The International Journal of Robotics Research
XX(X):1–15
©The Author(s) 2016
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Transition State Clustering: Unsupervised Surgical Trajectory Segmentation For Robot Learning

Sanjay Krishnan*[1], Animesh Garg*[1], Sachin Patil[1], Colin Lea[2], Gregory Hager[2], Pieter Abbeel[1], Ken Goldberg[1]

Abstract

We present a new unsupervised segmentation algorithm, Transition State Clustering (TSC), which combines results from hybrid dynamical systems and Bayesian non-parametric statistics to segment kinematic recordings of robotic surgical procedures. TSC treats each demonstration trajectory as a noisy observation of an underlying switched linear dynamical system (SLDS) and clusters spatially and temporally similar transition events (i.e., switches in the linear regime). TSC uses a hierarchical Dirichlet Process Gaussian Mixture Model to avoid selecting the number of segments *a priori*. We compare TSC to five alternatives on the respective algorithms' correspondence to a known ground truth in a synthetic example: Gaussian Mixture Model, Gaussian Hidden Markov Model, Coresets, Gaussian Hidden Semi-Markov Model, and an Autoregressive Hidden Markov Model. We find that when demonstrations are corrupted with process and observation noise, TSC recovers the ground truth 49% more accurately than alternatives. Furthermore, TSC runs 100x faster than the Autoregressive Models which require expensive MCMC-based inference. We also evaluated TSC on 67 recordings of surgical needle passing and suturing. We supplemented the kinematic recordings with manually annotated visual features denoting grasp and penetration conditions. On this dataset, TSC finds 83% of needle passing transitions and 73% of the suturing transitions annotated by human experts.

Experimental code and datasets are available at: http://berkeleyautomation.github.io/tsc/

Keywords

Trajectory Segmentation, Surgical Robotics, Robot Learning

Introduction

The adoption of robot-assisted minimally invasive surgery (RMIS) is generating datasets of kinematic and video recordings of surgical procedures. This data can facilitate robot learning from demonstrations (Murali et al. 2015), surgical training and assessment (Reiley et al. 2010; Gao et al. 2014), and automation (Kehoe et al. 2014; Mahler et al. 2014). Segmenting this demonstration data into meaningful sub-trajectories can benefit learning since individual segments are often less complex, have lower variance, and it is easier to remove outliers. However, even in a consistent data collection environment, such as tele-operation on identical tissue phantoms, surgical trajectories can vary significantly both spatially and temporally. These trajectories are further corrupted by random noise, spurious motions, and looping actions where a surgeon repeatedly retries a motion until success. The primary challenge in surgical trajectory segmentation is to identify consistent segments across a dataset of demonstrations of the same procedure in the presence of such disturbances.

Segmentation algorithms fall into two broad categories: (1) supervised approaches that learn from manual annotations or match sub-sequences to pre-defined dictionaries of primitives (Lin et al. 2005; Varadarajan et al. 2009; Tao et al. 2013; Lea et al. 2015), and (2) unsupervised approaches that infer the latent parameters of some underlying generative process (Calinon et al. 2010; Lee et al. 2015; Krüger et al. 2012; Niekum et al. 2012). Consistency and supervisory burden are a key concern in supervised segmentation as it is often difficult to precisely characterize what defines a segment and labeling can be time-consuming. Similarly, it may be unclear how to specify a dictionary of primitives at the correct level of abstraction. To the best of our knowledge, prior work in the surgical setting has been supervised, and we draw from several studies of unsupervised segmentation in non-surgical settings to develop a new unsupervised approach (Calinon et al. 2010; Lee et al. 2015; Krüger et al. 2012; Niekum et al. 2012). Unsupervised approaches largely apply clustering or local regression models to identify locally similar states. Transition State Clustering (TSC) extends this work with a two-phase algorithm that first identifies **transitions states**, defined as consecutive time-steps assigned to different segments, and then, **clusters** spatially and temporally similar transition states across demonstrations with a non-parametric mixture model.

This paper focuses on segmenting trajectories derived from kinematic and video recordings of surgical robots in

* Authors contributed equally
[1]University of California, Berkeley
[2]Johns Hopkins University

Corresponding author:
Sanjay Krishnan
Email: sanjay@eecs.berkeley.edu

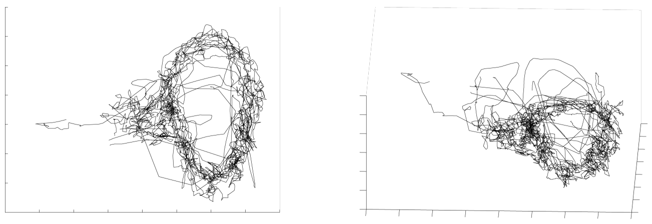

Figure 1. We plot 10 trajectories of the end-effector (x,y,z) positions on an identical circle cutting task on the dVRK. This plot illustrates the variability of demonstrations even when the task is identical. The goal of TSC is to identify consistent transition structure in the demonstrations, and our experiments in Section illustrate the results.

tele-operation. TSC is also relevant to problems in other domains, but those applications are outside of the scope of this paper. Consider two trajectories $\mathbf{x}_1$ and $\mathbf{x}_2$. $\mathbf{x}_1$ and $\mathbf{x}_2$ may represent two different linear paths to the same needle insertion transition. That needle insertion is a invariant across $\mathbf{x}_1$ and $\mathbf{x}_2$. TSC attempts to infer such invariants using spatial and temporal clusters of transitions detected by changes in motion. The crucial insight is that the sequence of transition events often has a consistent partial order across demonstrations–even if the motion between those transition events is very different. This model can also be coupled with a series of merging and pruning steps to ensure that only the most important transition state clusters are retained.

Contributions and Main Results

We present a new unsupervised segmentation algorithm called Transition State Clustering (TSC), which given a set of discrete-time trajectories identifies a common transition structure to segment the trajectories. We treat each demonstration trajectory as generated from a locally linear dynamical system with i.i.d process noise $\mathbf{w}_t$:

$$\mathbf{x}_{t+1} = A_t\mathbf{x}_t + \mathbf{w}_t : A_t \in \{A_1, ..., A_m\} \tag{1}$$

A transition states is a state at which the dynamics matrix changes at the next time-step $A_t \neq A_{t+1}$. We model the set of all transition states as sampled from a probability density generated from a hierarchical nonparametric Bayesian model, where the number of regions are determined by a Dirichlet Process. A series of merging and pruning steps remove outliers.

We evaluate TSC on real surgical data from the JIGSAWS surgical training dataset consisting of joint-space trajectories and video from a fixed camera (Gao et al. 2014), and a synthetic example consisting of randomly generated motions of a point robot with variable levels of noise. On the synthetic examples, we evaluate the ability of five alternative algorithms, Gaussian Mixture Model, Gaussian Hidden Markov Model, Coresets, Gaussian Hidden Semi-Markov Model, and a Autoregressive Hidden Markov Model, to return a segmentation that has a strong one-to-one correspondence with a known ground truth. Our experimental results suggest that TSC recovers this ground truth with greater accuracy than the alternatives, especially under low-frequency process noise. For the surgical experiments, TSC extends to state-spaces the include derived features from computer vision which are manually derived in this work. This paper presents the TSC model, and presents experiments in which we manually label the video stream with two features: a binary variable identifying object grasp events and a scalar variable indicating surface penetration depth. On 67 kinematic and video of surgical needle passing and suturing tasks from the JIGSAWS surgical training dataset (Gao et al. 2014) TSC finds 83% of needle passing transitions and 73% of the suturing transitions found by human experts.

This paper is a substantially revised and expanded version of Krishnan et al. (2015). This version includes several new experiments evaluating TSC against 5 alternative algorithms (Figure 2-6), an expanded technical discussion about the model, and revised accuracy metrics consistent with recent segmentation work in robotics and computer vision.

Background and Related Work

TSC exploits the structure of repeated demonstrations by first identifying transitions and then correlating them spatially and temporally across different demonstrations.

Segmentation in Surgical Robotics

To the best of our knowledge, prior work in surgical robotics has only considered supervised segmentation using either segmented examples or a pre-defined dictionary called surgemes. For example, given manually segmented videos, Zappella et al. (2013) use features from both the videos and kinematic data to classify surgical motions. Similarly, Quellec et al. (2014) use manually segmented examples as training for segmentation and recognition of surgical tasks based on archived cataract surgery videos. Several studies use the surgemes to bootstrap learning segmentation, however, this involved time-consuming the process of identifying surgemes in existing data sources for use as training and testing data (Lin et al. 2005; Varadarajan et al. 2009; Tao et al. 2013; Lea et al. 2015)).

Unsupervised Segmentation Models

Unlike supervised segmentation techniques, unsupervised techniques do not use labels or dictionaries. As such, a key differentiating factor is the underlying probabilistic model for segmentation. Consider a continuous time vector-valued trajectory, which is a sequence of T vectors $\mathbf{x}_t$ in some vector-space $\mathbb{R}^p$.

Gaussian Mixture Models: Many unsupervised segmentation techniques are based on Gaussian Mixture models (GMM). GMMs are particularly intuitive since Ghahramani and Jordan (1993) showed that GMMs are a form of local linear regression; linearizing around the mixture means.

