [13].
II. RELATED WORKS

A. POMDP Planning

Broadly, POMDP solving approaches can be divided into two categories based on whether their state, action and observation spaces are discrete or continuous. Discrete space POMDP solvers, in general, either approximate the value function using point-based methods [5, 6] or use Monte-Carlo sampling in the belief space [14, 15] to make the POMDP problem tractable. Continuous space POMDP solvers often approximate the belief over states as a distribution having finite parameters (typically Gaussian) and either solve the problem analytically using gradients [10, 11, 12] or use random sampling in the belief space [7, 8]. Other approaches have also extended point-based methods to continuous domains [6].

Discrete space POMDP solvers have been shown to be able to successfully plan for large discrete space domains, however, continuous space domains are infinite-dimensional, and discrete space solvers often fail to find feasible solutions for planning horizons longer than a few steps [15]. Among continuous space POMDP solvers, Agha-Mohammadi et al. [7] and Hollinger and Sukhatme [8] have proposed sampling based methods that can find effective solutions even in complex domains. However, they suffer from the problem of obtaining sub-optimal solutions which can only be probabilistically optimal at best [16]. Gradient-based POMDP solvers [10, 11, 12] form another class of very powerful POMDP solvers which can find locally optimal solutions, but in the context of manipulation planning, sudden changes in dynamics due to contacts result in non-finite gradients at the transition points and restrict the applicability of such methods.

POMDP solvers for hybrid domains, such as the one discussed in this work, have been previously discussed by Brunskill et al. [4], Sreenath et al. [9] and Agha-mohammadi et al. [17]. Brunskill et al. [4] proposed a point-based POMDP planning algorithm, SM-POMDP planner, for solving continuous-state POMDPs based on the hybrid system dynamics. They approximated the complex nonlinear system dynamics using a hybrid multi-modal dynamics model with continuous state-dependent discrete mode switching conditions. However, unlike our POMDP planner, SM-POMDP planner plans only in the continuous domain and the discrete states are obtained “passively” using the switching conditions. While this approach can be used to find feasible motion plans, it is not leveraging some of the major natural advantages of the hybrid dynamics representation such as shorter planning horizons and a structured way to leverage dynamics for state uncertainty reduction. Sreenath et al. [9] discussed the problem of bipedal walking on a varying terrain by formulating it as a POMDP problem defined on a continuous-time hybrid system. They proposed a bi-level POMDP controller to track the transitions in the terrain as a set of discrete states and were able to show stable bipedal walking in simulated domains. However, this is a passive approach as well, as it uses hybrid dynamics only to capture the transitions in the terrain and not to simplify the POMDP problem. Agha-mohammadi et al. [17] discussed a POMDP solver with hybrid states to solve health-aware stochastic motion planning problem for quadrotors, however, the proposed solution is restricted only to the domains in which the discrete and continuous states evolve independently.

B. Manipulation Planning and Control

As hybrid dynamics models are very effective in modeling nonlinearities that are due to sudden transitions in the dynamics, a natural application domain for the proposed POMDP solver is contact-rich robot manipulation. One of the current approaches for solving the robot manipulation planning problem is to search for an optimal sequence of parameterized manipulation actions or primitives to perform the task [18, 19]. Kroemer et al. [19] have proposed to represent primitives for different phases (modes) of a multi-phase manipulation task using dynamic movement primitives (DMPs) and learn a library of such manipulation skills which can be optimally sequenced to perform a task. Unfortunately, a lack of a task dynamics model prevents these methods from generalizing to novel manipulation tasks, e.g. having different cost functions, even if it involves the same objects.

In more recent works by Levine et al. [20] and Fu et al. [21], authors have used deep learning techniques to develop end-to-end control policies directly from vision; however, under sparse availability of training data (especially in robotics), these approaches tend to fail to develop generalized control policies for all system states or initial conditions.

III. PRELIMINARIES AND DEFINITIONS

A. POMDPs

Partially Observable Markov Decision Processes (POMDPs) provide a mathematical framework for the problem of sequential decision making under uncertainty [2]. Let $X \subset \mathbb{R}^n$ be the space of all possible states $x$ of the robot, $U \subset \mathbb{R}^m$ be the space of all possible control inputs $u$ and $Z \subset \mathbb{R}^k$ be the space of all possible sensor measurements $z$ the robot may receive. To account for state uncertainty, a distribution of the state $x_t$ of the robot given all past control inputs and sensor measurements is defined as the belief $b(x_t)$, given as

$$b[x_t] = p[x_t | u_0, ..., u_{t-1}, z_1, ..., z_t]$$

where $x_t \in X$, $u_t \in U$ and $z_t \in Z$ are the robot’s state, control input and received measurement at time step $t$, respectively and $B \subset \{X \rightarrow \mathbb{R}\}$ represent the space of all possible beliefs.

