Efficient Planning for Near-optimal Compliant Manipulation Leveraging Environmental Contact

Charlie Guan, William Vega-Brown, and Nicholas Roy

***Abstract*— Path planning classically focuses on avoiding environmental contact. However, some assembly tasks permit contact through compliance, and such contact may allow for more efficient and reliable solutions under action uncertainty. But, optimal manipulation plans that leverage environmental contact are difficult to compute. Environmental contact produces complex kinematics that create difficulties for planning. This complexity is usually addressed by discretization over state and action space, but discretization quickly becomes computationally intractable. To overcome the challenge, we use the insight that only actions on configurations near the contact manifold are likely to involve complex kinematics, while segments of the plan through free space do not. Leveraging this structure can greatly reduce the number of states considered and scales much better with problem complexity. We develop an algorithm based on this idea and show that it performs comparably to full MDP solutions at a fraction of the computational cost.**

I. INTRODUCTION

Robot assembly has traditionally relied on powerful and expensive manipulators capable of very precise positioning, such as those used in automobile manufacturing. Humans approach object manipulation differently, and intuitively use contact with the environment to overcome the effects of action uncertainty and achieve high precision. Leveraging environmental contact allows the same task to be completed with a more noisy, or less powerful manipulator, and possibly one with fewer degrees of freedom [1]. Objects inevitably make contact during assembly tasks, and therefore are already amenable to contact-rich plans. A motivating example of contact-rich manipulation is the SE2 peg-in-hole problem with friction [2]. We will examine this scenario, along with others, including an implementation on the physical Baxter robot. Baxter naturally exhibits action uncertainty, but provides precise state estimates, and is sufficiently compliant to be safe around humans as well.

We begin by formulating the contact-rich manipulation problem as a Markov Decision Process (MDP). Specifically, we consider the case of action uncertainty but perfect observability. We also assume prior knowledge of the noise model and environment geometry. By densely discretizing the state space using random sampling, we can construct and solve an MDP that is an approximation to the original problem, without domain-specific knowledge. We prove that the resulting MDP policies are asymptotically optimal with the density of the discretization, and observe that such policies mitigate action uncertainty though environmental contact.

*This work was supported by Lockheed Martin Co.

All authors are with the Department of Aeronautics and Astronautics and the Robust Robotics Group within the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, 77 Massachusetts Ave., USA {cguan, wrvb, nickroy}@mit.edu

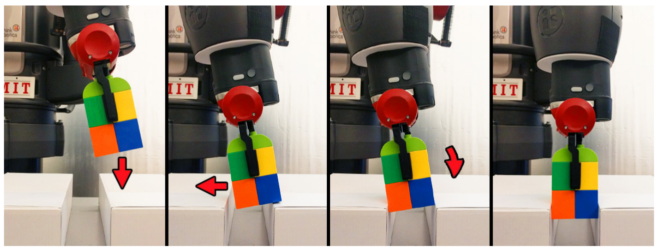

Fig. 1. Baxter uses the policy found through solving the composite MDP to place a toy block into a narrow opening. The policy pushes the block against the left side of the opening to reduce uncertainty in the left-right direction, and then is able to rotate the block into the opening.

However, constructing an MDP through conventional discrectization is computationally infeasible with larger environments. To mitigate this difficulty, we use the insight that complex kinematic constraints are only active in contact-likely regions. Therefore, we take advantage of the structure of the problem, and only explicitly generate configuration space samples in these regions and simulate the contact kinematics there. For everywhere else, we use a simple feedback policy. This composite MDP construction greatly reduces the complexity of constructing the MDP and solving for the optimal policy. We show that the approximate solutions perform well in simulation and in practice.

The main contributions of this paper are:

1) An algorithm for the efficient construction of a 'Composite MDP' which can be solved to find near-optimal solutions to contact-rich manipulation problems (Section III)
2) Proof of asymptotic-optimality of solutions derived from MDP approximations constructed using the sampled nearest neighbour transition function technique (Section IV)

II. PRELIMINARIES

Our objective in this paper is to find a policy for a stochastic system that achieves a state in the goal set and minimizes the expected time to goal from every point in the state space, subject to contact kinematic constraints.

A. *Problem Statement*

Let the state space X be a compact subset of $\mathbb{R}^{d_x}$ and the action space U be a compact subset of $\mathbb{R}^{d_u}$. Let $x(t)$ be the (fully-observable) state of the system at time t.

Also let $\{u(t) \in U | t \in \mathbb{R}_{\geq 0}\}$ be the control process and $\{w(t) \in \mathbb{R}^{d_w} | t \in \mathbb{R}_{\geq 0}\}$ be a Weiner process. Let d_x, d_u, and d_w be the dimension of the state space, control space, and noise, respectively. We consider a (diffusion) process model of the form:

$$\begin{aligned} x(t) = x(0) + \int_0^t f(x(\tau), u(\tau)) d\tau \\ + \int_0^t F(x(\tau), u(\tau)) dw(\tau) \\ s.t.\ x(t) \in X,\ \forall t > 0 \end{aligned} \tag{1}$$

where the instantaneous dynamics $f : X \times U \mapsto \mathbb{R}^{d_x}$ and and noise dynamics $F : X \times U \mapsto \mathbb{R}^{d_x \times d_w}$ are bounded measurable and continuous functions, and do not allow the state to reach the boundary of X, which effectively model contact. The process stops once a state in the goal set is reached.