Lee et al. (2015) identify segmentation points in a trajectory by fitting a GMM directly to the data. That is, the sequence $[\mathbf{x}_1, ..., \mathbf{x}_T]$ is modeled as a sample from a GMM, and they assign each $\mathbf{x}_t$ to the most likely mixture component. They tune the number of mixture components using the Bayesian Information Criterion, and apply PCA dimensionality reduction before applying the GMM for tractability. By treating the sequence as a sample from a GMM, this approach does not consider dynamics and correlation between demonstrations.

One solution is to draw from the dynamical systems literature (Khansari-Zadeh and Billard 2011; Moldovan et al. 2015) and model the trajectory $\mathbf{x}_t$ as a noisy autonomous dynamical system:

$$\dot{\mathbf{x}} = \xi(\mathbf{x}) + \mathbf{w}$$

It can be shown to linearize this system in the same way as in (Lee et al. 2015), but instead of applying the GMM to the set of states $\mathbf{x}_t$, we have to apply a GMM to samples of $\binom{x}{\dot{x}}$ or $\binom{\mathbf{x}_t}{\mathbf{x}_{t+1}}$ in discrete-time. This approach is used by Krüger et al. (2012), and like TSC they also use a Dirichlet process prior to learn locally linear dynamics. Transition State Clustering builds on these GMM approaches with a two-phase algorithm that first identifies transitions states in a way similar to Krüger et al. (2012) and then, clusters spatially and temporally similar transition states across demonstrations with a non-parametric mixture model under the assumption that each demonstration follows the same partial order of transitions up-to noise and loops.

Hidden Markov Models: One approach to deal with spatial and temporal variation is to model the high-level progression of a task as a finite-state Markov Chain (Asfour et al. 2008; Calinon and Billard 2004; Kruger et al. 2010; Vakanski et al. 2012), for example, primitive A progresses to B with probability 0.75 and to C with probability 0.25. Such a model is class of Hidden Markov Models since this Markov chain is not directly observed. Given the current state of this Markov chain, the system will either "emit" different states (e.g., Gaussian HMM) or have different dynamics (e.g., Autoregressive HMM). This logic has been extended to more complex transition dynamics such as the Hidden Semi-Markov Model, which additionally models the amount of time spent in a given state (Tanwani and Calinon 2015). There is also the related "workflow" HMM proposed by Padoy et al. (2009). The HMM, HSMM, and their variants impose a probabilistic grammar on transitions, and the inference algorithm estimates the transition probabilities from data. Accordingly, they can be sensitive to hyper-parameters such as the number of segments, the amount of data, and noise (Tang et al. 2010). The problem of robustness in GMM+HMM (or closely related variants) has been addressed using down-weighting transient states (Kulić and Nakamura 2008) and sparsification (Grollman and Jenkins 2010; Mohan et al. 2011).

There have been several Bayesian extensions to these models, which model the time-series as a stochastic process and learn the parameters with MCMC or Stochastic Variational Inference. Willsky et al. (2009) proposed Beta Process-Autoregressive-Hidden Markov Model, which was applied by Niekum et al. (2012) in robotics. This model is fits an autoregressive model to time-series, where $\mathbf{x}_{t+1}$ is a linear function of states $\mathbf{x}_{t-k}, \ldots, \mathbf{x}_t$. The linear function switches according to an HMM with states parametrized by a Beta-Bernoulli model (i.e., Beta Process). While HMMs and GMMs also draw from the Bayesian literature, they differ from recently proposed Bayesian segmentation models as they are typically solved more efficiently with analytic expectation maximization algorithms. Similarly, TSC is motivated by Bayesian non-parametrics, there are several features of the algorithm that are motivated by frequentist statistics (e.g., outlier rejection using merging and pruning).

We evaluate against these approaches (GMM+HMM, ARHMM, and HSMM) in our experiments and find that TSC is more robust to un-modeled noise. The intuition is that Hidden Markov approaches make the implicit assumption that "the low-level dynamics within a segment are more structured and predictable than the higher-level dynamics that govern transitions between segments" (Saeedi et al. 2016). In contrast, TSC explores the converse, suppose the low-level dynamics are uncertain and noisy, but the high-level dynamics follows a consistent partial order of events across demonstrations and these events are spatially and temporally correlated.

Coresets: A coreset is defined as a query-dependent compression of a dataset D, such that running the query q results in a provably approximate result. This idea can be used to devise a locally linear segmentation technique Rosman et al. (2014); Sung et al. (2012); Volkov et al. (2015); Rosman et al. (2014). The query q is the solution to the k-line segment problem, which fits a k line segments to a trajectory. The idea is to find a compressed dataset such that the lines can be accurately reconstructed. The main benefit is that this results in segmentation with provable properties, such as sample complexity and convergence. However, the models used in prior work do not consider switched linear dynamics, non-parametrics when choosing k, or robustness to loops.

Problem Setup

This section describes the problem setting, assumptions, and notation.

Demonstrations

Let $D = \{d_i\}$ be a set of demonstrations. Each demonstration d_i is a discrete-time sequence of T_i state vectors in a state-space $\mathbb{R}^p$. Associated with D is a set of k transitions which are informally defined as state-space and temporal conditions that trigger a change in motion, in the following section, we will precisely characterize this definition. Thus, each demonstration d_i can be represented as a sequence of labeled transitions $\{1,...,k\}$, e.g., $d_1 = [S_1, S_2, S_4]$, $d_2 = [S_3, S_1, S_4]$. The goal of Transition State Clustering is to learn the sequence of transitions that consistently occur across all demonstrations e.g., $[S_1, S_4]$, and associate them with states in a trajectory.

Regularity

Without regularity assumptions on the demonstrations, there may not be a meaningful common structure. For example, we could observe:

$$d_1 = [S_1, S_3, S_5], d_2 = [S_2, S_4, S_6],$$

where there does not exist any overlap between d_1 and d_2. Therefore, we assume, the set of demonstrations is *regular*, meaning there exists a non-empty sequence of transition U^* such that the partial order defined by the elements in the sequence (i.e., S_1 happens before S_2 and S_3) is satisfied by every U_i. For example,

$$U_1 = [S_1, S_3, S_4],\ U_2 = [S_1, S_1, S_2, S_4],\ U^* = [S_1, S_4]$$

An example of an irregular demonstration set is

$$U_1 = [S_1, S_3, S_4],\ U_2 = [S_2, S_5],\ U^* \text{ no solution}$$

Intuitively, this condition states that there have to be a consistent ordering of transitions over all demonstrations up to some additional transitions (e.g., spurious actions). We will show that we can extend this condition such that only a fraction ρ of the demonstrations need to be regular; thereby *pruning* the inconsistent transitions.

Looping

Loops, or repetitions of an action until the desired outcome, are common in surgical demonstrations. For example, a surgeon may attempt to insert a needle 2-3 times. For example, let us assume that a surgeon is attempting to insert a needle and fails to do so 2 times. If in all other demonstrations, there is only a single transition representing needle insertion (i.e., transition “1”), we might detect multiple transitions S'_1 and $S^{\dagger}_1$ demonstration with loops. Ideally, we would like to compact these repeated motions into a single transition:

$$U_1 = [S_1, S_3, S_4],\ U_2 = [S_1, S'_1, S^{\dagger}_1, S_3, S_4],\ U^* = [S_1, S_3, S_4]$$

We assume that these loops are modeled as repeated transitions, which is justified in our experimental datasets. This assumption may seem at odds with our argument that surgical demonstrations are highly variable. We find that while the motions between transitions are variable and noisy, up-to loops and extra transitions, the high-level sequence of transitions is relatively consistent. In the future, we hope to explore more complex models for failure and retrial, and we believe that variants of our approach can be applied in conjunction with Hidden Markov Models.

Problem Statement

In this paper, we focus on segmentation for surgical robotics applications on the da Vinci surgical robot. However, we do believe that TSC is broadly applicable to other robotics domains and data collected from other platforms. Other than the regularity assumptions above and the implicit assumptions about local linearity discussed in the next section, we make no assumptions about the nature of the trajectories given to TSC. In our experiments, we define the state-space to be the 6-DOF orientation of the robot end-effector.

One challenge is that the kinematic state of the robot may not be sufficient to describe the system or its interaction with the environment. We additionally demonstrate TSC with features constructed from video. Suppose at every time t, there is a feature vector v_t composed of discrete and continuous features. Then, we define an augmented state of both the robot kinematic state and the features denoted is:

$$\mathbf{x}_t = \begin{pmatrix} c_t \\ v_t \end{pmatrix}$$

Consider the following features:

1. *Grasp*. 0 if empty, 1 other wise.
2. *Needle Penetration*. We use an estimate of the penetration depth based on the robot kinematics to encode this feature. If there is no penetration (as detected by video), the value is 0, otherwise the value of penetration is the robot's z position.

Our goal with these features was to illustrate that TSC applies to general state-spaces as well as spatial ones, and not to address the perception problem. These features were constructed via manual annotation, where the Grasp and Needle Penetration were identified by reviewing the videos and marking the frames at which they occurred. We completely characterize TSC without such features on a synthetic dataset against alternatives, and for the surgical data present results with and without these features.