In a general case, considering a stochastic dynamics and observation model for the process given as

$$x_{t+1} \sim p[x_{t+1} | x_t, u_t], \quad z_t \sim p[z_t | x_t]$$

for a given control input $u_t$ and a measurement $z_{t+1}$, the belief can be propagated using Bayesian filtering as

$$b[x_{t+1}] = \eta p[z_{t+1} | x_{t+1}] \int p[x_{t+1} | x_t, u_t] b[x_t] dx_t$$

where $\eta$ is a normalizing constant independent of $x_{t+1}$.

In this work, we consider hybrid beliefs to represent the
distribution of hybrid states, with the beliefs over continuous
states represented as a mixture of Gaussians and a discrete
distribution that represents confidence over the active local
dynamics model. Further details on belief propagation are
discussed in the section [IV]

B. Trajectory Optimization using Direct Transcription

Direct Transcription is a trajectory optimization method
in which a constrained nonlinear optimization problem is
set up with the user-defined objective function over a set
of knot-points \( \{x_i, u_i\} \) chosen to discretize the continuous
space trajectory into a set of decision variables. The system
dynamics are imposed as the constraints on the optimization
problem. For discrete-time systems, these knot-points can
be taken as the system state \( x_t \) and the control input \( u_t \)
at each time step \( t \). However, planning for longer horizons
will then require specifying a high number of knot-points
\( \{x_i, u_i\} \) which can result in very high computational
costs. This can be resolved by approximately parameterizing
the space of possible trajectories by a series of \( M \) segments
and solving the optimization problem for a knot points only at
the start and end points of segments. The intermediate points
on the trajectory can be obtained via numerical integration.
Let \( x'_{1:M} \) and \( u'_{1:M-1} \) be sets of state and action variables
that parameterize the trajectory in terms of segments. The \( i^{th} \)
segment can be assumed to start at time \( i\delta \) and ends at time
\( i\delta + \delta - 1 \), where \( \delta = \frac{T}{M} \) for a time horizon \( T \).

A general objective function for trajectory optimization can be
given as

\[
J(x_{1:T}, u_{1:T}) \approx J(x'1 : M, u'1:M) = \sum_{j=1}^{M} x'^{T}_j Q x'_j + u'^{T}_j R u'_j
\]

where \( Q \) and \( R \) represent the cost matrices associated with
the state and the input respectively. The system dynamics
incorporated as constraints can be defined as:

\[
x'_2 = \phi(x'_1, u'_1)
\]

\[
\vdots
\]

\[
x'_{k} = \phi(x'_{k-1}, u'_{k-1})
\]

where the function \( \phi(x'_i, u'_i) \) can be seen as performing
numerical integration of the current state variable \( x'_i \) till the next
state variable \( x'_{i+1} \). The function \( \phi \) is given as

\[
x'_{i+1} = \phi(x'_i) = F(x'_i, u_i)
\]

\[
+ \sum_{l=i\delta}^{i\delta+\delta-1} [F(x_{l+1}, u_l) - F(x_l, u_l)]
\]

where \( F(x, u) \) represents the system dynamics.

C. Hybrid Dynamics

A hybrid dynamics model of a system is a dynamics model
in which the states of the system evolve with time over both
continuous space \( x \in X = \mathbb{R}^N \) and a finite set of discrete
states \( q \in Q \subset \mathbb{W} \). Each discrete state of the system
corresponds to a separate dynamics model that governs the
evolution of continuous states. These types of dynamical
models are sometimes referred to as switched dynamical
systems in the literature [22].

In a hybrid model, discrete state transitions of the system
can be represented as a directed graph with each possible
discrete state \( q \) corresponding to a node and edges \( (e \in E \subseteq Q \times Q) \) marking possible transitions between the nodes.
These transitions are conditioned on the continuous states. A
transition from the discrete state \( q \) to another state \( q' \) happens
if the continuous states \( x \) are in the guard set \( G(q, q') \) of the
dege \( e_q^q \) where \( e_q^q = \{q, q'\} \), \( G(\cdot): E \rightarrow P(X) \) and \( P(X) \)
is the power set of \( X \). Thus, a hybrid dynamics model \( H \) can
be defined as

\[
x_{t+1} = f_q(x_t, u_t)
\]

\[
z_t = h_q(x_t)
\]

where \( x \in \mathbb{R}^n \), \( u \in \mathbb{R}^m \) and \( z \in \mathbb{R}^l \) are the continuous state,
control input and observation variables, respectively and \( q \in Q \)
represents the active discrete state of the system. Evolution of
the discrete state of the system can be modeled by a finite
state Markov chain as

\[
\mu_{t+1} = \Pi \mu_t
\]

where \( Q = \{\pi_{ij}\} \) is the discrete state transition matrix and
\( \mu \in \mathbb{R}^Q \) is the probability distribution over discrete states at
time \( t \).

