Motion Planning with Competency-Aware Transition Models for Underactuated Adaptive Hands

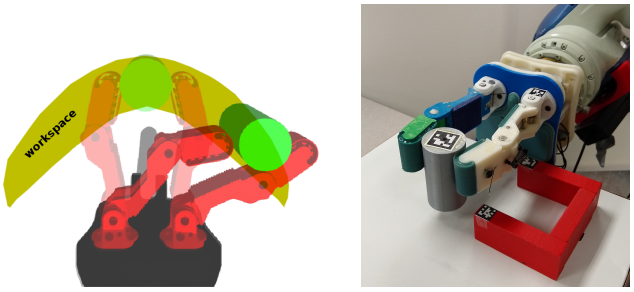
Avishai Sintov*, Andrew Kimmel*, Kostas E. Bekris and Abdeslam Boularias

***Abstract*— Underactuated adaptive hands simplify grasping tasks but it is difficult to model their interactions with objects during in-hand manipulation. Learned data-driven models have been recently shown to be efficient in motion planning and control of such hands. Still, the accuracy of the models is limited even with the addition of more data. This becomes important for long horizon predictions, where errors are accumulated along the length of a path. Instead of throwing more data into learning the transition model, this work proposes to rather invest a portion of the training data in a *critic* model. The critic is trained to estimate the error of the transition model given a state and a sequence of future actions, along with information of past actions. The critic is used to reformulate the cost function of an asymptotically optimal motion planner. Given the critic, the planner directs planned paths to less erroneous regions in the state space. The approach is evaluated against standard motion planning on simulated and real hands. The results show that it outperforms an alternative where all the available data is used for training the transition model without a critic.**

I. Introduction

Dexterity and affordable hardware are desirable properties for a robotic hand to be viable in practical applications. Such features, however, are often conflicting since dexterity requires many degrees of freedom and sophisticated control, which raise cost. Underactuated hands with compliant fingers, such as the ones seen in Figure 1, are appealing in this context due to their adaptability and simplicity [1]. They enable stable and robust grasps with open-loop control, and can perform precise within-hand manipulation [2]–[4]. Such manipulation capabilities are required in tasks where the robotic arm is limited in movement, such as placing items in a loaded shelve, confined closet, or during invasive medical procedures.

Due to uncertainty in the manufacturing process, open-sourced hands differ in size, weight, friction and inertia [5]. For example, 3D-printed units of the same hand model often differ in their mechanical properties due to variations in fabrication. Consequently, precise analytical models for such hands are often unavailable, as they are hard to derive. Thus, and as in previous work [4], data-based modeling enables useful predictions and can be used for motion planning and closed-loop control. Nevertheless, the accuracy of a learned model is limited, even with increased data. For instance, neural networks with a fixed number of units and layers have a limited capacity to fit functions. Their accuracy begins to plateau after utilizing a certain amount of data for training.



(a) (b)

Fig. 1. Two-fingers underactuated hands manipulating a cylinder in (a) Gazebo environment with the illustration of the $x-y$ workspace (in yellow) and in (b) a real experiment.

Inaccurate predictions of transition models are particularly problematic when the models are used in a Non-Observable Markov Decision Process (NOMDP) [6], where predictions are performed for a long horizon based solely on an initial state and a sequence of future actions. Hence, prediction errors are accumulated, yielding rollouts that fail to accurately track the planned paths. This makes NOMDP planning a challenge.

This work claims that it is more effective to invest more data, if available, to generate a *critic* of the learned model rather than improving the model. The critic, trained with data independent of the model, will provide information on the average model prediction error given a current state and a future action, along with a history of the past actions that led to the state. By incorporating the critic into a model-based motion planning framework, the planner will be able to avoid regions of erroneous predictions. This paper presents the generation of such critic, based on a learned model, and its incorporation to an asymptotically optimal motion planning algorithm. The planning algorithm has as objective to minimize the error along the path, thus, diverting the path to regions better predicted by the model. The approach is implemented for in-hand manipulation tasks with underactuated adaptive hands, including a physics-engine simulation as well as demonstrations on a real hand. Both the transition model and the critic are learned using a neural network.

II. Related Work

This section reviews prior work related to underactuated adaptive hands and data-driven transition models.

A. Sintov is with the School of Mechanical Engineering, Tel-Aviv University, Israel. e-mail: sintov1@tauex.tau.ac.il

A. Kimmel, K. Bekris and A. Boularias are affiliated with the Computer Science Department of Rutgers University, New Brunswick, NJ, USA. email:{andrew.kimmel, kostas.bekris, abdeslam.boularias}@cs.rutgers.edu

*These authors have equally contributed to the work.

This work is supported by NSF awards IIS-1734492, IIS-1723869, IIS-1846043, and CCF-1934924. The opinions and findings in this paper do not necessarily reflect the sponsor's views.

A. Underactuated adaptive hands

Underactuated hands exhibit complex responses due to joint passivity and contact between fingers and the environment [4]. Open-sourced hands [5] are fabricated through 3D printing and therefore, make it hard to extract precise mechanical properties (e.g, joint friction, spring stiffness and contact models). For this reason, along with the inability to control individual joint positions, accurate models are difficult to derive. Modeling tools for underactuated manipulation have been introduced in several efforts [7]–[9], which examine joint configurations, torques, and energy with a simplified frictional model [10]–[12]. A popular modeling technique applies a hybrid parallel/serial approach using screw theory, which further simplifies the derivation of a model [13]. Nevertheless, these proposed techniques have been shown to be sensitive to assumptions in external constraints and are mostly suitable for simulations.