We use the infinite-horizon discounted reward formulation since there is no general method to bound the maximum length of the optimal policy. The value function is defined as:

$$v_\pi(x_0) = \mathrm{E}\left[\int_0^\infty \gamma^\tau r(x(\tau)) d\tau\right] \tag{2}$$

where x_0 is the initial state, $\gamma \in \mathbb{R}_{(0,1)}$ is the discount factor, and $r(x) = \mathbb{1}_{x \in X_{goal}}$ is the reward function. The evolution of $x(t)$ is governed by equation 1 where $u(\tau) = \pi(x(\tau))$, in other words, under the policy $\pi : X \mapsto U$.

We seek the policy π^* that maximizes $v_\pi(x_0)$ for every initial state $x_0 \in X$, which is not solvable in general and usually only for simple problem classes such as those with linear dynamics, Gaussian noise, and quadratic cost. Therefore, we use the MDP approximation in the following section to find an asymptotically-optimal policy.

B. MDP Approximation

We define an MDP as a tuple $\mathcal{M} = \{S,\ A,\ T,\ R,\ dt\}$. Let S be a finite set of states and A be a finite set of actions. Let $T : S \times A \times S \mapsto \mathbb{R}_{[0,1]}$ be a Markovian transition function. Note that $\sum_{s' \in S} T(s, a, s') = 1$, where $s, s' \in S$ are the initial and final states, respectively. Lastly, let $R(s) = \mathbb{1}_{s \in S_{goal}}$ be the reward function. A reward is only earned whenever the state transitions into a goal state. Each step of the MDP occurs over a predefined time step *dt*. This concept is extended in the composite MDP presented in this paper to account for feedback-control path segments which apply over several time steps. In accordance with equation 2, we let rewards be accumulated over an infinite time horizon and exponentially discounted by γ.

Notice that this MDP formulation allows us to model state-dependent noise. We can find an approximate solution By converting the problem described in equation 1 into an MDP.

C. Fully-sampling State Space

Although structured discretizations, such as grid-based techniques, can be more efficient in representing a state space, these approaches usually require domain knowledge to employ. For higher-dimensional state spaces, or state spaces with complex geometry, structured discretizations are often impractical.

Therefore, we randomly generate the state space of the approximating MDP by sampling the kinematically-feasible (valid) regions of the configuration space using rejection sampling. The goal set is also explicitly added to the state space. Furthermore, for problems with complex contact kinematics, the MDP fidelity can only be improved by additional samples near contact manifolds. Therefore, we attempt to coerce invalid samples into valid configuration space using the physics engine. This is represented as $\textsc{ForwardSim}(s, 0)$ in Algorithm 1 (i.e., forward simulating with no input, such that only forces generated by surface penetration are applied to the object).

D. Control Input Discretization

Apart from the state space, we must also discretize the action space. The action set is discretized uniformly as in Huynh *et al.* [3]. In fact, it is imperative to explore the full action set, without bias or pruning, to ensure the optimal solution is found.

E. Generating Transition Function

Having the input space, we can proceed to calculating the transition function.

This is the key step in the construction of the MDP that allows the complex contact kinematics of the problem to be modeled, thus enabling the policy to learn to use the environment to reduce action uncertainty. However, solving for the transition function is the most computationally intensive part of the construction of the full MDP. In fact, since we have an unevenly discretized state space by virtue of our random sampling approach to creating the state space, need to compute a different transition function for each state.

For each state, the transition function is calculated by repeatedly forward simulating the result of applying each control input a, from the initial state s, with action noise w, for some constant time dt. The closest configuration in the discretized state space to each simulated sample, by Euclidean distance, is recorded as a neighbour of the initial state. The probability of reaching any neighbour, given the control input applied, is the percentage of simulated samples which arrive at this neighbour. This is formally expressed in Section IV-A. Using this transition function, we can now solve the MDP using value iteration.

F. Value Iteration

We use a standard value-iteration-based approach,

$$V_{i+1}(s) = \max_{a \in A'} \left\{ \sum_{s' \in S} V_i'(s, a, s') \right\} + \gamma V_i(s), \tag{3}$$

where

$$V_i'(s, a, s') = \gamma\ T(s, a, s')\big(V_i(s') + R(s, a, s')\big). \tag{4}$$

G. *Full-MDP Solutions*

As we will show in Section IV, formulating and solving the problem as a full MDP, where states are uniformly sampled from the kinematically-feasible region of the configuration space, and all control inputs are considered everywhere, produces asymptotically-optimal solutions in the limit of infinite samples and infinite action-space samples.

These solutions exhibit the desired behaviour of leveraging contact to overcome errors due to action noise. For instance, in the 2D peg-in-hole problem, policies learn to press the peg against the edge of the hole to reduce action uncertainty, and often succeeds on the first attempt. Similar results are shown in [4], but derived using a domain-specific approach. For comparison, a simple feedback-guided waypoint policy usually results in missing on the first try, and bouncing around the hole without success.