IV. APPROACH

We propose to solve the problem of motion planning under
uncertainty for tasks governed by highly nonlinear dynamics as
a POMDP problem defined on a hybrid dynamics model. Dif-
ferent local dynamics models constituting the task dynamics
are represented as distinct discrete states of the hybrid model.
Under uncertainty over the robot state, a separate discrete
distribution needs to be maintained to represent our confidence
over the active local dynamics model at each time step. Jointly,
a hybrid belief over the hybrid state of the system can be
defined with a continuous part representing uncertainty over
the robot state and a discrete part representing uncertainty in
the active local dynamics model.

In this work, we assume that the continuous part of hybrid
belief is represented by a mixture of \( L \) Gaussian distributions
given as

\[
b^q_t = \sum_{l=1}^{L} \alpha_l N(\mu_t, \Sigma_l)
\]

where \( \alpha_t \) represents the mixing weight of \( l \)-th Gaussian.

A. Belief Propagation under Hybrid Dynamics

A hybrid belief can be defined as \( B = \{b^x, b^q\} \), where \( b^x \)
and \( b^q \) correspond to the belief over the continuous robot state
\( x \) and the discrete states (local dynamics models) \( q \) respec-
tively. Propagation of hybrid belief using Bayesian filtering can
be separated into two steps: prediction based on the dynamics
model to obtain a belief prior and an update step using the
received observation to find belief posterior.
1) Belief Prior: At each time step $t$, we can propagate the current belief $b^T_t$ through the system dynamics of each discrete state, $\mathcal{F}^q$ ($x_t, u_t$), individually and then take a weighted sum of the propagated belief set to obtain a belief prior for the next time step $\hat{b}^T_{t+1}$

$$\hat{b}^T_{t+1} = \sum_{q'} \mathcal{F}^q (b^T_t, u_t) b^{q'}_t$$  (8)

where $b^{q'}_t = p(q_t = q'|x_t)$ is $q'$-th component of $b^T_t$, and $x_t$, $q_t$ and $u_t$ represent the continuous states, discrete state and continuous control input to the system at time $t$ and $b|x(t+1)]$ is denoted as $\hat{b}^T_{t+1}$.

Under stochastic continuous state dynamics, the definition of the discrete state transition matrix as given in Equation 6 needs to be extended. Assuming the transitions of discrete states are given by a directed graph with self-loops, we can define the extended discrete state transition matrix $\Pi_t$ at time $t$ as $\Pi_t = \{\pi_t(i,j) = p(q_{t+1}|q_t, b^T_t) \forall q^i, q^j \in Q\}$ as

$$\pi_t(i,j) = \eta \int_{\mathbb{R}^N} 1^{q_t'}(x) b^{q_{t+1}}_t(x) dx, \text{ if } \exists e^{q_t'}_t = \epsilon, \text{ otherwise}$$  (9)

where $1^{q_t'}(x)$ is an indicator function defined as

$$1^{q_t'}(x) = \begin{cases} 1, & \text{if } x \in G(q^t, q^j) \\ 0, & \text{otherwise} \end{cases}$$  (10)

$\eta$ is a normalization constant, given as $\eta = \sum_i^{1|Q|} \pi_t(i,k)$ and $\epsilon$ is a small probability to handle scenarios in which the received observations do not correspond to any legal discrete transition based on the current belief.

Calculating the extended discrete state transition matrix $\Pi_t$ at each time step using Eq. 9 can be computationally expensive. An approximation of $\Pi_t$ can be obtained by sampling $n$ random points from the belief over continuous states $b^T_{t+1}$ and calculating ratio of points lying in the guard set $G(q^t, q^j)$ to the total number of sampled points for each discrete state $q^j$.