Few attempts were made to control underactuated hands. A recent work provided a linear approximation of the hand kinematics through manipulation primitives [2]. While applying these primitives, the manipulated object tends to move in non-linear, arc-like trajectories [4], enabling solely the use in visual servoing with substantial tuning, and not for model-based planning. Nevertheless, the method was used to track paths planned with an optimization-based model-free planner [3]. The planning was performed solely on the basis of simple point-to-point local connection. Consequently, state uncertainty cannot be taken into account and the planner does not reason about the probability of success. A robust planner, however, that finds the highest probability path to successfully manipulate an object with an underactuated hand to a desired goal was recently proposed [14]. Using a data-driven learned transition model proposed in [4] (and discussed below), a belief-space planner reasons about the distribution of propagated states derived from model stochasticity and initial state uncertainty. A model-free approach [15] applied tactile sensing with reinforcement learning to learn in-hand manipulation motions, and was demonstrated on an underactuated hand.

B. Data-based transition models

A transition model is a mapping from a given state and action to the next state. Such models are used in model-based RL [16]–[19] and motion planning [14]. They are often obtained through non-linear regression in a high-dimensional space. Usages of data-driven models include learning the probability distribution of an object after a grasp [20] or during regrasping [21], and a hybrid modeling approach combining analytical and data-based models to improve accuracy in feed-forward control [22]. Neural networks have became more popular recently thanks to their simplicity, capacity of learning, and scalability to large amounts of data.

C. Competency-aware Learning

The increased popularity of machine learning techniques in robotics and other application areas led to the question of creating tools that can independently assess the accuracy of the learned predictive models [23]–[26]. In object detection and localization, for example, popular models such as Mask R-CNN [27] typically provide a confidence score on their predictions. The score can be used by other component of a system to make decisions accordingly. Bayesian methods such as GPs [28] also provide confidence scores as probabilities of their predictions. Confidence scores are widely used in a closely related approach known as active learning [29]–[31]. In active learning, confidence scores are used to guide an exploration policy towards regions where the learned model is under-performing, in order to gather more data there and to improve it.

In the proposed approach, the critic is used during planning to penalize trajectories that go through regions where the learned model is inaccurate. The method differs from previous works in the way the critic is learned. In previous efforts, confidence scores are given by the learned model itself. Here, the critic is learned independently as a separate entity. This work highlights that the confidence scores provided by the learned transition model (GP or neural network) cannot be always trusted since they are obtained from the same data that was used for training the model. For example, prior work [32] provided examples where function uncertainty cannot be obtained by applying a softmax on the output of a neural network, as typically done in the literature. Thus, the proposed critic uses a separate set of data for learning to predict the accuracy of the learned transition model.

III. Problem Setup and Notation

Let $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$ be an observable state vector of a given underactuated hand and $\mathbf{u} \in \mathcal{U}$ be an action taken from a set $\mathcal{U}$ of possible actions. The state space is decomposed into a valid subsect $\mathcal{X}_{valid}$ and an invalid one. Validity typically refers to the state being collision-free or not dropping the object. The system is governed by the transition $f : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$, such that a given state-action pair $(\mathbf{x}_t, \mathbf{u}_t)$ at time t is mapped to the next state $\mathbf{x}_{t+1}$ according to $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$. While the actual transition f is considered unknown, a learned model $\tilde{f}(\mathbf{x}_t, \mathbf{u}_t) \approx \mathbf{x}_{t+1}$ can be acquired through regression of recorded data as presented before [4] and briefly discussed in the next section.

This work considers a NOMDP problem. In this setup, observations are not available and planning is performed solely based on the initial state $\mathbf{x}_o$. Given the learned model $\tilde{f}$, the problem is to plan a path $\pi_{\mathbf{x}} : [0, 1] \to \mathcal{X}$ and acquire the corresponding sequence of actions $\pi_{\mathbf{u}} : [0, 1] \to \mathcal{U}$, such that $\tau(1) \in \mathbb{G}(\mathbf{x}_g)$, where $\mathbb{G}(\mathbf{x}_g)$ is the region of the goal state $\mathbf{x}_g$. $\tau : [0, 1] \to \mathcal{X}$ is the real tracked path when rolling-out $\pi_{\mathbf{u}}$ on the system from $\pi_{\mathbf{x}}(0) = \tau(0) = \mathbf{x}_o$. In addition, path $\pi_{\mathbf{x}}$ must optimize a cost function C, such as path duration or length.

IV. Approach

The proposed system has three main components: a transition model learned from data, a competency-aware (i.e., critic) model learned also from data, and an open-loop NOMDP planner based on the transition and critic models.

