Template paper for the Robotics: Science and Systems Conference

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—TODO**

I. Introduction

Learning from Demonstrations (LfD) [**?** **?**] has demonstrated promising results in a variety of robotics applications including personal robotics [**?**], autonomous helicopter control [**?**], and control of humanoid robots [**?**]. In a typical LfD setting, a robot is provided a set of *demonstrations* which are ordered sequences of state-action pairs. From these demonstrations, the goal is to learn a *policy* a function of the current state which returns an action.

However, set of all state-actions pairs is usually infinite. As demonstrations are often human-initiated, and thus, relatively small in number, learning a policy over an infite dimensional state-action space is infeasible. A recent approach is to parameterize the action space with a finite generating set of action *primitives* [**?** **?** **?** **?**]. In the literature, this approach has been refered to as motion primitive learning, sparse coding, and dictionary learning. LfD can learn a policy over the set of primitives, and then *generalize* this policy by encoding new observations in terms of already learned primitives. There are many variants of this approach (see Survey [**?**]), and there is also strong connection to recent results in Reinforcement Learning using Policy Gradients [**?**] which find optimal policies by applying a Stochastic Gradient-like method to find optimal parameterized policies.

In this work, we apply these recent results in LfD to surgical robotics. Increasingly, the surgical robotics community has studied automating surgical subtasks [**?** **?** **?** **?**]. This focus on autonomous execution marks a transition for surgical robots, such as the Da Vinci and the Raven, which originally designed for tele-operation. There is a wealth of knowledge in the surgical community on precise and accurate tele-operation of these platforms, and expert surgeons are often very skilled and familiar with the tools. Our goal is to build a learning framework that can learn primitives from these expert demonstrations, and help make autonomous execution of surgical subtasks more robust. In fact, the seminal work in surgical primitive learning (also called Surgemes) [**?**] was not initially motivated with autonomous execution. Lin et al. proposed a Bayesian classification framework to model surgical suturing in terms of discrete actions; with the goal of evaluating surgical skill based on expert demonstrations.

The results of Lin et al. introduce some of key domain-specific challenges in surgical primitive learning: (1) demonstration trajectories were heavily pre-processed before learning with de-noising and dimensionality reduction, (2) there was significant gap between expert demonstrations and non-expert demonstrations, and (3) a majority of segmentation errors happened at transition points. Building on these results and taking these challenges to heart, in this work, we approach the problem of surigical primitive learning from a perspective of robust execution. **What are our contributions?**

II. Background and Problem Statement

In this paper, we address the following problems: (1) parameterize the action space in terms of a generating set of primitive motions, (2) reconstruction of new trajectories in terms previously learned primitives, and (3) finite-state machine learning.

A. Definitions and Notation

SK: What is the typical level of formality in a RSS paper? I made it as formal as made sense we can dial it back if we think this is unneccessary Segmentation without an objective function is an ill-defined operation. In the literature, many different techniques have been proposed; optimizing widely different criteria. In this section, we try to abstract of the key components to avoid an apples to oranges comparison of different approaches. We also introduce the notation that we will use throughout the paper.

Let $\mathcal{D} = \{d_1, d_2, ..., d_n\}$ be a set of demonstrations. Each demonstration is a ordered sequence of T_i observations of a d-dimensional state $x(t_i) \in \mathbb{R}^d$. Thus, each $d_i \in \mathbb{R}^{d \times T_i}$. Let $\omega(d, s)$ be the windowing operation; that it applies a sliding window of size s to d and breaks d into a set of smaller observations. Formally, ω maps every d_i to a set of $T_i - s$ sequences of observations of size $\{\mathbb{R}^{d \times s}\}$. We define the windowing operation Ω on the full set of demonstrations to be:

$$\Omega(\mathcal{D}, s) = \cup_i^n \omega(d_i, s)$$

SK: most specific definition that can encompass DS3, Submodularity, and BP

Definition 1: Set of Motion Primitives. We represent the set of all possible windows as Θ. Θ is a semigroup that has an operation:

$$* : \Theta \mapsto Theta$$

called the combination operation that takes two windows as input and maps them to another window. This operation is commutative and associative. Under this semigroup $< \Theta, * >$, the set of primitves $\mathcal{P}$ is a generating set for Θ. That is,

$$\forall \theta \in \Theta \exists p \subset \mathcal{P} : p_1 * p_2 * p_3 * ... = \theta$$

This definition, is intentionally pedantic and broad, and we will find that in fact different trajectory segmentation methods in the literature make different assumptions with respect to this definition to make their computation tractable.

- The DS3 model restricts Θ to only those sliding windows that are observed. The combination operation $*$ in this model is a rotation invariant convex convex combination.
- The Beta Process model relaxes the assumption on Θ treating each window as a sample from an infinitely supported continuous probability distribution. However, it additionally adds the restriction on $*$ that it can only be a Euclidean addition.
- The Submodular approach, like the DS3 model, restricts Θ to only those sliding windows that are observed. Where the $*$ operation is a union all other elements within some ϵ of similarity.

An important point to note is that these algorithms require a definition of similarity between two elements in Θ, which we abstract away in our semigroup model of the motion primitive. We treat that operations as external to the combination operator $*$. Formalizing this notion of similarity becomes imporant when we want to define reconstruction. Metrics are imporant when we want to discuss the optimality of a primitive representation, as there may be many possible generating sets. The concept of reconstruction gives us a framework to analyze this, however it requires us to formalize reconstruction in terms of a metric space on Θ.