2) Belief Posterior: We use a hybrid estimation algorithm based on Bayesian filtering to reduce the uncertainty over states using noisy continuous state observations. The proposed algorithm consists of two layers of filters: first to estimate the continuous states of the system and second to estimate the discrete states of the system. Upon receiving observation $z_{t+1}$, the continuous state prior is updated by taking a weighted sum of a bank of extended Kalman filters running independently, with each discrete mode having an individual filter. The weights for the sum is determined using the prior for the discrete mode $\hat{b}^q_{t+1}$. The complete update step for continuous states can be written as

$$b^T_{t+1} = \hat{b}^T_{t+1} + \sum_{q^j} \left( K^j_{t+1}(z_{t+1} - H^j_{t+1}(\hat{b}^q_{t+1})) \right) b^{q^j}_{t+1}$$  (11)

where $K^j_{t+1}$ is the Kalman Gain for discrete state $q^j$ at time $t + 1$ and $b^{q^j}_{t+1}$ is $q^j$-th component of $\hat{b}^q_{t+1}$. The update for the discrete state can be obtained by using a Bayesian filter update given as

$$b^q_{t+1} = \gamma M_{t+1} \circ \hat{b}^q_{t+1}$$  (12)

where $M_{t+1} = [P(z_{t+1}|q_{t+1} = q^j)]^T \forall q^j \in Q$, $\circ$ is the element-wise multiplication operator, $\gamma$ is a normalization constant and

$$P(z_{t+1}|q_{t+1} = q^j) = z_{t+1} \sim H^j_{t+1}(b^q_{t+1})$$

where $H^j_{t+1}(.)$ is the observation function for state $q^j$.

Mixing weights for the mixture of Gaussians are also updated based on the received observations as

$$\alpha^j_{t+1} = \mathcal{N}(\nu|0, \Sigma^j_{t+1})$$  (13)

where innovation $\nu$ is given as $\nu = z_{t+1} - \hat{z}^j_{t+1}$ and

$$\hat{z}^j_{t+1} = \sum_{q^j} \mathcal{H}^j_{t+1}(\mu_t) \times \hat{b}^{q^j}_{t+1}$$

A new mixture of $L$ Gaussians is then chosen to represent the continuous belief $b^T_{t+1}$ at time step $t + 1$.

B. Direct Planning

With the hybrid belief propagation equations defined, we can now use the trajectory optimization technique to solve the POMDP problem. We assume maximum likely observations (MLO) obtained by propagating the current belief over continuous states through the system dynamics (Eqn. 8) as true observations for developing locally optimal motion plans.

This is a standard assumption to make while solving POMDP problems and has been discussed previously by Platt et al. [23]. In this work, the nonlinear optimization problem set up for trajectory optimization is posed as a sequential least squares programming (SLSQP) problem and solved using the SNOPT software package [24, 25]. We denote this approach as the direct planning approach.

C. Hierarchical Planner

Although the direct planning approach can be used to solve the POMDP problem, planning for longer horizons in complex tasks, such as contact-rich manipulation tasks, can result in infeasible computational costs [3]. To tackle this challenge, we propose a hierarchical planner that decomposes the POMDP problem into smaller subproblems which can be solved with significantly less effort. The proposed hierarchical planner has two levels: a higher level to find the best sequence of local dynamics models that should be visited along the path (by visiting the region in the continuous state space corresponding to its guard set, $G(q^j, \cdot)$) and a lower level that is similar to the direct planning approach discussed above. The higher level planner starts by generating a set of high-level plans consisting of all possible permutations of the discrete states of the task. A high-level plan is then converted into a sequence of continuous state goals which represent the most likely points in the continuous state space for activating the corresponding set of discrete states (see Algorithm 1). The lower level planner is then called for each of these continuous state
goals and a complete continuous state path for the high-
level plan is generated by combining the outputs of lower
level planner. An additional discrete state is added to each
high-level plan which represents the desired goal of the task
and is considered to be active within an $\epsilon$–neighbourhood
of the actual task goal. High-level plans are then ranked by
calculating a divergence cost on the distribution of planner’s
confidence on the active discrete state at the final point of
the plan and the desired confidence distribution (all the
probability mass within the $\epsilon$–neighbourhood of the goal).
Continuous state plan corresponding to the high-level plan
with the minimum cost is chosen as the best path for the task.
In this work, we have used $\text{Hellinger distance}$ to calculate
the divergence cost $[26]$ between the discrete distributions as
it forms a symmetric bounded metric with a value between
0 and 1, and was found to be more numerically stable than
the Bhattacharyya distance, KL-divergence, and its symmetric
form on the tested application domains. Radial basis functions
were used to interpolate the divergence costs throughout the
domain and the differential evolution method was used to find
the approximately globally optimal solutions of the generated
cost map $[27]$. 