A. Learning Transition Model

As discussed previously, a precise model of a system $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$ is not always available. An approximate data-driven model $\mathbf{x}_{t+1} = \tilde{f}(\mathbf{x}_t, \mathbf{u}_t)$ offers an alternative and can be used in model-based algorithms. A training set is acquired by applying random actions to the system, while recording the observable states. Consequently, the resulting data is a set of state-action trajectories $\lambda_i \in \mathcal{P}$, where $\lambda_i = \big((\mathbf{x}_0^i, \mathbf{u}_0^i), \ldots, (\mathbf{x}_k^i, \mathbf{u}_k^i)\big)$. For generating the critic as discussed in the next section, we divide the data in $\mathcal{P} = \{\mathcal{H}_{model}, \mathcal{H}_{critic}\}$. Trajectories in subset $\mathcal{H}_{model} \subset \mathcal{P}$ are pre-processed to a set of training inputs $\bar{\mathbf{x}}_i$, where $\bar{\mathbf{x}}_i = (\mathbf{x}_i^T,\ \mathbf{u}_i^T)^T$, and corresponding output labels of the next state $\mathbf{x}_{i+1}$. Thus, defining the training set $\mathcal{T} = \{(\bar{\mathbf{x}}_i, \mathbf{x}_{i+1})\}_{i=1}^N$, later used to train a recurrent neural network. Each $\bar{\mathbf{x}}_i$, is also labeled with $\mathbf{d}_i = \{\texttt{success}, \texttt{fail}\}$ indicating whether the transition from $\mathbf{x}_i$ with action $\mathbf{u}_i$ resulted in a failure. This is used to train a classifier that provides the probability of $\bar{\mathbf{x}}_i$ to fail (object dropped or actuators overload).

B. Critic Model

Since the available model $\tilde{f}$ is learned based on data, errors are inevitable. Moreover, planning in a NOMDP fashion imposes the accumulation of error, where the magnitude mainly depends on the regression method, length of the path, regions of motion in the state-space and number of action changes. Therefore, the new objective is an independent critic model Γ that will estimate the error of the learned transition function at a certain state with the intended actions to be applied in the near future. One can query the critic with regards to the future sequence of actions with some horizon. However, since the critic is used within a sampling-based planning framework, it is required to sample solely one action to be applied for a number of steps. Furthermore, it is important to estimate the error that led to the current state, that is, we include past actions for some horizon along the search tree and their resulting prediction error. This reflects the fact that different sequences of actions can lead to varying levels of accuracy of the predictive model. For instance, certain maneuvers, or shapes of trajectories, are more difficult to perform or predict accurately than others. This can be due to the distribution of the training data that may not sufficiently cover all types of trajectories, or to uncertainties that are inherent to adaptive hands. Since we do not know *a priori* which sequences of actions lead to higher prediction errors, we include the history of past actions as an input to the critic and train it to predict how accurate the learned transition model would be given that history. For comparison, we distinguish between the `CRITIC` and the History-`CRITIC` (`H-CRITIC`), where only the latter has past actions as input.

To generate the `H-CRITIC` model, we require recorded state paths not used in training the transition model $\tilde{f}$. Thus, the data is a set of state-action trajectories $\mathcal{H}_{critic} \subset \mathcal{P}$ pre-processed according to Algorithm 1. In brief, we generate the critic input and output sets, $\mathcal{Y}$ and $\mathcal{E}$, respectively. To create a data point $(\mathbf{y}, e)$ (i.e., $\mathbf{y} \in \mathcal{Y}$ and $e \in \mathcal{E}$), we sample a trajectory λ_i from $\mathcal{H}_{critic}$, a state-action pair on the trajectory $(\mathbf{x}_j^i, \mathbf{u}_j^i)$ and the number of steps $n \in [m_l, m_h]$ to apply action $\mathbf{u}_j^i$. Thus, the input point $\mathbf{y}$ of the critic is composed of: 1) the state-action pair $(\mathbf{x}_j^i, \mathbf{u}_j^i)$, 2) the sampled number n, and 3) the sequence A of previous n_h actions that led to $\mathbf{x}_j^i$, where n_h is a pre-defined number. The output label $\mathbf{e}$ is the sum of past and future Root-Mean-Square-Errors (RMSE) between the path predicted by the learned transition model and the ground-truth for the specific path segment. Note that unlike `H-CRITIC`, past RMSE is not included in the `CRITIC` version that does not take histories into account.