However, the full MDP approach quickly becomes computationally intractable. The number of state samples required empirically scales with the volume of the valid configuration region. Furthermore, calculating the transition function—which is the most computationally-intensive step—scales with $O(|S||A|)$.

III. DESCRIPTION OF COMPOSITE MDP

The key insight of the composite MDP is that a simple feedback policy can be used wherever the probability of contact given any action is very low. As a result, wherever contact is unlikely, the policy can be derived without explicit state representation. We only need discrete samples near contact manifolds. This concept is more concretely described in the following subsection. The remainder of the work is designed to accommodate this policy paradigm within an MDP framework.

A. *Generating Composite MDP State Samples*

The approach we take is similar in spirit to [5], since our approach can also be thought of as a method for variable-resolution sampling. However, rather than directly using characteristics of the value function or policy to choose where to concentrate samples, which require expensive roll-outs and an incremental algorithm, we use our knowledge of where kinematics are complex, i.e., where contact is likely.

We formally define near-contact configuration points that satisfy the following,

$$\{s \in S : \exists\ a \in A\ s.t.\ P_{contact}(s, a) > \beta\} \tag{5}$$

where $P_{contact}(s, a)$ is the probability of the state coming into contact with the environment given the initial state s and some action a, and β is a parameter which determines how far the configuration point cloud extends above the contact manifold. Algorithm 1 details how these configuration points are generated.

Essentially, we first use the sampling method described in Section II-C for state space construction, and accept the sample if it is initially invalid, but can be coerced into valid configuration space by the physics engine. In this way, we ensure our sample is in contact. If the sample is initially valid, then we choose a random action a and simulate forward an additional $n\sigma$, where σ is the noise magnitude, and n is a parameter controlling β. We repeat the simulation, resetting to the initial state and applying a different action each time, until the state makes contact, or an iteration limit is reached. If any action pushes the state into contact with the environment, the initial sample is accepted. We use the Bullet physics engine [6] to forward simulate configuration samples in the FORWARDSIM() function.

Algorithm 1 Generate Contact-likely States

```
1:  repeat
2:      $s \leftarrow$ GENERATERANDOMCONFIG()
3:      if not ISVALID($s_i$) then
4:          repeat
5:              $s \leftarrow$ FORWARDSIM($s$, $a$)
6:              if ISVALID($s$) then Accept Sample
7:                  break
8:          until Iteration Limit Reached
9:      else
10:         repeat
11:             $a \leftarrow$ GENERATERANDOMCONTROL()
12:             $a \leftarrow a(1 + n\sigma)$
13:             $s' \leftarrow s$
14:             repeat
15:                 $s' \leftarrow$ FORWARDSIM($s'$, $a$)
16:                 if INCONTACT($s'$) then Accept Sample
17:                     break
18:             until Iteration Limit Reached
19:         until INCONTACT($s'$) or Itn. Limit Reached
20: until Sufficient Samples Generated
```

Note that it should then be possible to express β as a function of n. For example, for Gaussian noise with standard deviation σ, $\beta = \frac{1}{2}\Big[1 + \mathrm{erf}(\frac{n}{\sqrt{2}})\Big]$.

Algorithm 1 terminates when samples are sufficiently dense. The samples are then placed into a KD-tree for efficient nearest-neighbour lookup and sorting by distance to an arbitrary point. An additional spatial decimation step can further reduce total computation time. This is done by iterating over all samples, and removing the sample if it has more than a certain number of neighbours within a certain radius.

B. *The Composite MDP Action Set*

The action set for the composite MDP includes both the primitive actions as constructed in Section II-D and additional feedback actions. We define these feedback actions as actions which, given an initial state and a target state, apply feedback control until the state has a high probability of contact, i.e., satisfies equation 5. In practice, the distance to the closest state in the composite MDP state space is used as a proxy to the above criteria. In other words, the feedback action terminates when the state drifts too close to any state in the composite MDP state space. Sutton *et al.* refer to these multiple time step actions as 'options' [7].

The feedback control can be quickly calculated using knowledge of the system dynamics and the immediate target configuration, since the process model in free space is simply

$$s_{i+1} = s_i + (\hat{u}_i + w)dt. \tag{6}$$

An example of a simple feedback controller used is

$$u(x) = \frac{x_g - x}{||x_g - x||} \tag{7}$$

where x_g is a feedback target. Under the feedback controller, likely paths form a cone-like volume, as seen in Figure 2.

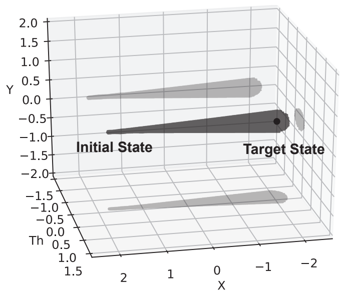

Fig. 2. This is the volume of likely trajectories achieved using a feedback controller. (Plotted in 3D and projected onto each plane for clarity.) Hypothetically, if any state existed within the volume, this goal would not be reachable.

Although it is feasible to apply a feedback action using any state in the state space as a target, this can be highly redundant, due to the termination condition mentioned above. Therefore, we only evaluate feedback actions to 'reachable' states, which we informally define as states which, when used as feedback targets, will be the neighbour which is reached with highest probability. These reachable states are returned by Algorithm 2.