\begin{algorithm}
\caption{High-Level Plan $\rightarrow$ Continuous State Goals}
\begin{algorithmic}[1]
\Function{ds_plan_to_cs_goals}{(high-level plan)}
\ForEach{$q_k^g$ in high-level plan}
\State Find equivalent confidence distribution $W_{goal}$:
\State $W_{goal}(q) = \begin{cases} 1, & \text{if } q = q_k^g \\ 0, & \text{else} \end{cases}$
\State Sample $n$ random points:
\State $X_{sample} = \{x_1, ..., x_n\} \sim \mathcal{X}$;
\ForEach{$x_i \in X_{sample}$}
\State Find confidence distributions on discrete
\State states $w_i \in W_{sample}$:
\State Sample a random set $X' \sim \mathcal{X}$;
\ForEach{$q' \in \mathcal{Q}$}
\State Find cost of divergence $c_i \in C' \subset \mathbb{R}$:
\State $c_i(x_i) = \text{Hellinger}(w_i, W_{goal})$;
\State Define cost map on complete domain $\mathcal{X}$:
\State $C_{complete}(x) = \text{Interpolate}(C')$;
\State Find best representative point in continuous state:
\State $x_{best} = \text{global\_optimization}(x, C_{complete})$;
\State Append $x_{best}$ to $X_{cs\_goals}$;
\EndFor
\EndFor
\EndFor
\Return $X_{cs\_goals}$;
\EndFunction
\end{algorithmic}
\end{algorithm}

\subsection{D. Trajectory Stabilization}

With the MLO assumption, it is very likely that during execution
the belief over robot state will diverge from the nominal
trajectory planned. To ensure that the execution phase belief
follows the plan, a belief space LQR (B-LQR) controller can be
defined around the nominal trajectory. B-LQR controllers were introduced by Platt et al$[23]$ and can be seen as belief-space
extension of Linear-Quadratic Regulators (LQR). For systems
with linear-Gaussian process and observation dynamics, a B-
LQR controller is optimal and equivalent to a linear-Quadratic
Gaussian (LQG) controller. In B-LQR, each point in the
nominal trajectory is defined as a set point and quadratic costs
are defined for distance from the set point and the control effort
required to converge to the set point. Closed form solutions
exist to ensure convergence to the set point within a finite
time horizon. While stabilizing the trajectory, the most likely
active discrete state is taken to define the governing dynamics
of the system. However, it may happen that B-LQR controller
is unable to stabilize the execution phase (actual) belief around
the nominal trajectory. If the planned belief for the next step
deviates more than a $\delta$-threshold from the actual belief after
the observation update, a replanning call to the planner is
triggered.

\section{V. Experiments}

The proposed POMDP solver for hybrid dynamics was
tested on three tasks: autonomous navigation for information-
gathering in a benchmark light-dark domain with spatially
varying dynamics, navigation and localization in a walled-
domain with extremely poor observations, and a real manipulation
task of partially assembling a toy airplane $[13]$ under noisy
observations by leveraging contacts to reduce uncertainty.

\subsection{A. Domain-I: Light-Dark Domain}

The first task aims to test the capability of the lower-level
planner to develop effective motion plans under spatially-
varying dynamics. A 2D light-dark domain $[23]$ was consid-
ered for the task in which observation quality is proportional
to the degree of light at a given state. The task objective was to
reach the pre-specified goal with minimum state uncertainty.
The 2D domain $(\{x, y\} \in [-10, 10])$ was considered to contain
different local dynamics functions, given as
\begin{align*}
  f(x_i, u) = \begin{cases} 
  x_i + 0.5u, & \text{if } x < -1 \\
  x_i + u, & \text{if } x \in [-1, 4] \\
  x_i + [2u_1, u_2]^T, & \text{if } x > 4 
  \end{cases}
\end{align*}
where $x_i = \{x_i, y_i\}^T$. Belief over continuous states was
considered to be a Gaussian distribution $b = \{\mu, \sigma\}$. The
observation function was taken as $h(x_i) = x_i + w$ with
zero-mean Gaussian observation noise $w \sim \mathcal{N}(0, W(x))$
where $W(x) = \frac{1}{2}(5 - x)^2 + \text{const}$. Matrices defining the
cost function over error in states, control input and addi-
tional cost for final state error and covariance were taken as
$Q = \text{diag}(0.5, 0.5)$, $R = \text{diag}(0.5, 0.5)$, $Q_{large} = 30$ and
$\Lambda = 400$ respectively.

We compare the performance of the direct-planning ap-
approach with a baseline approach which, instead of having a
probabilistic belief over the active dynamics model, determines
the active dynamics model based on the maximum likelihood
state of the continuous belief. Trajectories planned and the
actual executions are shown in Figure 7. Figure 7 shows that
the direct planning approach was able to develop effective belief space plans that move into the light to reduce uncertainty before heading to the goal even under changing local dynamics, while the baseline approach fails to do so. It can be seen from Figure 1 that if the uncertainty over continuous states is high, having a completely deterministic approach over active discrete states while planning can cause the belief to diverge completely from the actual robot state. Hence, it is critical to maintain a belief over the discrete states as well in order to develop meaningful belief space plans.