Algorithm 1: `generate_critic_data`$(\tilde{f}, \mathcal{H}_{critic}, n_h)$

Initialize empty sets $\mathcal{Y}$ and $\mathcal{E}$;
for $k \leftarrow 1$ *to* N **do**
  Randomly choose trajectory λ_i from $\mathcal{H}_{critic}$;
  Sample $n \in [m_l, m_h]$;
  Sample $j \in [0, |\lambda_i| - n]$;
  $\mathcal{X}_f \leftarrow \{\mathbf{x}_j^i, \ldots, \mathbf{x}_{j+n}^i\}$;
  $\mathcal{X}_h \leftarrow \{\mathbf{x}_{j-n_h}^i, \ldots, \mathbf{x}_j^i\}$;
  if $n_h \leq j$ **then**
    $A \leftarrow \{\mathbf{u}_{j-n_h}^i, \ldots, \mathbf{u}_{j-1}^i\}$;
  else
    $A \leftarrow \{\mathbf{0}, \ldots, \mathbf{0}, \mathbf{u}_0^i, \ldots, \mathbf{u}_{j-1}^i\}$; `// padding`
  $\mathbf{y} \leftarrow (\mathbf{x}_j^i, \mathbf{u}_j^i, n, A)$;
  Add $\mathbf{y}$ to $\mathcal{Y}$;
  $\mathbf{s} \leftarrow \mathbf{x}_j^i$;
  Initialize set $\mathcal{S}_f$ with $\mathbf{s}$; `// future actions`
  for $m \leftarrow 0$ *to* n **do**
    $\mathbf{s} \leftarrow \tilde{f}(\mathbf{s}, \mathbf{u}_{j+m}^i)$;
    Add $\mathbf{s}$ to $\mathcal{S}_f$;
  Initialize set $\mathcal{S}_h$ with $\mathbf{s}$; `// only H-CRITIC`
  for $m \leftarrow n_h$ *to* 1 **do**
    $\mathbf{s} \leftarrow \tilde{f}(\mathbf{s}, \mathbf{u}_{j-m}^i)$;
    Add $\mathbf{s}$ to $\mathcal{S}_h$;
  Add $\texttt{RMSE}(\mathcal{X}_f, \mathcal{S}_f) + \texttt{RMSE}(\mathcal{X}_h, \mathcal{S}_h)$ to $\mathcal{E}$;
return $\mathcal{Y}, \mathcal{E}$;

Once the training data has been acquired, the critic model predicts an error based on local GP regression. Given a query point $\mathbf{y}_k$, the K nearest neighbors $\mathcal{Y}_K \subset \mathcal{Y}$ are found. Then, GP regression is performed on $\mathcal{Y}_K$ and its matching output set $\mathcal{E}_K$ to acquire the predicted error $\tilde{e}_k$.

C. Planning with a Learned Transition Model

The objective of the planner is to compute an optimal sequence of controls $\mathbf{u}^*$ for a system with unknown/complex dynamics or noise, such that 1) the cost function is optimized; 2) the trajectories rolled out from these controls have a high likelihood of reaching the goal and 3) the trajectories remain valid by avoiding collisions or undesirable states (e.g. object dropped). There are two main components necessary to perform such kinodynamic motion planning - state validity and state transition. Related work has proposed using learned models built from collected data from the adaptive hand to generate a classification of the valid state space [33] and

a state transition model [4]. This work accordingly utilizes such data-driven models in the planner.

The high dimensionality of the system, coupled with the non-trivial amount of time required to inquiry the data-driven models (transition and validity), restricts the use of more traditional planning approaches (e.g. A*-like or an RRT).

D. Integration with Competency-Aware Models

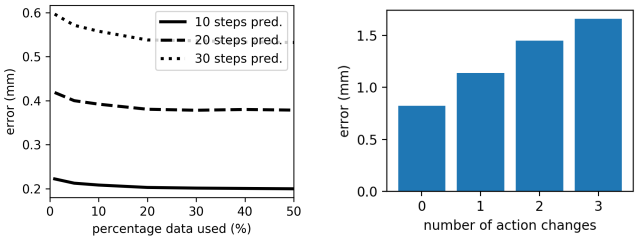
We extend the planners proposed in [14], [34] to utilize the competency-aware model described above. Specifically, we use the deterministic planner `STANDARD`, and incorporate the critic error into the cost function. Let the cost of an edge $E_t = (\mathbf{x}_t, \mathbf{u}_t)$ from state $\mathbf{x}_t$ with action $\mathbf{u}_t$ be the error estimated by the critic model (Equation 1) as described in Section IV-B. For the `CRITIC` variant, the query $\mathbf{y}_t = (\mathbf{x}_t, \mathbf{u}_t)$ depends only on the current state-action. For the `H-CRITIC` variation, the last n_h actions are included, so $\mathbf{y}_t = (\mathbf{x}_t, \mathbf{u}_t, ..., \mathbf{u}_{t-n_h})$. Then, the cost of a node at the approximate next state $\mathbf{x}_{t+1} = \tilde{f}(\mathbf{x}_t, \mathbf{u}_t)$ is given as cumulative moving average (Equation 2).

$$c(E_t) = \tilde{e}_k, \tag{1}$$

$$c(\mathbf{x}_{t+1}) = c(\mathbf{x}_t) + \frac{c(E_t) - c(\mathbf{x}_t)}{t+1} \tag{2}$$

Although [14], [34] has a detailed description of each other component of the algorithm, we briefly describe the planning process here. At each iteration, the planning process samples a state x_{random}, and finds the nearest node on the planning tree - where "nearness" is a function of both state space distance as well the node's individual cost (using Equation 2). The selected node is then given a chance to propagate one of its candidate actions - here we prioritize actions which bring us closer to the goal by using an approximation of action's effect on the workspace (e.g., using a straight-line interpolation). At this point the transition model is queried to provide the next state, and the critic model is queried to provide the cost associated with this new edge. A new node is then added, and its set of candidate actions are generated by sampling randomly from the continuous action space, and applying a random duration to each sampled action. This process continues until either a) the planner expands a new node within the goal region, or b) the planner runs out of time. If the planner reaches the goal before the time limit, it then proceeds with a "branch-and-bound" process, which prunes any potential new edge from the tree which exceeds the current found solution cost. This, along with an optimistic heuristic, ensures that the planner is asymptotically-optimal with regards to its cost function.