Algorithm 2 GETREACHABLE()

```
 1: INPUT: $s_0$
 2: $S' \leftarrow \text{SORTBYDIST}(S,\ s_0)$
 3: $C \leftarrow \varnothing$
 4: $\vec{V} \leftarrow \varnothing$
 5: $S_{vis}$
 6: for all $s \in S'$ do
 7:     $C' \leftarrow \frac{s-s_0}{|s-s_0|} \cdot \vec{V}$
 8:     if $c > 0\ \forall\ c$ in $C - C'$ then
 9:         $\vec{V} \leftarrow \vec{V} \cup \frac{s-s_0}{|s-s_0|}$
10:         $C \leftarrow C \cup \frac{|s-s_0|}{\sqrt{|s-s_0|^2+(\rho dX)^2}}$
11:         if $|s - s_0| > \sigma dt$ then
12:             $S_{vis} \leftarrow S_{vis} \cup s$
13: return $S_{vis}$
```

Algorithm 2 begins by checking the angle between $s - s_0$ and each $s - s_i$ where s_i is a previously accepted point. If all angles are sufficiently large, we say the new point is 'reachable' and is added to the reachable set. However, the allowable angle to each previously-visited point varies, depending on the distance from s_0 to the visited point. This angle is stored as a vector of cosines C. Unit vectors from the initial point to each accepted point are also stored as $\vec{V}$. This is illustrated in Figure 3.

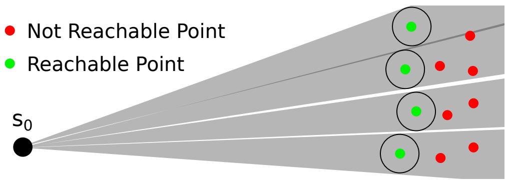

Fig. 3. This figure illustrates how reachable points are found. The four reachable points (green) are not occluded by any other points, while the unreachable points (red) are occluded. This is achieved by Algorithm 2.

Only admitting actions to reachable states also reduces the error in the assumption that all states within the feedback volume have the same best next state, as assumed in Section IV-B, since the feedback volume widens with the length. The smaller distance also reduces the likelihood of reaching a rogue state which is very far from the intended target, since paths will deviate more from the hypothetical noise-free path given longer distances.

C. Calculating the Composite MDP Transition Function

The standard one-step MDP transition function is calculated wherever a sufficient number of neighbours are available. Otherwise, reachable targets for a feedback action are found, and are added as additional actions, and their corresponding transition function is calculated.

Concretely, given a state s, if any control input a results in too many simulated samples that are each too far from the closest state in the contact-likely state space, then that control input is pushing the state into free space. Any state that can be pushed into free space in one time step is labeled a 'surface' point.

All reachable states from a surface point, as returned by Algorithm 2, are evaluated as feedback targets by the FBSIM() method, which forward simulates the surface point under feedback control to the target. Each feedback target is added as a feedback action, that can be taken from the initial state, and the associated transition probabilities are recorded.

Algorithm 3 Calculate Transition Function

```
 1: for all $s \in S$ do
 2:     $surface\_point \leftarrow False$
 3:     for all $a \in A$ do
 4:         $s_{samples} \leftarrow \text{FORWARDSIM}(s,\ a,\ num\_samples)$
 5:         if $\text{NUMTOOFAR}(s_{samples},\ S) < \mu$ then
 6:             $\text{ADDTOTRANS}(s,\ a,\ \text{TOPROB}(s_{samples}))$
 7:         else $surface\_point \leftarrow True$
 8:     if $surface\_point$ then
 9:         $X_{vis} \leftarrow \text{GETREACHABLE}()$
10:         for all $s_{tgt} \in X_{vis}$ do
11:             $s_{samples} \leftarrow \text{FBSIM}(s,\ s_{tgt},\ n\_samples)$
12:             $\text{ADDTOTRANS}(s,\ s_{tgt},\ \text{TOPROB}(s_{samples}))$
```

Discovering feedback actions between surface points is required to connect states in a non-convex configuration region, or states between distinct configuration regions (e.g. between two separate obstacles). This approach stems from the intuition that an optimal path in space consists of straight segments (feedback paths) and segments that lie on the edge or connect at the corner of an obstacle.

Concretely, if the goal state is obstructed from the current state by an obstacle, as in Figure 4, said goal state should be reachable from a sample generated around the obstacle (Point B). Therefore, the value of the goal state can then be propagated to a point reachable from the current state (Point A), which can then be used as an initial feedback target. This is detailed in lines 8-12 in Algorithm 3.

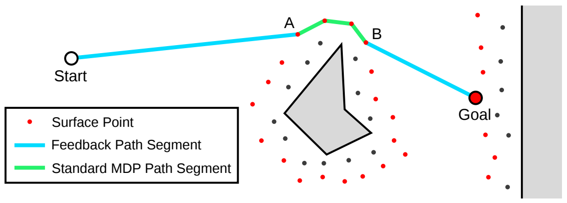

Fig. 4. The optimal path in this scenario involves using feedback control to reach point A, then the standard MDP policy until point B, then feedback control until the goal. A precomputed feedback action is used to achieve the last path segment.