Fig. 1: Plots showing planned and actual robot trajectories for the baseline and the direct planning approach. Initial belief mean \( \mu = \{2, 2\} \), \( \text{cov} = \text{diag}(5, 5) \) (yellow cross on red trajectory). True start position: \( \{2.5, 0\} \) (yellow cross on white trajectory), goal position: \( \{0, 0\} \) (shown by green square). Brightness reflects quality of observation. Dashed lines separate different dynamics modes.

B. Domain-II: Walled Domain

In the second domain, we compared the performance of the hierarchical planner with the direct planning approach. Note that the direct planning approach is similar in principle to the SM-POMDP planner proposed by Brunskill et al. and hence, provides a comparison of the proposed hierarchical planner with a similarly passive planning approach. The 2D domain \( \{x, y\} \in [-2, 15] \) consisted of two perpendicular walls parallel to the x and y axis respectively. Observations were considered to be extremely noisy and had zero-mean Gaussian observation noise \( w \) given as \( w \sim \mathcal{N}(0, 15) \). As the motion along the wall is constrained to be only parallel to the wall, the robot can use it to efficiently localize itself in a direction orthogonal to the wall. The hybrid dynamics model of the domain can be given as

\[
f(x_t, u) = \begin{cases}
    x_t + u, & \text{if } x > -2, y > -2 \\
    x_t + 0 0, & \text{if } x < -2 \\
    x_t + 1 0, & \text{if } x > -2, y < -2 \\
    x_t + 0 0, & \text{otherwise}
\end{cases}
\]

where \( x_t = \{x_t, y_t\}^T \). Beliefs over continuous states were considered to be a Gaussian distribution \( b^r = \{\mu, \sigma\} \). Observation function was taken as \( h(x_t) = x_t + w \). Matrices defining the cost function over error in states, control input, additional cost for final state error and covariance were taken as \( Q = \text{diag}(0.5, 0.5) \), \( R = \text{diag}(10.0, 10.0) \), \( Q_{\text{large}} = 1e4 \) and \( \Lambda = 1e7 \) respectively.

Sample trajectories planned by the direct planning and the hierarchical planner are shown in Figure 2. It is evident from the figures that the hierarchical planner plans to selectively visit the two discrete states representing the walls, in contrast to the direct method. Also, the hierarchical planner is able to converge to the goal faster and with a much lower uncertainty than the direct planning approach. As the direct planner does not leverage the knowledge of local dynamics models in a structured way, it needs to plan longer trajectories to gather more information. However, due to high noise in the observations, it still fails to converge to the goal with high accuracy.

Additional statistical analysis to compare the two approaches in terms of total planning time, final error and final belief uncertainty are presented in Table 1. It can be seen from the table that, for comparable final error and final belief uncertainty, the hierarchical planner is able to find a solution approximately 5 times faster than the direct planning approach. The planning horizon was set to 20 time steps. Optimized planning parameters for both approaches were first obtained by conducting multiple test runs in the preparation phase. Belief and actual robot start conditions were taken as \( [5, 5]^T \) and \( [3.5, 2.0]^T \) respectively. The termination condition was triggered when the maximum likelihood point of the belief converged within a ball of 0.2 unit radius around the set goal \( ([0, 0]^T) \) with a maximum covariance of 1 unit.

| Metric                  | Direct | Hierarchical |
|-------------------------|--------|--------------|
| Total time (in seconds) | 51.905 | 10.695       |
| Final Error             | \([-0.168, 0.172]^T\) | \([-0.086, 0.198]^T\) |
| Final Max. Belief Uncertainty | 0.696 | 0.625 |

TABLE 1: Comparison of direct and hierarchical planning. Values are averaged over 5 runs for both methods.

C. Domain-III: Airplane assembly

Finally, we experimentally demonstrate that the hierarchical POMDP planner can be used to tractably solve a real world manipulation task—the partial assembly of a toy airplane from the YCB dataset (shown in Figure 3). We considered the first step of inserting the landing gear into the wing as a test case for our planner. The task requires high precision, as the maximum final state error margin for a successful execution is ±0.2 cm. As an added challenge, no direct feedback on the location of the hole on the wing was made available to the planner. Only noisy feedback on the location of the airplane in the world frame was provided (average estimation error ±2.0 cm); obtained by doing an online object cluster extraction, using multi-plane segmentation from the Point Cloud Library (PCL) on the point cloud data of a Microsoft Kinect v2 sensor). This experiment demonstrates two important features of the proposed planner: first, the planner can be scaled to solve real-world manipulation planning.
Fig. 2: A comparison of planned and actual trajectories using the direct planning and hierarchical planning approaches on the walled domain. For both cases, Initial belief mean \( \mu = \{5, 5\} \), \( \text{cov} = \text{diag}(11.5, 11.5) \), True start position:=\{3.5, 2.0\}. Gray circles represent belief covariance.

under uncertainty problems and second, due to the hierarchical planning approach, the planner essentially enables the robot to plan and “feel around” to localize itself even when the direct visual observations aren’t available, similar to what a human might do.