An important component of [35] is the use of a heuristic function. In [14], a straight-line Euclidean distance to the goal was used. However in this work, such a heuristic would not match the cost function, and therefore was not suitable to be used. Several different heuristics were therefore attempted, such as assuming a minimal RMSE applied at each time step. Experimentally, many of these did not work well. Thus, to maintain the asymptotic-optimality property, we removed the heuristic (i.e. return 0), and to keep guidance towards reaching the goal we introduce goal-biasing during the state space sampling (10% of the time).



(a) (b)

Fig. 2. (a) RMSE of a neural-network model for the simulated hand with the increase of data size. The x-axis is the percentage of data used from the 1,631,225 data points collected through 719 episodes in the Gazebo environment. (b) Average error with regards to the number of action changes along a path segment taken by the critic, and for the Gazebo system.

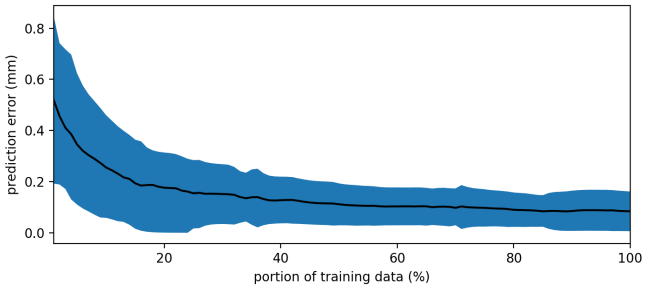


Fig. 3. Prediction accuracy with regards to the percentage of data (out of 978,735 data points) used to train the critic for the Gazebo system.

V. Evaluation

The method is evaluated using neural-network models over a physics-engine simulated adaptive hand, as seen in Figure 1a, and the real Model-T42 adaptive hand [5], as seen in Figure 1b. The compliance of the simulated hand in Gazebo was modeled given prior work [9].

A sufficient representation of the state of an underactuated hand is an observable 4-dimensional state composed of the object's position and the actuator loads [4]. The hand is controlled through the change of actuator angles, where an atomic action is, in practice, unit changes to the angles of the actuators at each time step. That is, an action moves actuator i with an angle of $\lambda\gamma_i$, where λ is a predefined unit angle and γ_i is in the range $[-1, 1]$. In the experiments below, the simulated and real hands were trained while grasping cylinders with $19.2\ mm$ and $35\ mm$ diameters, respectively.

A. Learned model evaluation

The experiments used recurrent feed-forward neural-networks to learn the transition models of both simulated and real systems. Both neural-networks have two hidden layers of 128 neurons each and ReLU activations along with a dropout of 10% to control overfitting. After experimenting with various architectures, this one yielded the most accurate predictions for both the systems. 1,631,225 transition points were collected in Gazebo over 719 random episodes. The focus is on evaluating the prediction accuracy with regards to the data size required. Figure 2a shows results for prediction error as the percentage of training data increases, where

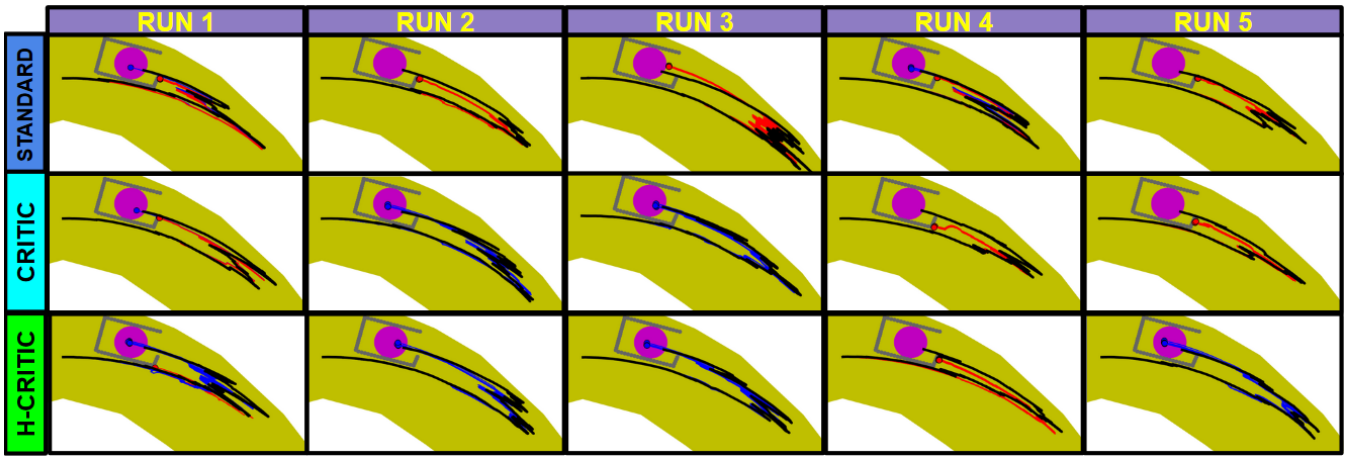


Fig. 4. Comparison of the different algorithms for the task of manipulating a cylindrical object into a goal region (magenta circle) between a "horseshoe" obstacle (shown in gray) using the underactuated hand in simulation. Each run consists of a planning phase followed by 10 rollouts of the planned path (shown as a black curve). Successful rollouts that reach the goal are shown as blue curves, while failed (i.e. colliding) rollouts are shown as red curves. The yellow region is the approximated workspace of the hand.