Since the calculation of the transition function at each point is independent of all other points, this step is amenable to parallelization for reliable speed-ups given sufficient computing resources.

D. Solving the Composite MDP

To account for accumulated rewards in the composite MDP, the value iteration update must be modified. Equation 4 is replaced with the following:

$$V_i'(s,a,s') = \gamma^\tau T(s,a,s')\big(V_i(s') + R(s,a,s')\big) \quad (8)$$

and the set of actions to search through A', is now the union of the standard set A and the available feedback actions. Note that the reward R is zero everywhere except in the goal set. Also, the exponent of γ should be the expected number of steps to state s'. However, no noticeable difference was found compared to using the configuration space distance to calculate the expected number of time steps. Specifically, we use $\tau = \frac{\textsc{Dist}(s,s')}{dt}$, since the control input is of unit magnitude.

Value iteration terminates after no values change by more than a set threshold. The optimal policy can then be extracted by taking the argmax of each value update.

$$\pi(s) = \operatorname*{argmax}_{a\in A'} \left\{ \sum_{s'\in S} \gamma^\tau T(s,a,s')\big(V_i(s') + R(s,a,s')\big)\right\} \quad (9)$$

E. Executing the Composite MDP Policy

If the initial state is in free space, we begin by searching for the best feedback target. This is the only relatively computationally-intensive step of the procedure, but is rarely required after the first step. Once this first feedback target is reached, we simply take the optimal action encoded in the closest samples to the current state until the goal is reached. Feedback targets can also be fed to external controllers to achieve, as is done with the Baxter experiments in Section V-C.

IV. OPTIMALITY ANALYSIS

Our approach is asymptotically optimal, in the sense that the policies returned by our approach converge to the optimal solution of equation 2 as the number n of samples goes to infinity, and we outline the key ideas here. We first argue that our calculated transition function satisfies a local consistency condition, implying that the full MDP solution is asymptotically optimal; this proof follows the same outline as the proof of Huynh *et al.* [3]. Next, we argue that using feedback control in lieu of an MDP solution does not prevent asymptotic optimality.

A. Local Consistency

Consider a sequence of MDPs $\{\mathcal{M}_n\}_{n=0}^\infty$ that approximate the continuous process of equation 1 with increasingly high resolution as n tends to infinity. We let $\mathcal{M}_n = (S_n,\ A,\ T_n,\ R_n,\ dt_n)$ and let π_n be the optimal policy derived from $\mathcal{M}_n$. Kushner and Dupuis [8] give sufficient conditions on the approximate MDPs $\{\mathcal{M}_n\}$ to ensure the sequence of policies $\{\pi_n\}$ will converge the true optimal policy in the underlying continuous problem.

$$\lim_{n\to\infty} dt_n = 0 \quad (10)$$

$$\lim_{n\to\infty} ||s_{i+1}^n - s_i^n||_2 = 0,\ \forall i \in \mathbb{N} \quad (11)$$

$$\lim_{n\to\infty} \frac{\mathrm{E}[s_{i+1}^n - s_i^n | s_i^n = z, u_i^n = u]}{dt_n} = f(z,u) \quad (12)$$

$$\lim_{n\to\infty} \frac{\mathrm{Cov}[s^n - s_i^n | s_i^n = z, u_i^n = u]}{dt_n} = F(z,u)F(z,u)^T \quad (13)$$

s_i^n and u_i^n are the i^{th} state and control in the sequence, respectively, in the n^{th} MDP in the sequence. Equations 10 and 11 hold trivially for samples drawn uniformly from free space and for the sequence of times $dt_n \propto (\frac{\log n}{n})^{1/d}$.

We sketch a proof that the conditions in 12 and 13 also hold in our algorithm. The transition model described by algorithm 3 can be described mathematically as

$$\hat{P}(\Delta s|s,a) = \frac{1}{|S_n|M}\sum_{q=1}^{|S_N|}\sum_{m=1}^{M} K_n\Big(\Delta s_q, \Delta s_m\Big)\delta(\Delta s{-}\Delta s_q) \quad (14)$$

where $\Delta s = s(t+dt) - s(t)$ is the change in state after applying action a from state s, and M is the number of forward-simulated samples. The function $\delta(x)$ denotes the Dirac delta function, and the functions $K_n(r)$ can be any sequence of kernel functions satisfying

$$\int \mathrm{d}\Delta s K_n(\Delta s_q, \Delta s_m) = 1 \quad (15)$$

$$\lim_{n\to\infty} K_n(\Delta s_q, \Delta s_m) = \delta(\Delta s). \quad (16)$$

We recover Algorithm 3 when we let $K(\|\Delta s\|)$ be proportional to 1 when $\|\Delta s\|$ is less than or equal to the distance h to the closest neighbor of the forward-simulated sample, and 0 otherwise.

$$K_n(\Delta s_q, \Delta s_m) = \begin{cases} \frac{1}{\zeta_d h^d} & \|\Delta s_q - \Delta s_m\| \le h \\ 0 & \text{else} \end{cases} \tag{17}$$