In a robot manipulation task involving contacts, based on the type of contact between the bodies, the number of state-dependent local dynamics models can be large, or even infinite. We simplify the problem by assuming an approximate hybrid dynamics model, in which the local dynamics models correspond to possible motion constraints that the robot can encounter while executing the task. For example, the task of placing a cup on a table can be considered to be approximately made of two local dynamics models: one when the two objects are in contact and the other when the cup is in contact with the table plane. The second dynamics model represents the motion constraint placed on the cup by the table by restricting its motion to be only along its plane and not penetrating it. This approximation helps in having a succinct and effective representation of the task dynamics; as under this approximation, for a specific set of inputs, the relative motion between the two objects in contact will always be the same independent of the type of contact between them. In this case, the specific set of inputs would be the set of all inputs which do not result in moving the cup away from the table plane, resulting in breaking the contact between them.

In this experiment, we consider the domain to be made up of four distinct local dynamics models: two corresponding to the linear motions along the wing plane edges, one corresponding to the corner of the plane and one to represent free-body motion elsewhere in the domain. At the highest level, the planning problem can be broken down into two steps: first, to localize the gear at a point in a plane parallel to the wing and second, to insert the gear into the hole. A hybrid dynamics model in a plane parallel to the wing can be given as:

\[
\begin{align*}
    f(x_t, u) = &\begin{cases}
        x_t + \begin{bmatrix}
            0 & 0 \\
            0 & 1
        \end{bmatrix} u, & \text{if } x \in [4, 4.5], y > -13.5 \\
        x_t + \begin{bmatrix}
            1 & 0 \\
            0 & 0
        \end{bmatrix} u, & \text{if } x < 4, y \in [-14, -13] \\
        x_t + 0 \ast u, & \text{if } x \in [4, 4.5], y \in [-14, -13.5] \\
        x_t + u, & \text{otherwise}
    \end{cases}
\end{align*}
\]

where 1 unit in continuous space = 1 cm. The observation function was given as \( h(x_t) = x_t + w \) with zero-mean Gaussian observation noise \( w \sim N(\cdot|0, 2I_2) \).

Experiments were conducted using a bi-manual manipulator robot with two Kinova \textit{Jaco} 2 7-dof arms. The planner took 14.682 seconds to plan the path. Figure 5 shows the trajectory planned by the hierarchical planner and the actual trajectory taken by the robot in a plane parallel to the wing. Figure 4

Fig. 3: Toy airplane: assembled (left) and unassembled (right)
Fig. 4: Snapshots of the robot assembling the toy airplane

Fig. 5: Planned and Actual trajectories for the airplane assembly task. Bold black lines represents the edges of the airplane wing

shows snapshots of the trajectory executed by the robot during the task from two perpendicular angles. It can be see from the second panel of the Fig. 4 that planner plans to activate the motion constraint parallel to the wing in order to reduce its uncertainty. Once localized in the plane parallel to the wing, the robot changes planes to move to a point directly above the hole and then proceeds to insert the landing gear into the airplane.

VI. CONCLUSION

Nonlinear task dynamics, especially due to sudden changes in the dynamics, can be effectively modelled using a hybrid dynamics model. A hybrid dynamics model consists of a set of local dynamics models with only one of them being active at a time. In this work, we propose a hierarchical POMDP planner for hybrid dynamics which can develop locally optimal motion plans for tasks involving nonlinear dynamics under noisy observations. The proposed planner generates hierarchical motion plans at two levels: first, a high-level motion plan that sequences the local dynamics models to be visited and second, based on the best high-level plan, a detailed continuous state motion plan to be followed by the robot. The hierarchical planning approach breaks the large POMDP problem into multiple smaller segments with shorter planning horizons, which significantly increases the computational efficiency of the planner. High-level planning also enables the robot to leverage task dynamics to improve its performance—for example, reducing uncertainty using the task motion constraints in order to develop motion plans which are more robust to state uncertainty.

In the present work, a hybrid model of the task dynamics needs to be provided to the planner by an expert. Hence, a natural extension of this work is to autonomously learn the hybrid dynamics model of the task. For example, Niekum et al. have proposed methods [28, 29] to learn the articulation motion models encountered while manipulating an object. In the future, the proposed POMDP planner may be combined with these methods to develop an end-to-end approach for learning hybrid dynamics models for the manipulation tasks and use them to generate motions plans that are robust to state uncertainty.