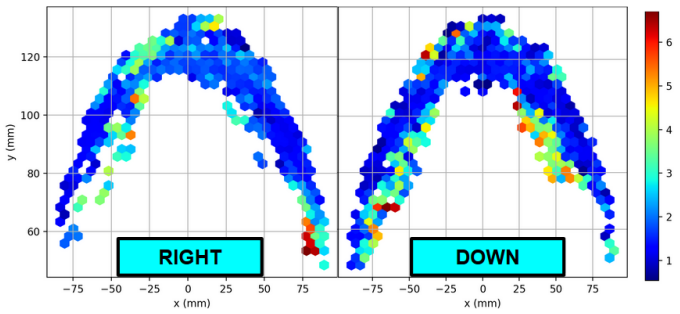


Fig. 5. Heatmap illustrations of the critic (for the simulated hand) error values projected on the $x-y$ plane with regards to different action directions.

increasing the data above 25% does not improve accuracy. This motivates the need to augment the model with a critic that assists in avoiding erroneous regions of $\mathcal{X}$. For the real system, 328,483 data points were collected over 364 episodes. The experiments below use neural-networks trained over only 40% of the data as their accuracy did not increase beyond that, leaving the rest for the critic.

B. Critic evaluation

To generate the critic, the first step is to evaluate the amount of data required for sufficient error prediction accuracy. Figure 3 shows the accuracy of the `H-CRITIC` prediction with the increase of data, out of the remaining 60% not used for training (978,735 transition points). The accuracy does not change significantly above 40% of the data. Figure 2b shows the actual model prediction error with regards to the number of action changes along the past and future path segments. This data only exists when including past actions, as in the `H-CRITIC`.

As seen in Figure 1a, the reachable subset of the $x-y$ workspace of an underactuated hand is banana-shaped. To get an understanding of the critic's representation on this space, Figure 5 provides the data in the critic as a heatmap. This illustrates the errors of the critic projected on the $x-y$ plane with regards to actions that will direct the object towards different directions. As expected, the error is lower in the inner region of the $x-y$ workspace where it is easier to manipulate. On the other hand, the errors are higher on the margins as they are harder to reach and collect motions of diverse actions.

C. Planning Experiments

This section first evaluates a physics-engine simulation of the adaptive hand across a variety of planning benchmarks for in-hand manipulation. Then, a demonstration of a peg-in-the-hole task on the real-hand is shown.

Algorithms: `STANDARD` uses the planning approach based on prior work [14], which does not integrate the proposed competency-aware models and optimizes a cost function based on path length. Both `CRITIC` and `H-CRITIC` use the planning approach described in Section IV-D, and optimize a cost function based on the critic error. `H-CRITIC` utilizes a horizon ($n_h = 40$) of its past actions as part of the query to the competency model. All methods make use of a learned state transition model and a failure classifier (both obtained from data), as discussed in Section IV-C.

Setup: All methods were evaluated on a single Intel Xeon E5-4650 processor with 8 GB of RAM. The planning approaches were given the models described above and tasked with computing a solution for reaching a goal region within a specified time limit (1,200 seconds). Solutions that were found within this time limit were subsequently rolled out 10 times each, recording whether the rolled out path reached the goal region or failed.

1) Physics-Engine Experiments: The first set of experiments, shown in Fig. 4, evaluate all three methods for a benchmark with a goal region hidden inside a set of obstacles, in the form of a 'horseshoe'. The purpose of this is to highlight the importance of minimizing RMSE, as in this case, there is limited clearance for the planner to reach the goal. The average success rates for `STANDARD`, `CRITIC` and `H-CRITIC` are 17.5%, 42% and 61.2%, respectively.

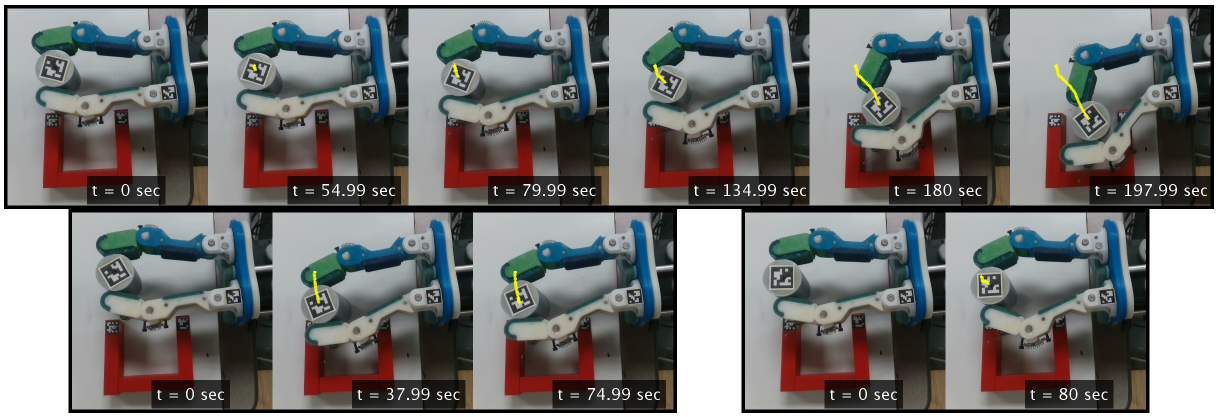


Fig. 6. Snapshots of the real hand experiments to manipulate the cylinder into the red horseshoe: roll-outs of (top) successful H-CRITIC planned path, (bottom left) standard planned path that collided and (bottom right) standard planned path that reached overload of the actuators.