Problem 1. Transition State Clustering. *Given a set of regular demonstrations D, partition each $d_i \in D$ into a sequence of sub-trajectories defined by transitions $[S^{(i)}_1, \ldots S^{(i)}_k]$. Each transition should correspond to exactly one other transition in at least a fraction of ρ demonstrations.*

A Probabilistic Model For Transitions

In this section, we formalize the definition of transitions and transition states.

Demonstrations as Dynamical Systems

Each d_i is a trajectory $[\mathbf{x}_1, \ldots, \mathbf{x}_T]$. We model each demonstration as a realization of a noisy dynamical system governed by the dynamics ξ and i.i.d Gaussian white noise:

$$\mathbf{x}_{t+1} = \xi(\mathbf{x}_t) + w_t \tag{2}$$

We assume that ξ is locally-linear and can be modeled as switched linear dynamical system. That is, there exists m $d \times d$ matrices $\{A^{(1)}, \ldots, A^{(m)}\}$:

$$\mathbf{x}_{t+1} = A_t \mathbf{x}_t + \mathbf{w}_t : A_t \in \{A^{(1)}, \ldots, A^{(m)}\} \tag{3}$$

A *transition states* is defined as a state at which the dynamics matrix changes at the next time-step $A_t \neq A_{t+1}$.

Transition State Distributions

Over all of the demonstrations D, there is a corresponding set Γ of all transition states. We model the set Γ as samples from an underlying parametrized distribution over the state-space $x \in \mathbf{R}^p$ and time $t \in \mathbb{R}_+$.

$$\Gamma \sim f_\theta(x, t)$$

As the name suggests, Transition State Clustering fits mixture models to f_θ, and this has the interpretation of correlating transition events spatially and temporally. Depending on how we choose to define this joint distribution, we can model different phenomena. We use different hierarchies of Gaussian Mixture Models.

Time-Invariant Transition State Model: The most straight-forward approach is to consider an f that is independent of time. This means that:

$$\forall t, t' \in [0, T] : f_\theta(x, t) = f_\theta(x, t')$$

Then, we can model the distribution as a GMM over just the state-space:

$$f_\theta(x) = GMM(\pi, \{\mu_1, ..., \mu_k\}, \{\Sigma_1, ..., \Sigma_k\})$$

However, this model cannot handle trajectories that cross over the same state multiple times, e.g., a figure 8 trajectory.

Time-Varying Transition State Model: We can extend the above model to consider time-varying distributions. We do this by splitting the distribution into a product of two components, one that is time-invariant and one that depends on time conditioned on the current state. This is a natural consequence of the chain rule where we can decompose $f_\theta(x,t)$ into two independently parametrized densities p, q:

$$f_\theta(x,t) = p_{\theta_p}(x) \cdot q_{\theta_q}(t \mid x)$$

If we model p as a GMM, then for every x will be drawn from one of the $\{1, ..., k\}$ mixture components. We then make a simplifying assumption that this mixture component is a sufficient statistic for q_{θ_q}. Let $z \in \{1, ..., k\}$ be this mixture component, then, we can apply the time-invariant model from above for p, and we can apply a separate GMM for q conditioned on each possible z:

$$q_{\theta_q}(t \mid z) = GMM(\lambda, \{\mu_1, ..., \mu_{l_z}\}, \{\sigma_1, ..., \sigma_{l_z}\})$$

In other words, a GMM models the spatial transition states distribution, and within each Gaussian, the states are further drawn from a GMM over time. The resulting mixture model for f has $\sum_{i=1}^{k} l_i$ components.

Multi-modal Transition State Model: The same logic can be used to model multiple sensing modalities (e.g., kinematics, vision). Let $\binom{x}{v}$ be a state-space constructed of kinematics and visual features $\mathbf{x}$ and $\mathbf{v}$ respectively. Consider the following decomposition p, q, r:

$$f_\theta(\mathbf{x}, t) = p_{\theta_p}(x) \cdot q_{\theta_q}(v \mid x) \cdot r_{\theta_r}(t \mid x, v)$$

As in the time-varying case p, q, r are each modeled as GMMs conditioned on the the mixture component of $\mathbf{x}$ and $\mathbf{x}, \mathbf{v}$ respectively.

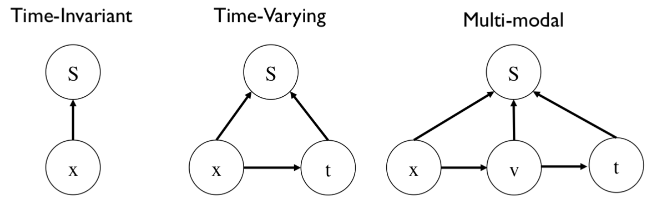

Figure 2. Transition State Clustering models the set of transition states S as a sample from a mixture model that depends on the state x, the time t, and other features v. Different probabilistic models can capture different phenomena

Feedback Model

The proposed model describes systems controlled with linear state feedback controllers to the centroids of the k targets $[\mu_1, ..., \mu_k]$. We can show that the Transition State Clustering model naturally follows from a sequence of stable linear full-state feedback controllers sequentially controlling the system to each μ_i (up-to some tolerance defined by α).

Consider a single target μ_i. Suppose, we model the robot's trajectory in feature space as a linear dynamical system with a fixed dynamics. Let A_r model the robot's linear dynamics and B_r model the robot's control matrix:

$$\mathbf{x}_{t+1} = A_r \mathbf{x}_t + B_r \mathbf{u}_t + \mathbf{w}_t.$$

The robot applies a linear feedback controller with gain G_i, regulating around the target state μ_i. This can be represented as the following system (by setting $u(t) = -G_i \hat{\mathbf{x}}$):

$$\hat{\mathbf{x}}_t = \mathbf{x}_t - \mu_i.$$

$$\hat{\mathbf{x}}_{t+1} = (A_r - B_r C_i)\hat{\mathbf{x}}_t + \mathbf{w}_t.$$

If this system is stable, it will converge to the state $\hat{\mathbf{x}}_t = \mathbf{0}$ which is $\mathbf{x}_t = \mu_i$ as $t \to \infty$. However, since this is a finite time problem, we model a stopping condition, namely, the system is close enough to $\mathbf{0}$. For some z_α (e.g., in 1 dimension 95% quantiles are $Z_{5\%} = 1.96$):

$$\hat{\mathbf{x}}_t^T \Sigma_i^{-1} \hat{\mathbf{x}}_t \leq z_\alpha.$$

If the robot's trajectory was modeled as a sequence $1...k$ of such controllers, we would observe the that the set of transition states Γ would be described as a mixture model around each of the targets $[\mu_1, ..., \mu_k]$. The GMM is a tractable mixture model that approximates this distribution.

Transition State Clustering Algorithm

In this section, we describe the hierarchical clustering process of TSC. TSC is two-phase algorithm that first identifies **transitions states**, defined as consecutive time-steps assigned to different segments, and then, **clusters** spatially and temporally similar transition states across demonstrations with a non-parametric mixture model under the assumption that each demonstration follows the same partial order of transitions up-to noise and loops. The algorithm is summarized in Algorithm 1.

Non-Parametric Mixture Models

Hyper-parameter selection is a known problem in mixture models. Recent results in Bayesian statistics can mitigate some of these problems by defining a soft prior of the number of mixtures. Consider the process of drawing samples from a Gaussian Mixture Models (GMM). We first sample some c from a categorical distribution, one that takes on values from (1...k), with probabilities ϕ, where ϕ is a K dimensional simplex:

$$c \sim cat(k, \phi)$$

Then, conditioned on the event $\{c = i\}$, we sample from a multivariate Gaussian distribution:

$$\mathbf{x}_i \sim N(\mu_i, \Sigma_i)$$

We can see that sampling a GMM is a two-stage process of first sampling from the categorical distribution and then conditioning on that sample.

Algorithm 1: The Transition State Clustering Algorithm

```
1: Input: $D$ demonstrations, $\ell$ a window size, $\rho$ pruning parameter, $\delta$ compaction parameter, and $\alpha$ a Dirichlet Process concentration prior.
2: $\mathbf{n}_t^{(\ell)} = [\mathbf{x}_{t-\ell}, ..., \mathbf{x}_t]^\intercal$.
3: Fit a mixture model to $\mathbf{n}_t$ using DP-GMM assigning each state to its most likely component.
4: Transition states are when $\mathbf{n}_t$ has a different most likely mixture component than $\mathbf{n}_{t+1}$.
5: Fit a mixture model to the set of transition states in the state-space using DP-GMM.
6: Conditioned on each possible state-space mixture component, apply DP-GMM to the set of times.
7: Assign every time step to its most likely mixture component, prune mixture components that do not have at least 1 observation from a fraction $\rho$ demonstrations.
8: Merge together transitions that are within $\delta$ L2 distance after Dynamic Time Warping.
9: Output: A set of transitions, which are regions of the state-space and temporal intervals defined by Gaussian sub-level sets.
```

The key insight of Bayesian non-parametrics is to add another level (or multiple levels) to this model. The Dirichlet Process (DP) defines a distribution over discrete distributions; in other words, a categorical distribution with certain probabilities and setting of k itself is a sample from a DP (Kulis and Jordan 2012). To sample from the Dirichlet Process-GMM model, one must first sample from the DP, then sample from the categorical distribution, and finally sample from the Gaussian:

$$(K, \phi) \sim DP(H, \alpha) \qquad c \sim cat(K, \phi) \qquad \mathbf{x} \sim N(\mu_i, \Sigma_i)$$

The parameters of this model can be solved with variational Expectation Maximization. We denote this entire clustering method in the remainder of this work as DP-GMM. DP-GMM is applied in multiple steps of the TSC algorithm including both transition identification and state clustering.