We argue informally that this model satisfies equations 12 and 13. For example, the expectation in equation 12 can be written explicitly using equation 14:

$$\hat{\underset{\Delta s}{\mathrm{E}}}\left[\Delta s\right] = \frac{1}{|S_n|M} \sum_{q=1}^{|S_n|} \sum_{m=1}^{M} K_n\Bigg(\|\Delta s_q - \Delta s_m(s,a)\|\Bigg)\Delta s_q \tag{18}$$

where the expectation is taken with respect to the empirical transition probability. Consider this expectation as a function of the forward simulated samples δs_m; it follows from the consistency of kernel density estimation and the Lipschitz continuity of the actual transition probability $P(\Delta s|s,a)$ that

$$\lim_{M\to\infty} \hat{\underset{\Delta s}{\mathrm{E}}}\left[\Delta s\right] = \frac{1}{|S_n|} \sum_{q=1}^{|S_n|} (p(\Delta s_q|s,a) + o(\bar{h}))\Delta s_q \tag{19}$$

where $\bar{h}$ is the average distance between samples. Again relying on the continuity of the actual transition probability, it follows that

$$\lim_{|S_n|,M\to\infty} \hat{\underset{\Delta s}{\mathrm{E}}}\left[\Delta s\right] = f(s,a)dt_n + o(\bar{h}). \tag{20}$$

Because our samples are drawn uniformly from free space, we can show that $\bar{h} = \mathcal{O}(n^{1/d})$. Since $dt_n = (\frac{\log n}{n})^{1/d}$, it is straightforward to show that

$$\lim_{n\to\infty} \frac{o(h)}{dt_n} = \lim_{n\to\infty} (\log n)^{-1/d} = 0 \tag{21}$$

and therefore 12 holds. A similar argument applies for equation 13. Since our approach satisfies the conditions in equations 10-13, the sequence of policies converge to the true optimal policy.

B. *Near-optimality of Feedback Policies*

Our second deviation from conventional continuous MDP solutions is the use of feedback control instead of a policy computed from the approximate MDP in free space. We can use feedback control while maintaining near-optimality in regions without contact-likely states, since complex contact kinematics are not relevant to planning in this region.

Feedback control can be thought of as the solution to an LQR control problem, where there is some non-zero constant cost for each control, and a large coefficient for the quadratic error term. We stress that these assumptions do only apply to regions which are unlikely to come into contact with obstacles, and when goal states are appropriately chosen such that the target is reachable, such as by the GETREACHABLE() function.

However, we must make the assumption that at any point within the feedback volume, the best next state in the MDP does not change, since the feedback target cannot change until the feedback action terminates. We argue that this is not an unreasonable assumption, given the compactness of the feedback volume under reasonable noise, as seen in Figure 2. This condition ensures an accurate value update and the visibility-based feedback goal selection is a valid heuristic. For further discussion on the latter, see Section III-C.

When both these assumptions are true, the optimal policy produced by a full MDP in free space with infinite samples is simply a feedback policy. Since the feedback policy is an optimal MDP policy, and controlled Markov chains on the MDP converge to solutions of equation 1 as the probability of contact goes to zero ($\beta \to 1$ in equation 5), the optimality of solutions on the composite MDP is well-approximated.

V. EXPERIMENTAL RESULTS

A. *Peg-in-hole Environment*

The classic peg-in-hole environment is used to validate the composite MDP. The state and action space are defined in SE2, describing a fully-actuated 2D peg-in-hole scenario. The pivot point (and origin) of the peg is defined well above the centroid of the peg, to mimic the grasp of a manipulator. The peg interacts with the surrounding environment with realistic friction. Both these factors add difficulty to the task. We assume no gravity, simulations assume 16% Gaussian noise ($\sigma = 0.167$, where $|\hat{u}| = 1$). A very low noise magnitude precludes the need for accounting for action uncertainty, while noise magnitudes above a certain threshold are impossible to overcome by any control policy. In this scenario, the control scheme performed well ($> 85\%$ success rate) until $\sigma = 1.5$.

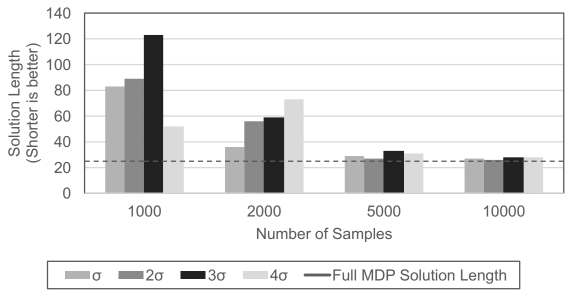

Fig. 5. Solution length plotted against number of samples for each contact cloud depth ($n\sigma$). Policies reaches optimal performance around 5000 samples, whereas the full MDP solution requires at least 25000 samples. In general, lower n produces better performance, but might require more feedback goal-searching during execution. The anomalous performance for 4σ at 1000 samples could be due to a sampling anomaly, retrials did not show this effect.

The composite MDP produces intelligent policies which use the side of the hole to reduce action uncertainty. Figure 5 compares the relative performance of using 1-4σ contact clouds. Computation time is approximately one hour per 1000 samples for the full and composite MDP solution. Table I summarizes the computation time required for various methods. We compare against SARSA [9], which also produces asymptotically optimal solutions. The SARSA method does not fare well, and only has a lower average solution length because unsuccessful paths are excluded. Our method requires less than a quarter of the computation time as the full MDP solution, while producing comparable performance.