TABLE I

ROLLOUTS RESULTS FOR PLANS IN THE SIMULATED SYSTEM

Goal	1		2		3		4		5	
	STA.	H-CRITIC	STA.	H-CRITIC	STA.	H-CRITIC	STA.	H-CRITIC	STA.	H-CRITIC
path length (mm)	111	87.42	206	91.33	169	175	97.5	93.3	83	64.9
rollout suc. rate (%)	0	20	50	80	0	100	0	90	100	100
RMSE (mm)	NA	2.27	3.61	2.16	NA	5.06	NA	2.57	1.45	1.21

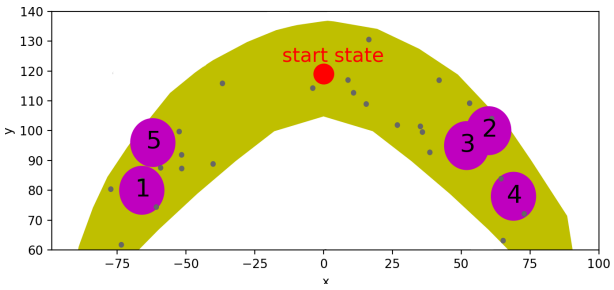


Fig. 7. Five different random goal regions (magenta circle) with random obstacles (gray dots) used for the evaluation of the STANDARD and H-CRITIC on the simulated system. Results are shown in Table I.

The results indicate that the critic provides an advantage for the planned paths over the traditional shortest-path cost function. Furthermore, the usage of the history actions in H-CRITIC over CRITIC is beneficial.

The second set of experiments defines five random goals within the workspace of the hand, with a set of random obstacles, and evaluates the STANDARD and H-CRITIC approaches. The objective is to evaluate the effectiveness of the methods for planning in different portions of the workspace. To make it more challenging, the neural-network model was trained over only 1% of the data which, as seen in Figure 2a, has higher average error. A corresponding critic was generated for it with only 20% of the remaining data.

For each goal, the planning methods were executed twice, and for each of these runs the planned path was rolled out 10 times. The setup of the goals and obstacles can be seen in Figure 7, with detailed statistical results for each goal shown in Table I. In the first four goals, the H-CRITIC was superior compared to the STANDARD approach with higher success rate. Goal 5 is in the low error region with not much interference from the obstacles. Thus, the results for the trial are equivalent to the STANDARD with a slight accuracy advantage to the H-CRITIC.

2) Real Hand Demonstration: The real hand demonstration replicates the 'horseshoe' setup, where the hand must manipulate the cylinder to the inside. Here again, paths are planned by the STANDARD and H-CRITIC methods. Figure 6 shows snapshots of one successful rollout of a plan with H-CRITIC and two failed ones planned with the STANDARD. The motion is quite slow as we assume quasi-static motion. As a result of inaccurate model predictions, plans with the STANDARD approach tended to collide with the obstacles or to pull the object too much toward the hand's base resulting in the actuators overloading. In contrast, plans with the H-CRITIC were tracked more accurately (RMSE of 2.63 mm for the demonstrated one) due to the minimization of critic error, and were therefore more successful overall in reaching the goal region.

VI. CONCLUSION

This work proposes an independent critic model to augment a given transition model and to improve accuracy. Instead of attempting to improve the transition model with more data, which often is unsuccessful, the method uses the surplus data to train a critic for evaluating the accuracy of the original model. The critic uses a history of prior actions along with the intended action from the current state to estimate the prediction error. A sampling-based planner integrates the critic into its cost function to direct solutions to regions of accurate predictions. A key future direction is the integration of the critic into belief-space planning [14], where uncertainty can be learned with the proposed method.

REFERENCES

[1] L. U. Odhner and A. M. Dollar, "Stable, open-loop precision manipulation with underactuated hands," *Int. J. of Rob. Res.*, vol. 34, no. 11, pp. 1347–1360, Sep 2015.

[2] B. Calli and A. M. Dollar, "Vision-based precision manipulation with underactuated hands: Simple and effective solutions for dexterity," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2016, pp. 1012–1018.

[3] B. Calli, A. Kimmel, K. Hang, K. Bekris, and A. Dollar, "Path planning for within-hand manipulation over learned representations of safe states," in *International Symposium on Experimental Robotics*, Buenos Aires, Argentina, 2018.

[4] A. Sintov, A. S. Morgan, A. Kimmel, A. M. Dollar, K. E. Bekris, and A. Boularias, "Learning a state transition model of an underactuated adaptive hand," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1287–1294, April 2019.