Transition States Identification

The first step is to identify a set of transition states for each demonstration in D. Suppose there was only one regime, then this would be a linear regression problem:

$$\arg\min_A \|AX_t - X_{t+1}\|$$

where $X_t = [\mathbf{x}_1, \dots, \mathbf{x}_T] \in \mathbb{R}^{p \times T}$ with each column as the state at time t: $\mathbf{x}_t \in \mathbb{R}^p$. Generalizing to multiple regimes, Moldovan et al. (2015) showed that fitting a jointly Gaussian model to $\mathbf{n}_t = \binom{\mathbf{x}_{t+1}}{\mathbf{x}_t}$ is equivalent to Bayesian Linear Regression, and a number of others have applied similar techniques (Khansari-Zadeh and Billard 2011; Kruger et al. 2010). This general logic defines a family of estimators, where we can define $\mathbf{n}_t^{(\ell)}$ as:

$$\mathbf{n}_t^{(\ell)} = [\mathbf{x}_{t-\ell}, ..., \mathbf{x}_t]^\intercal$$

In our experiments, unless otherwise noted, we use $\ell = 1$.

Therefore, to fit a switched linear dynamical system model, we can fit a DP-GMM model to $\mathbf{n}_t$. Each $\mathbf{n}_t$ is assigned to a most likely mixture component (i.e., cluster). To find transition states, we move along a trajectory from $t = 1, ..., t_f$, and find states at which $\mathbf{n}_t$ has a different most likely mixture component than n_{t+1}. These points mark transitions. The result is a set Γ of transition states across all demonstrations.

Linearization with GMMs Using a GMM (and by extension a DP-GMM) to detect switches in local linearity is an approximate algorithm that has been applied in several prior works Moldovan et al. (2015); Calinon et al. (2014); Khansari-Zadeh and Billard (2011).

Consider the following dynamical system:

$$\mathbf{x}_{t+1} = \xi(\mathbf{x}_t) + \mathbf{w}_t$$

where $\mathbf{w}_t$ is unit-variance i.i.d Gaussian noise $N(0, I)$. Let us first focus on linear systems. If ξ is linear, then the problem of learning ξ reduces to linear regression:

$$\arg\min_A \sum_{t=1}^{T-1} \|A\mathbf{x}_t - \mathbf{x}_{t+1}\|.$$

Alternatively, we can think about this linear regression probabilistically. Let us first consider the following proposition:

Proposition 1. *Consider the one-step dynamics of a linear system. Let $\mathbf{x}_t \sim N(\mu, \Sigma)$, then $\binom{\mathbf{x}_t}{\mathbf{x}_{t+1}}$ is a multivariate Gaussian.*

Following from this idea, if we let p define a distribution over $\mathbf{x}_{t+1}$ and $\mathbf{x}_t$:

$$p(\mathbf{x}_{t+1}, \mathbf{x}_t) \sim N(\mu, \Sigma)$$

For multivariate Gaussians the conditional expectation is a linear estimate, and we can see that it is equivalent to the regression above:

$$\arg\min_A \sum_{t=1}^{T-1} \|A\mathbf{x}_t - \mathbf{x}_{t+1}\| = \mathbf{E}[\mathbf{x}_{t+1} \mid \mathbf{x}_t].$$

The GMM model allows us to extend this line of reasoning to consider more complicated ξ. If ξ is non-linear p will almost certainly not be Gaussian. However, GMM models can model complex distributions in terms of Gaussian Mixture Components:

$$p(\mathbf{x}_{t+1}, \mathbf{x}_t) \sim GMM(k)$$

where k denotes the number of mixture components. The interesting part about this mixture distribution is that locally, it models the dynamics as before. Conditioned on particular Gaussian component i the conditional expectation is:

$$\mathbf{E}[\mathbf{x}_{t+1} \mid \mathbf{x}_t, i \in 1...k].$$

As before, conditional expectations of Gaussian random variables are linear, with some additional weighting $\phi(i \mid x_t, x_{t+1})$:

$$\arg\min_{A_i} \sum_{t=1}^{T-1} \phi(i \mid \mathbf{x}_t, \mathbf{x}_{t+1}) \cdot \|A_i\mathbf{x}_t - \mathbf{x}_{t+1}\|.$$

Every tuple $(\mathbf{x}_{t+1}, \mathbf{x}_t)$ probability $\phi(i \mid \mathbf{x}_t, \mathbf{x}_{t+1})$ of belonging to each ith component, and this can be thought of as a likelihood of belonging to a given locally linear model.

As we mentioned earlier, there are multiple techniques for the transition identification problem. We highlight the alternatives in the appendix.

Learning The Transition State Distribution

Now, we learn the parameters for the time-varying transition state distribution.

DP-GMM in State-Space: First, we fit a DP-GMM to the spatial distribution of transition states. There are numerous transition states at different locations in the state-space. If we model the states at transition states as drawn from a DP-GMM model:

$$x_t \sim GMM(\pi, \{\mu_1, ..., \mu_k\}, \{\Sigma_1, ..., \Sigma_k\})$$

Then, we can apply the DP-GMM again to group the state vectors at the transition states. After fitting the GMM, each $x \in \mathscr{X}$ will have a $\hat{z} \in \{0, 1, ..., k\}$ associated with it, which is the most likely mixture component from which it is generated.

DP-GMM in Time: Using this $\hat{z}$, we apply the second level of DP-GMM fitting over the time axis. Without temporal localization, the transitions may be ambiguous. For example, in circle cutting, the robot may pass over a point twice in the same task. Conditioned on $\hat{z} = i$, we model the times which change points occur as drawn from a GMM $t \sim GMM(\pi, \{\mu_1, ..., \mu_{l_i}\}, \{\Sigma_1, ..., \Sigma_{l_i}\})$, and then we can apply DP-GMM to the set of times. Intuitively, this can be viewed as a hierarchical clustering process that groups together events that happen at similar times during the demonstrations. The result is a distribution that models spatially and temporally similar transitions.

Interpreting the Distribution: The above model defines a mixture distribution with $m = \sum_{i=0}^{k} l_i$ components, where k is the number of state-space components, and conditioned on each state-space component i there are l_i time-axis components. If there are m total mixture components for the distribution $\{C_1, ..., C_m\}$. Each mixture component defines a Gaussian over the state-space and a distribution that is conditionally Gaussian over time. The quantiles of each component distribution will define an ordered sequence of regions $[\rho_1, ..., \rho_k]$ over the state-space (i.e., its sublevel set of the state-space Gaussian bounded by z_α and ordered by mean the time Gaussian).

Outlier Rejection and Loop Compaction

Next, we describe our approach to make the model resilient to noise in the form of spurious actions and loops.

Transition State Pruning: We consider the problem of outlier transitions, ones that appear only in a few demonstrations. Each transition state will have a most likely mixture component $\hat{z} \in \{1, ..., m\}$. Mixture components whose constituent transition states come from fewer than a fraction ρ demonstrations are *pruned*. ρ should be set based on the expected rarity of outliers. For example, if ρ is 100% then the only mixture components that are found are those with at least one transition state from every demonstration (i.e., the regularity assumption). If ρ is less than 100%, then it means that every mixture component must cover some subset of the demonstrations. In our experiments, we set the parameter ρ to 80% and show the results with and without this step.

Transition State Compaction: Once we have applied pruning, the next step is to remove transition states that correspond to looping actions, which are prevalent in surgical demonstrations. We model this behavior as consecutive transition states that have the same state-space GMM mixture component. We apply this step after pruning to take advantage of the removal of outlier mixtures during the looping process.

The key question is how to differentiate between repetitions that are part of the demonstration and ones that correspond to looping actions–the sequence might contain repetitions not due to looping. As a heuristic, we threshold the L2 distance between consecutive segments with repeated transitions. If the L2 distance is low, we know that the consecutive segments are happening in a similar location as well. In our datasets, this is a good indication of looping behavior.

For each demonstration, we define a segment $\mathbf{s}^{(j)}[t]$ of states between each transition states. The challenge is that $\mathbf{s}^{(j)}[t]$ and $\mathbf{s}^{(j+1)}[t]$ may have a different number of observations and may be at different time scales. To address this challenge, we apply Dynamic Time Warping (DTW). Since segments are locally similar up-to small time variations, DTW can find a most-likely time alignment of the two segments.