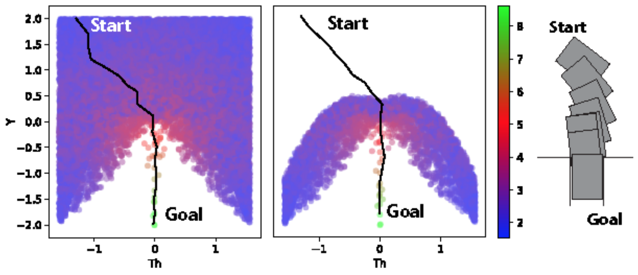

Fig. 6. The two plots show a solution path of the peg-in-hole problem, plotted over sample states coloured by the value function. The far left plot shows the fully-sampled MDP solution, while the middle plot shows the composite MDP solution. The latter approach performs similarly while requiring a fraction of the computation since far fewer samples are required. The diagram on the right shows a simulated solution path of the peg. Note how the peg swings left against the hole to reduce uncertainty in the x direction.

B. Narrow Corridor Environment

In this environment, a five-sided robot in SE2 is tasked with passing through a narrow corridor, with realistic friction. It can be thought of as an extension to the SE2 peg-in-hole, but the peg can be inserted either side down. Similar noise magnitudes to the previous environment are used.

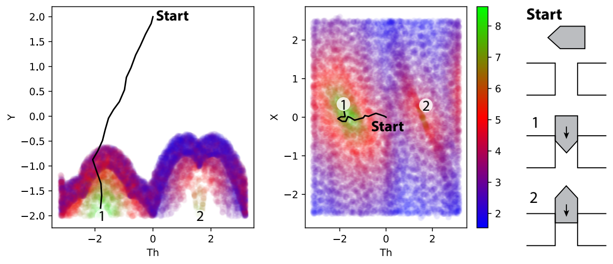

Fig. 7. The two plots show one executed trajectory in configuration space. The coloured dots represent the value function at each sampled point. Configurations of interest are labeled and illustrated on the right. The composite MDP algorithm correctly chooses to head into the corridor angled-side first.

The optimal policy must minimize solution length by appropriately choosing between entering the corridor blunt-side first or angled-side first. When starting from a neutral position, the policy always makes the correct choice, as shown in Figure 7. For comparison, entering the corridor blunt-side first takes 22.3 times steps versus 21.1 time steps entering angled-side first (averaged over 500 trials).

This behaviour only emerges when sufficient action noise is present in the system. In a completely noise-free scenario, both goal states would take the same number of steps from the goal position.

C. Baxter Experiments

The Baxter research robot is an excellent example of a platform with accurate state estimates, but actuators which cannot achieve the same precision. Less powerful actuators allow Baxter to be safe for close-quarters human-robot collaboration tasks, whereas traditional manipulators are not. We also set force and displacement limits on the end effector in the spirit of compliant manipulation, to further protect the object and environment against damage.

We assume the manipulator movement is largely free from any significant dynamic effects, such as momentum. In practice, this is a reasonable assumption when manipulator joint speeds are limited, and any effects can be handled as action uncertainty.

The optimal policy calculated for the peg-in-hole environment is evaluated on the Baxter research robot. The default endpoint controller is used to achieve feedback targets. The policy performs well. The peg-in-hole task was attempted using the default Baxter position controller and the MDP policy, for 100 trials each. The default Baxter controller failed 48 times, while the MDP policy only failed 3 times out of 100 trials. The policy accomplishes this by leveraging environmental contact to reduce action uncertainty despite discrepancies between real-world geometry and dynamics and the simulation environment, as described in Figure 1.

VI. PREVIOUS WORK

The SE2 peg-in-hole problem in general is a well-studied compliant manipulation task [10], [11], [12]. Approaches have involved hybrid control (force control in some dimensions and position control in the rest) [13], as well as mechanical solutions leveraging compliant components [14]. More recent approaches to the peg-in-hole problem, and other planning tasks in SE2 and SE3, are presented in [15]. This method also uses a physics engine to simulate complex dynamics, and focuses on the idea of 'reversible' actions. However, no guarantee of optimality is made. Compliant manipulation in general has been studied since Mason's seminal work [16], [17], [18]. Contact-rich manipulation in particular has been addressed with neural networks [19], [20], including recent policy search techniques such as Guided Policy Search [21]. However, these are also focused on feasibility rather than optimality, albeit in higher dimensional state spaces. LaValle *et al.* also address complex dynamic and obstacle-based constraints in [22], but do not address optimality. Approaches from stochastic optimal control literature do address optimality [23], [24], but require a Gaussian noise model. Our approach is compatible with a wide range of noise models, and performs well with highly-localized reward functions. This is not necessarily the case with other approaches to continuous control problems, [25], [26]. This method is largely inspired by Vega-Brown and Roy's work on optimal integrated task and motion planning [27].