[5] R. R. Ma and A. M. Dollar, "Yale openhand project: Optimizing open-source hand designs for ease of fabrication and adoption," *IEEE Rob. & Aut. Mag.*, vol. 24, pp. 32–40, 2017.

[6] Z. Littlefield, D. Klimenko, H. Kurniawati, and K. E. Bekris, "The importance of a suitable distance function in belief-space planning," in *International Symposium on Robotic Research (ISRR)*, A. Bicchi and W. Burgard, Eds., 2015, pp. 683–700.

[7] T. Laliberté and C. M. Gosselin, "Simulation and design of underactuated mechanical hands," *Mech. and Mach. The.*, vol. 33, no. 1-2, pp. 39–57, Jan 1998.

[8] L. U. Odhner and A. M. Dollar, "Dexterous manipulation with underactuated elastic hands," in *IEEE Int. Conf. on Rob. and Aut.* IEEE, May 2011, pp. 5254–5260.

[9] A. Rocchi, B. Ames, Z. Li, and K. Hauser, "Stable simulation of underactuated compliant hands," in *ICRA*, May 2016, pp. 4938–4944.

[10] I. Hussain, F. Renda, Z. Iqbal, M. Malvezzi, G. Salvietti, L. Seneviratne, D. Gan, and D. Prattichizzo, "Modeling and Prototyping of an Underactuated Gripper Exploiting Joint Compliance and Modularity," *IEEE Rob. and Aut. Let.*, vol. 3, no. 4, pp. 2854–2861, Oct 2018.

[11] D. Prattichizzo, M. Malvezzi, M. Gabiccini, and A. Bicchi, "On the manipulability ellipsoids of underactuated robotic hands with compliance," *Rob. and Aut. Sys.*, vol. 60, no. 3, pp. 337 – 346, 2012.

[12] G. Grioli, M. Catalano, E. Silvestro, S. Tono, and A. Bicchi, "Adaptive synergies: An approach to the design of under-actuated robotic hands," in *IEEE/RSJ Int. Conf. on Intel. Rob. and Sys.* IEEE, Oct 2012, pp. 1251–1256.

[13] J. Borras and A. M. Dollar, "A parallel robots framework to study precision grasping and dexterous manipulation," in *IEEE Int. Conf. on Rob. and Aut.* IEEE, May 2013, pp. 1595–1601.

[14] A. Kimmel, A. Sintov, B. Wen, A. Boularias, and K. Bekris, "Belief-space planning using learned models with application to underactuated hands," in *International Symposium on Robotics Research (to appear)*, 2019.

[15] H. van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *IEEE-RAS Int. Conf. on Hum. Rob.*, Nov 2015, pp. 121–127.

[16] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics," *J. of Intel. & Rob. Sys.*, vol. 86, no. 2, pp. 153–173, May 2017.

[17] S. Zhu, D. Surovik, K. E. Bekris, and A. Boularias, "Efficient model identification for tensegrity locomotion," in *IEEE International Conference on Intelligent Robots and Systems , IROS, Madrid, Spain*, 2018.

[18] S. Zhu, A. Kimmel, K. E. Bekris, and A. Boularias, "Fast model identification via physics engines for improved policy search," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden*, 2018.

[19] A. Boularias, J. A. Bagnell, and A. Stentz, "Learning to manipulate unknown objects in clutter by reinforcement," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, 2015, pp. 1336–1342. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9360

[20] R. Paolini, A. Rodriguez, S. S. Srinivasa, and M. T. Mason, "A data-driven statistical framework for post-grasp manipulation," *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 600–615, 2014.

[21] R. Paolini and M. T. Mason, "Data-driven statistical modeling of a cube regrasp," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 2554–2560.

[22] F. Reinhart, Z. Shareef, and J. Steil, "Hybrid analytical and data-driven modeling for feed-forward robot control," *Sensors*, vol. 17, 02 2017.

[23] R. Michelmore, M. Kwiatkowska, and Y. Gal, "Evaluating Uncertainty Quantification in End-to-End Autonomous Driving Control," *arXiv e-prints*, Nov. 2018.

[24] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[25] L. Smith and Y. Gal, "Understanding Measures of Uncertainty for Adversarial Example Detection," in *UAI*, 2018.

[26] A. Yamaguchi and C. G. Atkeson, "Neural networks and differential dynamic programming for reinforcement learning problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5434–5441.

[27] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2017.322

[28] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, The MIT Press, 2005.

[29] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[30] G. Maeda, M. Ewerton, T. Osa, B. Busch, and J. Peters, "Active incremental learning of robot movement primitives," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 37–46.

[31] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learn. Res.*, vol. 9, pp. 235–284, June 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1390689

[32] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059.

[33] B. Calli, K. Srinivasan, A. Morgan, and A. M. Dollar, "Learning modes of within-hand manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3145–3151.

[34] Z. Littlefield and K. E. Bekris, "Efficient and asymptotically optimal kinodynamic motion planning via dominance-informed regions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[35] ——, "Informed asymptotically near-optimal planning for field robots with dynamics," in *Field and Service Robotics*. Springer, 2017, pp. 449–463.