Definition 2: Reconstruction. Define a new operation:

$$d : \Theta \times \Theta \mapsto \mathbb{R}$$

If d is a metric, then $< \Theta, d >$ is a metric space. For a $t \notin \Theta$, we can define a projection onto Θ:

$$e = \min d(t, \Theta)$$

Since the projection exists in Θ, it can be constructed from the generating set of primitives. Therefore, the *Reconstruction Error* of t is e; the closest the generating set can come to reconstructing t.

This definition of reconstruction gives informs us on how we should evaluate differing approaches. If we ensure that the distance metric is consistent, then the projection error measures a consistent value. **SK: problem with the definitions below:** However, there are nuances as different approaches to segmentation actually define Θ differently. There might be only a subset of the observed Θ that is exactly represented by the generating set of primitives.

Finally, we need to introduce a notion of temporality. We model temporality in a Markovian way, where every next action (trajectory window) is conditionally independent with history given the current state.

Definition 3: Markov Chain of Primitives. Given a set of primitives $\mathcal{P}$. Let Z be a binary vector of dimension $\mid \mathcal{P} \mid$:

$$Z \in \mathbb{Z}_2^{|\mathcal{P}|}$$

. Given a demonstration d_i and an *ordered* sequence of windows $\omega(d_i, s)$. For each d_i, there exists an "active" subset of $\mathcal{P}$ such that d_i is generated from that subset of $\mathcal{P}$. If we assign each $p \in \mathcal{P}$ an index, then we can represent the *state* of each d_i with Z (one if active, zero if not). Then, for the $2^{|\mathcal{P}|}$ states, we learn a Markov Chain based on the observed transitions.

SK problem! Obviously, learning a state machine, whether probablistic or deterministic, faces an immense problem with the curse of dimensionality. For a set of k primitives, there are 2^k possible states the in which the system may be.

B. Problem Statement

1) Problem 1. Trajectory Segmentation: We take as input a set of demonstrations $\mathcal{D}$. The output is a set of motion primitives as defined in the previous sub-section. **SK: explain why we should care....**

2) Problem 2. Reconstruction: We take as input a set of primitives $\mathcal{P}$ and a new trajectory t. The output is a reconstruction of t with the elements in $\mathcal{P}$ that minimizes reconstruction error as defined before. **SK: explain why we should care....**

3) Problem 3. Markov Chain Learning: We take as input a set of primitives $\mathcal{P}$ and a set of demonstrations $mathcalD$. The output is a Markov chain showing the state transition probabilities. This can be turned into an FSM by taking the most likely path through the chain. **SK: explain why we should care....**

III. Beta Process Models For Segmentation

The Beta-Bernoulli model is a Bayesian non-parametric approach to assign a set of observations to a set of features [cite]. One of key aspects of this approach is it allows for clustering with *mult-tenancy*; that is observations may be members of multiple clusters. This approach has been applied the segmentation of dynamical trajectories [cite], however, has not yet been applied to identify kinematic primitives.

The kinematic case poses a variety of new challenges. First, we need to explictly make the model invariant to rotations. The next challenge is that kinematic primitives are more sensitive to time scales and thus we need to be careful with how we model the windowing operations and how those windows are allowed to interact with other properties such as rotation invariance.

The standard recipe for Bayesian approaches is to define a generative model; a plausible parameterized statistical model that generates the observed data. Then conditioned on the observations infer the expected model.

SK: The notation below is inconsistent with the stuff I added

A. Generative Model For Demonstration-Trajectory Relationship

We first look at the case for fixed K number of motion primitives. In this case, we assume the demontration are created by the following generative model for each demonstration i:

1) Draw $T_i \sim Poisson(\gamma)$

2) Draw $\theta \sim Beta(\alpha)$
3) Draw $R \sim N(I, \sigma I)$
4) For $1..t..T_i$
    a) For 1..k..K
        i) Choose $z_{ik} \sim Ber(\theta)$
    b) $\tau_i[t] = R(\sum_k z_{ik} p_k) + \epsilon$
    c) $\epsilon \sim N(0, \sigma I)$

Now, if we allow $K \to \infty$, that is a potentially infinite number of primitives, this model becomes a Beta Process model:

1) Draw $T_i \sim Poisson(\gamma)$
2) Draw $B \sim BP(1, B_0)$
3) Draw $R \sim N(R_0, \sigma I)$
4) For $1..t..T_i$
    a) $Z \sim BeP(B)$
    b) $\tau_i[t] = R(\sum_k^{\infty} Z_{ik} p_k) + \epsilon$
    c) $\epsilon \sim N(0, \sigma I)$

Due to the isotropic nature of the generative model, this can be solved using a variant of BP-Means proposed by Broderic, Kulis, and Jordan 2012 [cite]. {TODO: A relaxation is needed to make the translation invariance work}

B. Parameter Inference For $\mathbb{P}$

As in [cite], let $1...K^+$ index the current set of primitives $\mathbb{P}$.

1) For all demonstrations
    a) Calculate expected R: via orthonormally constrained least squares (as in Mahler et al.) fit a rigid transformation to all $\tau \in x_i$,$p \in \mathbb{P}$ and take the mean.
    b) For all trajectories within a demonstration
        i) Choose subset of primitives that sum to τ
        ii) Add current trajectory to set of primitives, if it improves the objective by more than λ^2 then include it else drop it.
    c) Update set of primitives via least squares.

ACKNOWLEDGMENTS