REFERENCES

[1] John Lygeros, Shankar Sastry, and Claire Tomlin. Hybrid systems: Foundations, advanced topics and applications. under copyright to be published by Springer Verlag, 2012.
[2] Sebastian Thrun. Probabilistic robotics. Communications of the ACM, 45(3):52–57, 2002.
[3] Christos H Papadimitriou and John N Tsitsiklis. The complexity of markov decision processes. Mathematics of operations research, 12(3):441–450, 1987.
[4] Emma Brunskill, Leslie Kaelbling, Tomas Lozano-Perez, and Nicholas Roy. Continuous-State POMDPs with Hybrid Dynamics. Symposium on Artificial Intelligence and Mathematics, pages 13–18, 2008.
[5] Hanna Kurniawati, David Hsu, and Wee Sun Lee. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In Robotics: Science and Systems, volume 2008. Zurich, Switzerland, 2008.
[6] Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based pomdp solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013.

[7] Ali-Akbar Agha-Mohammadi, Suman Chakravorty, and Nancy M Amato. Firm: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. *The International Journal of Robotics Research*, 33(2):268–304, 2014.

[8] Geoffrey A Hollinger and Gaurav S Sukhatme. Sampling-based robotic information gathering algorithms. *The International Journal of Robotics Research*, 33(9):1271–1278, 2014.

[9] Koushil Sreenath, Connie R Hill Jr, and Vijay Kumar. A partially observable hybrid system model for biped locomotion for adapting to terrain variations. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 137–142. ACM, 2013.

[10] Jur Van Den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.

[11] Vadim Indelman, Luca Carlone, and Frank Dellaert. Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments. *The International Journal of Robotics Research*, 34(7):849–882, 2015.

[12] Anirudha Majumdar and Russ Tedrake. Funnel libraries. *The International Journal of Robotics Research*, 36(8):947–982, 2017.

[13] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, 2017.

[14] David Silver and Joel Veness. Monte-carlo planning in large pomdps. In *Advances in neural information processing systems*, pages 2164–2172, 2010.

[15] Hanna Kurniawati and Vinay Yadav. An online pomdp solver for uncertainty planning in dynamic environment. In *Robotics Research*, pages 611–629. Springer, 2016.

[16] Mohamed Elbanhawi and Milan Simic. Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77, 2014.

[17] Ali-akbar Agha-mohammadi, N Kemal Ure, Jonathan P How, and John Vian. Health aware stochastic planning for persistent package delivery missions using quadrotors. In *Intelligent Robots and Systems (IROS) 2014*, 2014 IEEE/RSJ International Conference on, pages 3389–3396. IEEE, 2014.

[18] Mehmet R Dogar and Siddhartha S Srinivasa. A planning framework for non-prehensile manipulation under clutter and uncertainty. *Autonomous Robots*, 33(3):217–236, 2012.

[19] Oliver Kroemer, Christian Daniel, Gerhard Neumann, Herke Van Hoof, and Jan Peters. Towards learning hierarchical skills for multi-phase manipulation tasks. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 1503–1510. IEEE, 2015.

[20] Sergey Levine, Nolan Wagener, and Pieter Abbeel. Learning contact-rich manipulation skills with guided policy search. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 156–163. IEEE, 2015.

[21] Justin Fu, Sergey Levine, and Pieter Abbeel. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4019–4026. IEEE, 2016.

[22] Zoubin Ghahramani and Geoffrey E Hinton. Variational learning for switching state-space models. *Neural computation*, 12(4):831–864, 2000.

[23] R Platt Jr, Russ Tedrake, Leslie Kaelbling, and Tomas Lozano-Perez. Belief space planning assuming maximum likelihood observations. *Robotics: Science and Systems*, 2010. ISSN 2330765X.

[24] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Rev.*, 47:99–131, 2005.

[25] Philip E. Gill, Walter Murray, Michael A. Saunders, and Elizabeth Wong. User’s guide for SNOPT 7.6: Software for large-scale nonlinear programming. Center for Computational Mathematics Report CCoM 17-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2017.

[26] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[27] Rainer Storn and Kenneth Price. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[28] Scott Niekum, Sarah Osentoski, Christopher G Atkeson, and Andrew G Barto. Online bayesian changepoint detection for articulated motion models. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1468–1475. IEEE, 2015.

[29] Karol Hausman, Scott Niekum, Sarah Osentoski, and Gaurav Sukhatme. Active articulation model estimation through interactive perception. In *IEEE International Conference on Robotics and Automation*, 2015.