Let $\mathbf{s}^{(j+1)}[t^*]$ be a time aligned (w.r.t to $\mathbf{s}^{(j)}$) version of $\mathbf{s}^{(j+1)}$. Then, after alignment, we define the L_2 metric between the two segments:

$$d(j, j+1) = \frac{1}{T} \sum_{t=0}^{T} (\mathbf{s}^{(j)}[i] - \mathbf{s}^{(j+1)}[i^*])^2$$

When $d \leq \delta$, we compact two consecutive segments. δ is chosen empirically and a larger δ leads to a sparser distribution of transition states, and smaller δ leads to more transition states. For our needle passing and suturing experiments, we set δ to correspond to the distance between two suture/needle insertion points–thus, differentiating between repetitions at the same point vs. at others. However, since we are removing points from a time-series, this requires us to adjust the time scale. Thus, from every following observation, we shift the time stamp back by the length of the compacted segments.

Results

We present the results evaluating TSC on a synthetic dataset and three real data sets of kinematic and visual recordings of surgical training tasks on the dVRK.

Synthetic Example

One of the challenges in evaluating segmentation techniques on real datasets is that the ground truth is often not known. Comparing different segmentation models can be challenging due to differing segmentation criteria. We

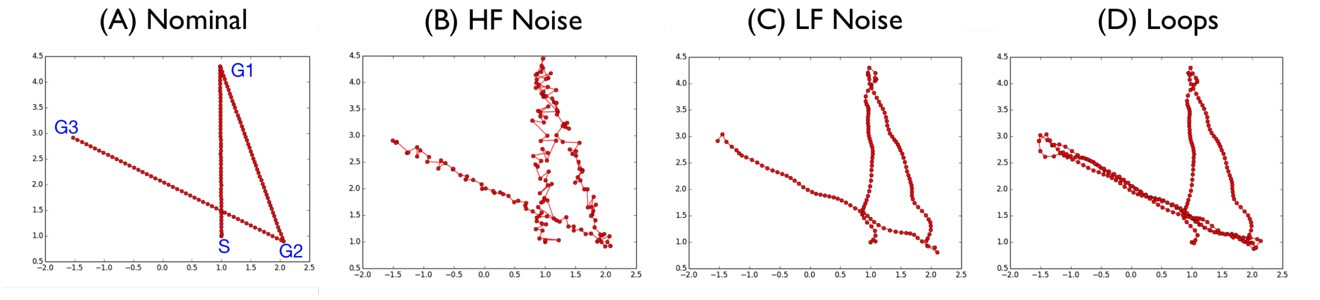

Figure 3. One of 20 instances with random goal points $G1$, $G2$, $G3$. (a) Observations from a simulated demonstration with three regimes, (b) Observations corrupted with Gaussian white sensor noise, (c) Observations corrupted with low frequency process noise, and (d) Observations corrupted with an inserted loop. See Figure 8 for evaluation on loops.

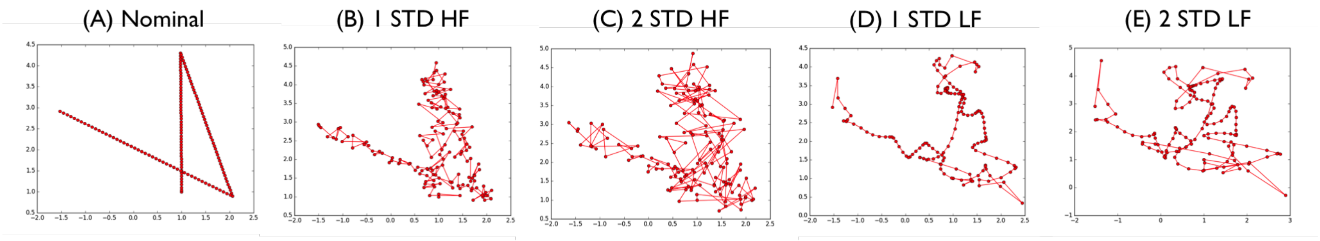

Figure 4. (a) Nominal trajectory, (b) 1 std. of high frequency observation noise, (c) 2 std. of high frequency observation noise, (d) 1 std. of low frequency process noise, and (e) 2 std. of low frequency process noise.

developed a synthetic dataset generator for segmentation and compared several algorithms on the generated dataset. Note, we do not intend this to be a comprehensive evaluation of the accuracy of the different techniques, but more a characterization of the approaches on a locally linear example to study the key tradeoffs. The primary purpose of our experiment is to evaluate the following hypothesis: TSC more accurately recovers the ground truth when the data is corrupted with observation noise and model noise.

Overview We model the motion of a holonomic point robot with two-dimensional position state (x,y) between k goal points $\{g_1,...,g_k\}$. We apply position control to guide the robot to the targets and without disturbance this motion is linear (Figure 3a). We add various types of disturbances (and in varying amounts) including Gaussian observation noise, low-frequency process noise, and repetitive loops (Figure 3b-d). We report noise values in terms of standard deviations. Figure 4 illustrates the relative magnitudes. A demonstration d_i is a sample from the following system.

Task: Every segmentation algorithm will be evaluated in its ability to identify the $k-1$ segments (i.e., the paths between the goal points). Furthermore, we evaluate algorithms on random instances of this task. In the beginning, we select 3 random goal points. From a fixed initial position, we control the point robot to the points with position control. Without any disturbance this follows a linear motion. For a given noise setting, we sample demonstrations from this system, and apply/evaluate each algorithm. We present results aggregated over 20 such random instances. This is important since many of the segmentation algorithms proposed in literature have some crucial hyper-parameters, and we present results with a *single* choice of parameters averaged over multiple tasks. This way, the hyper-parameter tuning cannot overfit to any given instance of the problem and has to be valid for the entire class of tasks. We believe that this is important since tuning these hyper-parameters in practice (i.e., not in simulation) is challenging since there is no ground truth. The experimental code is available at: `http://berkeleyautomation.github.io/tsc/`.

5 Algorithms: We compare TSC against alternatives where the authors explicitly find (or approximately find) locally linear segments. It is important to reiterate that different segmentation techniques optimize different objectives, and this benchmark is meant to characterize the performance on a common task.

1. *(GMM)* We compare to a version of the approach proposed by Lee et al. (2015). In this technique, we apply a GMM to a vector of states augmented with the current time. The authors cite Ghahramani and Jordan (1993) to argue that this is a form of local linear regression. In Lee et al. (2015), the authors use Bayesian Information Criterion (BIC) to optimize the hyper-parameter of the number of mixture components. In our experiments, we set the parameter to the optimal choice of 3 without automatic tuning.

2. *(GMM+HMM)* A natural extension to this model is to enforce a transition structure on the regimes with a latent Markov Chain (Asfour et al. 2008; Calinon and Billard 2004; Kruger et al. 2010; Vakanski et al. 2012). We use the same state vector as above, without time augmentation as this is handled by the HMM. We fit the model using the forward-backward (or Baum-Welsch) algorithm.

3. *Coresets* We evaluate against a standard coreset model (Sung et al. 2012; Volkov et al. 2015), and

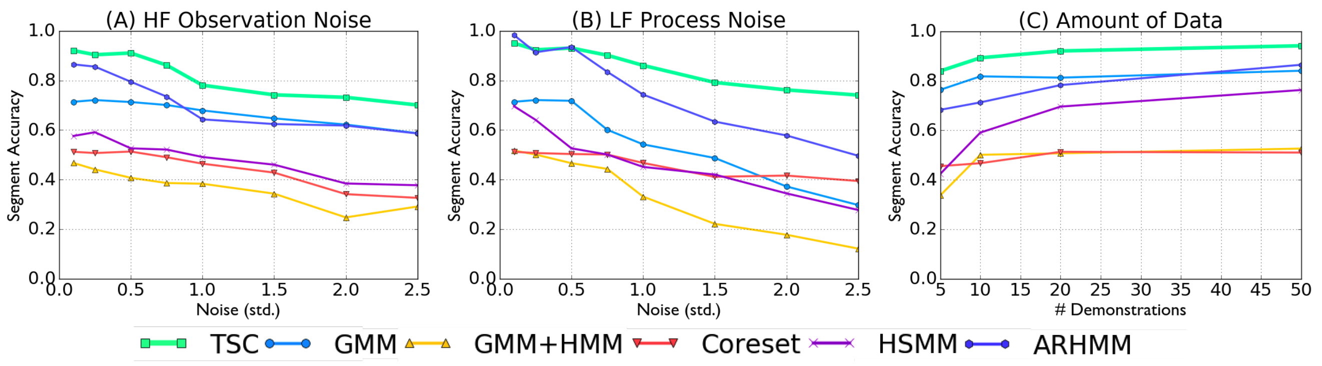

Figure 5. Each data point represents 20 random instances of a 3-segment problem with varying levels of high-frequency noise, low-frequency noise, and demonstrations. We measure the segmentation accuracy for the compared approaches. (A) TSC finds more a accurate segmentation than all of the alternatives even under significant high-frequency observation noise, (B) TSC is more robust low-frequency process noise than the alternatives, (C) the Bayesian techniques solved with MCMC (ARHMM, HSMM) are more sensitive to the number of demonstrations provided than the others.

the particular variant is implemented with weighted k-means. We applied this to the same augmented state-vector as in the previously mentioned GMM.

4. *HSMM* We evaluated a Gaussian Hidden Semi-Markov Model as used in Niekum et al. (2012). We directly applied this model to the demonstrations with no augmentation or normalization of features. This was implemented with the package pyhsmm. We directly applied this model to the demonstrations with no augmentation as in the GMM approaches.