VII. CONCLUSION

We have shown that the composite MDP solution is an efficient approach to finding near-optimal contact-rich

Environment	Method	# Samples	Total Computation Time (Minutes)	% Success	Average Solution Length (Time Steps)
Peg-in-hole	SARSA	25000	1220	14	21.28
Peg-in-hole	Full MDP Solution	25000	1290	100	24.86
Peg-in-hole	Composite MDP Solution (2-sigma)	5000	289	100	26.63

TABLE I
PERFORMANCE COMPARISON OF THE VARIOUS MDP SOLUTION METHODS

manipulation plans, since it scales with the total area of the contact surfaces in configuration space, rather than the full volume of the configuration space. These contact-likely clouds (and their associated transition functions) can also theoretically be reused for new environments involving the same object geometry, saving re-computation of the transition function. The transition function can also be reused to calculate policies to other goals in the environment. Our approach is also amenable to a wide range of noise models and does not require any domain knowledge beyond environment geometry.

Although the composite MDP and underlying physics engine is compatible with SE3 and more than 6 DOF, some work may be required to extend this to be feasible in SE3 space. The MDP could also be extended to account for system dynamics, such as momentum. The process currently also lacks compatibility with arbitrary cost functions, but this is also a plausible extension.

Another interesting direction would be applying the composite MDP to underactuated problems. Since underactuation effectively reduces the search space of the problem, the solutions should be quicker to compute!

ACKNOWLEDGMENT

The authors thank all the members of the Robust Robotics Group for their diligent review of this paper and all of their support.

REFERENCES

[1] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in *IEEE International Conference on Intelligent Robots and Systems*, no. December, 2015, pp. 6215–6222.

[2] M. Caine, T. Lozano-Perez, and W. Seering, "Assembly strategies for chamferless parts," in *IEEE International Conference on Robotics and Automation*, 1989.

[3] V. A. Huynh, S. Karaman, and E. Frazzoli, "An Incremental Sampling-based Algorithm for Stochastic Optimal Control," *International Journal of Robotics Research*, vol. 35, no. 4, pp. 1–34, 2016.

[4] K. Senda and Y. Tani, *Reinforcement Learning of Robotic Manipulators*, D. Zhang and B. Wei, Eds., Boca Raton, 2016.

[5] R. Munos, A. Moore, and S. Singh, "Variable Resolution Discretization," *Machine Learning*, vol. 49, no. 2, pp. 291–323, 2002.

[6] E. Coumans, "Bullet Physics Library," 2006. [Online]. Available: http://bulletphysics.org/

[7] R. S. Sutton and S. Singh, "Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning," *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

[8] H. Kushner and P. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time*, 1992.

[9] G. A. Rummery and M. Niranjan, "On-line Q-learning using connectionist systems, Tech. Rep. CUED/F/INFENG/TR 166, 1994.

[10] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor, "Automatic Synthesis of Fine-Motion Strategies for Robots," *The International Journal of Robotics Research*, vol. 3, no. 1, pp. 3–24, 1984.

[11] H. Bruyninckyx, S. Dutre, and J. De Schutter, "Peg-on-hole: a model based solution to peg and hole alignment," in *IEEE International Conference on Robotics and Automation*, vol. 2, 1995, pp. 1919–1924.

[12] S. Chhatpar and M. Branicky, "Search strategies for peg-in-hole assemblies with position uncertainty," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2001, pp. 1465–1470.

[13] Yangmin Li, "Hybrid Control Approach to the Peg-in-Hole Problem," *IEEE Robotics and Automation Magazine*, vol. 4, no. 2, 1997.

[14] W. Haskiya, K. Maycock, and J. a. G. Knight, "A passive compliant wrist for chamferless peg-in-hole assembly operation from vertical and horizontal directions," *Proceedings of the Institution of Mechanical Engineers*, vol. 212, no. 6, pp. 473–478, 1998.

[15] C. Phillips-Grafflin and D. Berenson, "Planning and Resilient Execution of Policies For Manipulation in Contact with Actuation Uncertainty," *Workshop on the Algorithmic Foundation of Robotics*, 2016.

[16] M. T. Mason, "Compliance and Force Control for Computer Controlled Manipulators," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 11, no. 6, pp. 418–432, 1981.

[17] M. T. Mason, *Compliant Manipulation*, M. Brady, J. M. Hollerbach, T. Johnson, T. Lozano-Perez, and M. T. Mason, Eds. Cambridge, Massachusetts: MIT Press, 1983.

[18] M. A. Peshkin, "Programmed Compliance for Error Corrective Assembly," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 473–482, 1990.

[19] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient Deep Reinforcement Learning for Dexterous Manipulation," 2017.

[20] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-End Training of Deep Visuomotor Policies," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[21] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 156–163.

[22] S. M. LaValle and J. J. Kuffner Jr, "Randomized Kinodynamic Planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[23] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine, "Combining Model-Based and Model-Free Updates for Trajectory-Centric Reinforcement Learning," in *International Conference on Machine Learning*, 2017, pp. 703–711.

[24] J. van den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[25] K. Doya, "Reinforcement Learning in Continuous Time and Space," *Neural Computation*, vol. 12, pp. 219–245, 2000.

[26] M. G. Lagoudakis, "Least-Squares Policy Iteration," *Journal of Machine Learning Research*, vol. 4, pp. 1107–1149, 2003.

[27] W. Vega-Brown and N. Roy, "Asymptotically optimal planning under piecewise-analytic constraints," in *Workshop on the Algorithmic Foundation of Robotics*, 2016.