5. *AR-HMM* We evaluated a Bayesian Autoregressive HMM model as used in Niekum et al. (2012). This was implemented with the packages pybasicbayes and pyhsmm-ar.

Evaluation Metric: There is considerable debate on metrics to evaluate the accuracy of unsupervised segmentation and activity recognition techniques, e.g. frame accuracy (Wu et al. 2015), hamming distance (Fox et al. 2009). Typically, these metrics have two steps: (1) segments to ground truth correspondence, and (2) then measuring the similarity between corresponded segments. We have made this feature extensible and evaluated some different accuracy metrics (Jaccard Similarity, Frame Accuracy, Segment Accuracy, Intersection over Union). We found that the following procedure led to the most insightful results–differentiating the different techniques.

In the first phase, we match segments in our predicted sequence to those in the ground truth. We do this with a procedure identical to the one proposed in Wu et al. (2015). We define a bi-partite graph of predicted segments to ground truth segments, and add weighted edges where weights represent the overlap between a predicted segment and a ground truth segment (i.e, the recall over time-steps). Each predicted segment is matched to its highest weighted ground truth segment. Each predicted segment is assigned to exactly one ground-truth segment, while a ground-truth segments may have none, one, or more corresponding predictions.

After establishing the correspondence between predictions and ground truth, we consider a true positive (a ground-truth segment is correctly identified) if the overlap (intersection-over-union) between the ground-truth segment and its

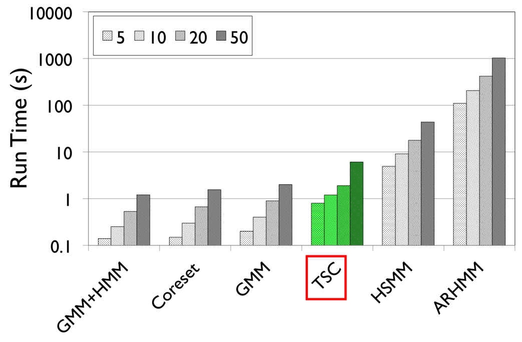

Figure 6. TSC is about 6x slower than using Coresets or the direct GMM approach, but it is over 100x faster than the MCMC for the ARHMM model.

corresponding predicted segments is more than a default threshold 60%. Then, we compute Segment Accuracy as the ratio of the ground-truth segments that are correctly detected. In Wu et al. (2015), the authors use a 40% threshold but apply the metric to real data. Since this is a synthetic example, we increase this threshold to 60%, which we empirically found accounted for boundary effects especially in the Bayesian approaches (i.e., repeated transitions around segment endpoints).

Accuracy v.s. Noise In our first experiment, we measured the segment accuracy for each of the algorithms. We also varied the amount of process and observation noise in the system. As Figure 4 illustrates, this is a very significant amount of noise in the data and successful techniques must exploit the structure in multiple demonstrations. Figure 5a illustrates the performance of each of the techniques as a function of high-frequency observation noise. Results suggest that TSC is more robust to noise than the alternatives (nearly 20% more accurate for 2.5 std of noise). The Bayesian ARHMM approach is nearly identical to TSC when the noise is low but quickly loses accuracy as more noise is added. We attribute this robustness to the TSC's pruning step which ensures that only transition state clusters with sufficient coverage over all demonstrations are kept. These results are even more pronounced for low-frequency process noise (Figure 5b). TSC is 49% more accurate than all competitors for 2.5 std of noise added. We find that the Bayesian approaches are particularly susceptible to such

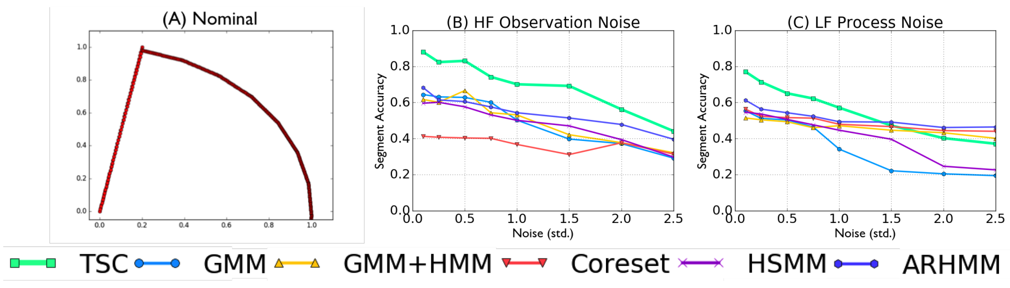

Figure 7. (A) illustrates a nominal trajectory of two linear dynamical motions. (B) TSC more accurately recovers the two segment ground truth than the alternatives under observation noise, (C) all of the techniques suffer in accuracy under process noise.

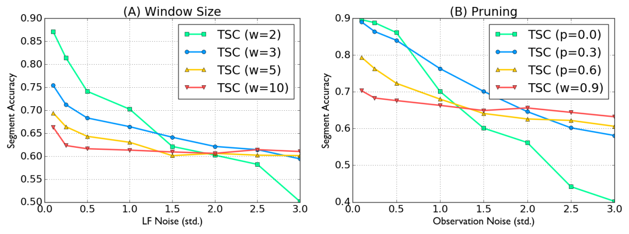

Figure 8. (A) shows the performance curves of different choices of windows as a function of the process noise. Larger windows can reject higher amounts of process noise but are less efficient at low noise levels. (B) the performance curves of different choices of the pruning threshold. Larger pruning thresholds are more robust to high amounts of observation noise but less accurate in the low noise setting. We selected $(w = 3, \rho = 0.3)$ in our synthetic experiments.

noise. Furthermore, Figure 5c shows requires no more data than the alternatives to achieve such robustness.

Another point to note is that TSC is solved much more efficiently than ARHMM or HSMM which require expensive MCMC samples. While parameter inference on these models can be solved more efficiently (but approximately) with Mean-Field Stochastic Variational Inference, we found that the results were not as accurate. TSC is about 6x slower than using Coresets or the direct GMM approach, but it is over 100x faster than the MCMC for the ARHMM model. Figure 6 compares the runtime of each of the algorithms as a function of the number of demonstrations.

TSC Hyper-Parameters Next, we explored the dependence of the performance on the hyper-parameters for TSC. We focus on the window size and the pruning parameter. Figure 8a shows how varying the window size affects the performance curves. Larger window sizes can reject more low-frequency process noise. However, larger windows are also less efficient when the noise is low. Similarly, Figure 8b shows how increasing the pruning parameter affects the robustness to high-frequency observation noise. However, a larger pruning parameter is less efficient at low noise levels. Based on these curves, we selected $(w = 3, \rho = 0.3)$ in our synthetic experiments.

Loops Finally, we evaluated 4 algorithms on how well they can detect and adjust for loops. TSC compacts adjacent motions that are overly similar, while HMM-based approaches correspond similar looking motions. An HMM grammar over segments is clearly more expressive than TSC's, and we explore whether it is necessary to learn a full transition structure to compensate for loops. We compare the accuracy of the different segmentation techniques in detecting that a loop is present (Figure 9). Figure 9a shows that TSC is competitive with the HMM approaches as we vary the observation noise; however, the results suggest that ARHMM provides the most accurate loop detection. On the hand, Figure 9b suggests that process noise has a very different effect. TSC is actually more accurate than the HMM

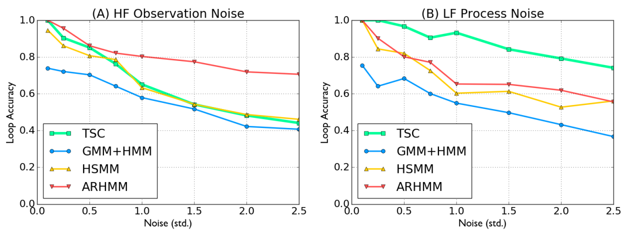

Figure 9. (A) illustrates the accuracy of TSC's compaction step as a function of observation noise. TSC is competitive with the HMM-based approaches without having to model the full transition matrix. (B) TSC is actually more robust to low-frequency process noise in the loops than the HMM-based approaches.

approaches when the process noise is high–even without learning a transition structure. This is an interesting property that we find is very useful in our experiments on real surgical data.

Dynamical Trajectories It is important to differentiate linear dynamical motions from linear trajectories. TSC models trajectories as linear dynamical systems and this allows for circular and spiral trajectories. Next, we evaluate TSC on an example with two linear dynamical systems. One system represents a straight line trajectory which transitions into a circular motion. Figure 7 illustrates the results. We find that this problem is substantially harder than the previous problem and all of the algorithms show reduced accuracy. TSC is still the most accurate.

Surgical Data Experiments

We describe the three tasks used in our evaluation and the corresponding manual segmentation (Figure 10). This will serve as ground truth when qualitatively evaluating our segmentation on real data. This set of experiments primarily evaluates the utility of segments learned by TSC. Our hypothesis is that even though TSC is unsupervised, it identifies segments that often align with manual annotations. In all of our experiments, the pruning parameter ρ is set to 80% and the compaction heuristic δ is to 1cm.

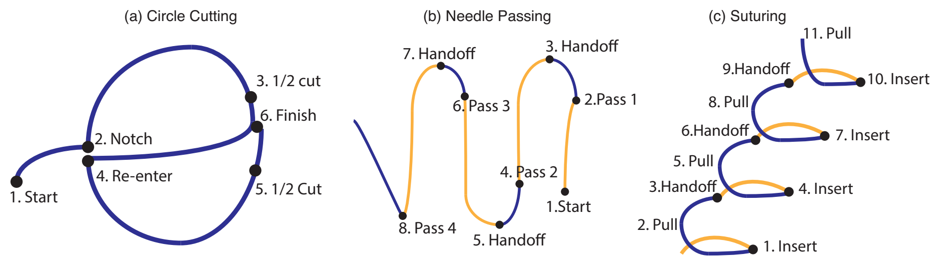

Figure 10. Hand annotations of the three tasks: (a) circle cutting, (b) needle passing, and (c) suturing. Right arm actions are listed in dark blue and left arm actions are listed in yellow.

Circle Cutting: A 5 cm diameter circle drawn on a piece of gauze. The first step is to cut a notch into the circle. The second step is to cut clockwise half-way around the circle. Next, the robot transitions to the other side cutting counter clockwise. Finally, the robot finishes the cut at the meeting point of the two cuts. As the left arm's only action is to maintain the gauze in tension, we exclude it from the analysis. In Figure 10a, we mark 6 manually identified transitions points for this task from Murali et al. (2015): (1) start, (2) notch, (3) finish 1st cut, (4) cross-over, (5) finish 2nd cut, and (6) connect the two cuts. For the circle cutting task, we collected 10 demonstrations by non-experts familiar with operating the da Vinci Research Kit (dVRK).

We also perform experiments using the JIGSAWS dataset Gao et al. (2014) consisting of surgical activity for human motion modeling. The dataset was captured using the da Vinci Surgical System from eight surgeons with different levels of skill performing five repetitions each of Needle Passing and Suturing.

Needle Passing: We applied TSC to 28 demonstrations of the needle passing task. The robot passes a needle through a hoop using its right arm, then its left arm to pull the needle through the hoop. Then, the robot hands the needle off from the left arm to the right arm. This is repeated four times as illustrated with a manual segmentation in Figure 10b.

Suturing: Next, we explored 39 examples of a 4 throw suturing task (Figure 10c). Using the right arm, the first step is to penetrate one of the points on right side. The next step is to force the needle through the phantom to the other side. Using the left arm, the robot pulls the needle out of the phantom, and then hands it off to the right arm for the next point.

Visual Features

TSC is compatible with visual features in addition to kinematic states. Our goal with these features was to illustrate that TSC applies to general state-spaces as well as spatial ones, and not to address the general perception problem. These features were constructed via manual annotation, where the Grasp and Needle Penetration were identified by reviewing the videos and marking the frames at which they occurred (Section).

We evaluate TSC in this featurized state space that incorporates states derived from vision. We illustrate the transition states in Figure 13 with and without visual features on the circle cutting task. At each point where the model transitions, we mark the end-effector (x, y, z) location (ignoring the orientation). In particular, we show a region (red box) to highlight the benefits of these features. During the cross-over phase of the task, the robot has to re-enter the notch point and adjust to cut the other half of the circle. When only using the end-effector kinematic pose, the locations where this transition happens is unreliable as operators may approach the entry from slightly different angles. On the other hand, the use of a gripper contact binary feature clusters the transition states around the point at which the gripper is in position and ready to begin cutting again. In the subsequent experiments, we use the same two visual features.

Pruning and Compaction In Figure 14, we highlight the benefit of pruning and compaction using the Suturing task as exemplar. First, we show the transition states without applying the compaction step to remove looping transition states (Figure 14a). We find that there are many more transition states at the "insert" step of the task. Compaction removes the segments that correspond to a loop of the insertions. Next, we show the all of the clusters found by DP-GMM. The centroids of these clusters are marked in Figure 14b. Many of these clusters are small containing only a few transition states. This is why we created the heuristic to prune clusters that do not have transition states from at least 80% of the demonstrations. In all, 11 clusters are pruned by this rule.

Results with Surgical Data

Circle Cutting: Figure 15a shows the transition states obtained from our algorithm. And Figure 15b shows the TSC clusters learned (numbered by time interval midpoint). The algorithm found 8 clusters, one of which was pruned using our $\rho = 80\%$ threshold rule.

The remaining 7 clusters correspond well to the manually identified transition points. It is worth noting that there is one extra cluster (marked $2'$), that does not correspond to a transition in the manual segmentation. At $2'$, the operator finishes a notch and begins to cut. While at a logical level notching and cutting are both penetration actions, they correspond to two different linear transition regimes due to the positioning of the end-effector. Thus, TSC separates them into different clusters even though a human annotators did not. This illustrates why supervised segmentation is challenging. Human annotators segment trajectories on boundaries that are hard to characterize mathematically, e.g., is frame 34 or frame 37 the segment boundary. Supervisors may miss crucial motions that are useful for automation or learning.

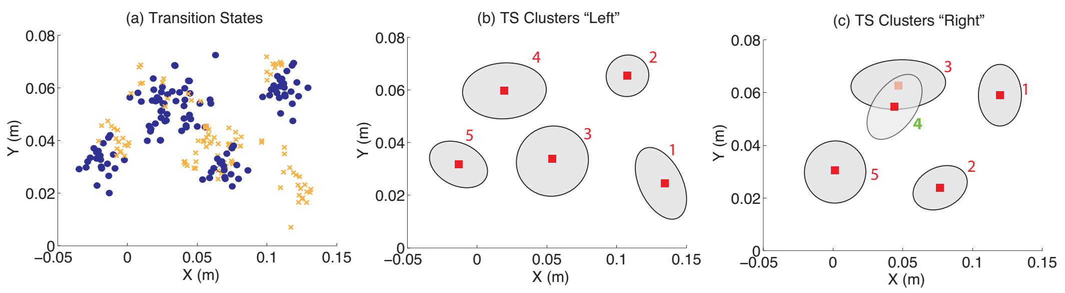

Figure 11. (a) The transition states for the task are marked in orange (left arm) and blue (right arm). (b-c) The TSC clusters, which are clusters of the transition states, are illustrated with their 75% confidence ellipsoid for both arms

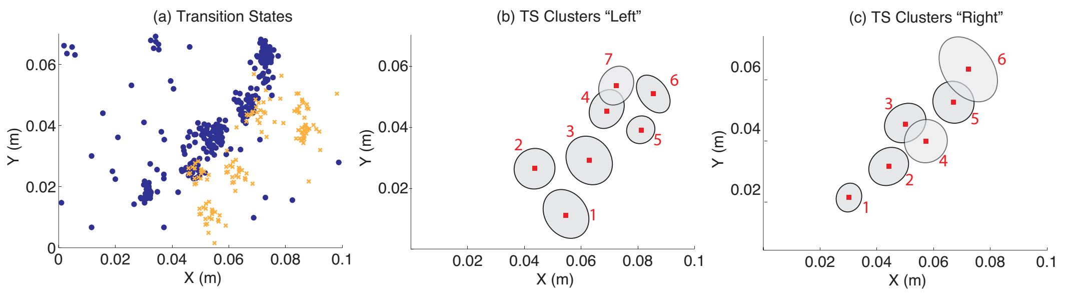

Figure 12. (a) The transition states for the task are marked in orange (left arm) and blue (right arm). (b-c) The clusters, which are clusters of the transition states, are illustrated with their 75% confidence ellipsoid for both arms

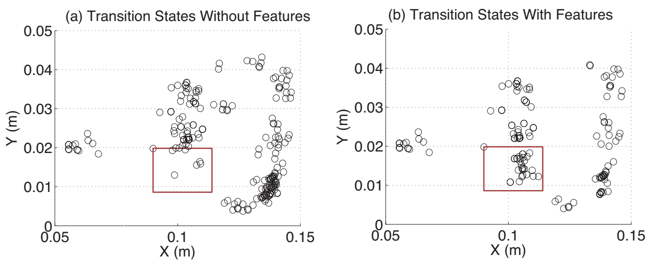

Figure 13. (a) We show the transition states without visual features, (b) and with visual features. Marked in the red box is a set of transitions that cannot always be detected from kinematics alone.

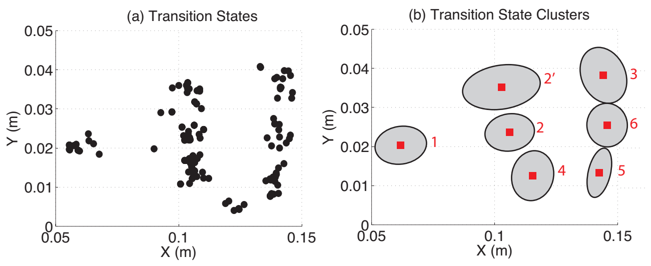

Figure 15. (a) The transition states for the circle cutting task are marked in black. (b) The TSC clusters, which are clusters of the transition states, are illustrated with their 75% confidence ellipsoid.

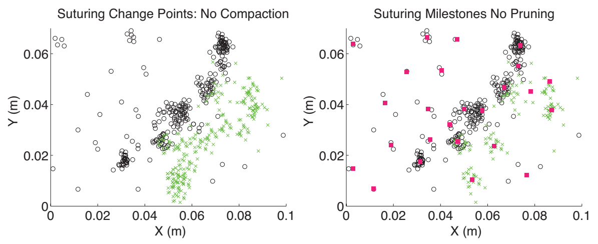

Figure 14. We first show the transition states without compaction (in black and green), and then show the clusters without pruning (in red). Compaction sparsifies the transition states and pruning significantly reduces the number of clusters.

Needle Passing: In Figure 11a, we plot the transition states in (x, y, z) end-effector space for both arms. We find that these transition states correspond well to the logical segments of the task (Figure 10b). These demonstrations are noisier than the circle cutting demonstrations and there are more outliers. The subsequent clustering finds 9 (2 pruned). Next, Figures 11b-c illustrate the TSC clusters. We find that again TSC learns a small parametrization for the task structure with the clusters corresponding well to the manual segments. However, in this case, the noise does lead to a spurious cluster (4 marked in green). One possible explanation is that the demonstrations contain many adjustments to avoid colliding with the needle hoop and the other arm while passing the needle through leading to numerous transition states in that location.

Suturing: In Figure 12, we show the transition states and clusters for the suturing task. As before, we mark the left arm in orange and the right arm in blue. This task was far more challenging than the previous tasks as the demonstrations were inconsistent. These inconsistencies were in the way the suture is pulled after insertion (some pull to the left, some to the right, etc.), leading to transition states all over the state space. Furthermore, there were numerous demonstrations with looping behaviors for the left arm. In fact, the DP-GMM method gives us 23 clusters, 11 of which represent less than 80% of the demonstrations and thus are pruned (we illustrate the effect of the pruning in the next section). In the early stages of the task, the clusters clearly correspond to the manually segmented transitions. As the task progresses, we see that some of the later clusters do not.

Comparison to Surgemes

Surgical demonstrations have an established set of primitives called surgemes, and we evaluate if segments

Table 1. This table compares transitions learned by TSC and transitions identified by manual annotators in the JIGSAWS dataset. We found that the transitions mostly aligned. 83% and 73% of transition clusters for needle passing and suturing respectively contained exactly one surgeme transition. These results suggest that TSC aligns with surgemes without any explicit supervision.

	No. of Surgeme Segments	No. of Segments + C/P	No. of TSC	TSC-Surgeme	Surgeme-TSC
Needle Passing	19.3 ± 3.2	14.4 ± 2.57	11	83%	74%
Suturing	20.3 ± 3.5	15.9 ± 3.11	13	73%	66%

discovered by our approach correspond to surgemes. In Table 1, we compare the number of TSC segments for needle passing and suturing to the number of annotated surgeme segments. A key difference between our segmentation and number of annotated surgemes is our compaction and pruning steps. To account for this, we first select a set of surgemes that are expressed in most demonstrations (i.e., simulating pruning), and we also apply a compaction step to the surgeme segments. When surgemes appear consecutively, we only keep the one instance of each. We explore two metrics: **TSC-Surgeme** the fraction of TSC clusters with only one surgeme switch (averaged over all demonstrations), and **Surgeme-TSC** the fraction of surgeme switches that fall inside exactly one TSC cluster. We found that the transitions learned by TSC often aligned with the surgemes. 83% and 73% of transition clusters for needle passing and suturing respectively contained exactly one surgeme transition (**TSC-Surgeme** metric). These results suggest that TSC aligns with surgemes without any explicit supervision.

Future Work

These results suggest several avenues for future work. First, we will explore using Convolutional Neural Networks to automatically extract visual features for segmentation. This will alleviate a key challenge in applying TSC to new datasets. We will also explore how other results in Deep Learning such as Autoencoders and Recurrent Networks can be used to segment data without linearity assumptions. We are also interested in exploring the connections between TSC and other time-series models such as Derivative Dynamic Time Warping which aligns the derivative of two signals. Segmentation is the first step in a broader robot learning pipeline, and we are actively exploring using segmentation to construct rewards for Reinforcement Learning.

Conclusion

We presented Transition State Clustering (TSC), which leverages the consistent structure of repeated demonstrations robustly learn segmentation criteria. To learn these clusters, TSC uses a hierarchical Dirichlet Process Gaussian Mixture Model (DP-GMM) with a series of merging and pruning steps. Our results on a synthetic example suggest that this approach is more robust than 5 other segmentation algorithms. We further applied our algorithm to three surgical datasets and found that the transition state clusters correspond well to manual annotations and transitions with respect to motions from a pre-defined surgical motion dictionary (surgemes).

Acknowledgements

This research was supported in part by a seed grant from the UC Berkeley Center for Information Technology in the Interest of Society (CITRIS), the UC Berkeley AMPLab, and by the U.S. National Science Foundation under Award IIS-1227536: Multilateral Manipulation by Human-Robot Collaborative Systems. This work has been supported in part by funding from Google and and Cisco. We also thank Florian Pokorny, Jeff Mahler, and Michael Laskey for feedback and discussions.

References

Tamim Asfour, Pedram Azad, Florian Gyarfas, and Rüdiger Dillmann. Imitation learning of dual-arm manipulation tasks in humanoid robots. *I. J. Humanoid Robotics*, 5(2):183–202, 2008.

Sylvain Calinon and Aude Billard. Stochastic gesture production and recognition model for a humanoid robot. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, September 28 - October 2, 2004*, pages 2769–2774, 2004.

Sylvain Calinon, Florent D'halluin, Eric L Sauser, Darwin G Caldwell, and Aude G Billard. Learning and reproduction of gestures by imitation. *Robotics & Automation Magazine, IEEE*, 17(2):44–54, 2010.

Sylvain Calinon, Danilo Bruno, and Darwin G Caldwell. A task-parameterized probabilistic model with minimal intervention control. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3339–3344, 2014.

Emily Fox, Michael I Jordan, Erik B Sudderth, and Alan S Willsky. Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems*, pages 549–557, 2009.

Yixin Gao, S Swaroop Vedula, Carol E Reiley, Narges Ahmidi, Balakrishnan Varadarajan, Henry C Lin, Lingling Tao, Luca Zappella, Benjamın Béjar, David D Yuh, Chi Chen, Rene Vidal, Sanjeev Khudanpur, and Greg D. Hager. The jhu-isi gesture and skill assessment dataset (jigsaws): A surgical activity working set for human motion modeling. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2014.

Zoubin Ghahramani and Michael I. Jordan. Supervised learning from incomplete data via an EM approach. In *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pages 120–127, 1993.

Daniel H Grollman and Odest Chadwicke Jenkins. Incremental learning of subtasks from unsegmented demonstration. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 261–266. IEEE, 2010.

Ben Kehoe, Gregory Kahn, Jeffrey Mahler, Jonathan Kim, Alex X. Lee, Anna Lee, Keisuke Nakagawa, Sachin Patil, W. Douglas Boyd, Pieter Abbeel, and Kenneth Y. Goldberg. Autonomous multilateral debridement with the raven surgical robot. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 1432–1439, 2014.

Mohammad Khansari-Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *Robotics, IEEE Transactions on*, 27(5):943–957, 2011.

Sanjay Krishnan, Animesh Garg, Sachin Patil, Colin Lea, Gregory Hager, Pieter Abbeel, and Ken Goldberg. Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. In *International Symposium of Robotics Research. Springer STAR*, 2015.

Volker Kruger, Dennis Herzog, Sanmohan Baby, Ales Ude, and Danica Kragic. Learning actions from observations. *Robotics & Automation Magazine, IEEE*, 17(2):30–43, 2010.

Volker Krüger, Vadim Tikhanoff, Lorenzo Natale, and Giulio Sandini. Imitation learning of non-linear point-to-point robot motions using dirichlet processes. In *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pages 2029–2034, 2012.

Volker Krüger, Vadim Tikhanoff, Lorenzo Natale, and Giulio Sandini. Imitation learning of non-linear point-to-point robot motions using dirichlet processes. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2029–2034. IEEE, 2012.

Dana Kulić and Yoshihiko Nakamura. Scaffolding on-line segmentation of full body human motion patterns. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2860–2866. IEEE, 2008.

Brian Kulis and Michael I. Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

Colin Lea, Gregory D. Hager, and Renè Vidal. An improved model for segmentation and recognition of fine-grained activities with application to surgical training tasks. In *WACV*, 2015.

Sang Hyoung Lee, Il Hong Suh, Sylvain Calinon, and Rolf Johansson. Autonomous framework for segmenting robot trajectories of manipulation task. *Autonomous Robots*, 38(2): 107–141, 2015.

Henry C. Lin, Izhak Shafran, Todd E. Murphy, Allison M. Okamura, David D. Yuh, and Gregory D. Hager. Automatic detection and segmentation of robot-assisted surgical motions. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2005, 8th International Conference, Palm Springs, CA, USA, October 26-29, 2005, Proceedings, Part I*, pages 802–810, 2005.

Jeffrey Mahler, Sanjay Krishnan, Michael Laskey, Siddarth Sen, Adithyavairavan Murali, Ben Kehoe, Sachin Patil, Jiannan Wang, Mike Franklin, Pieter Abbeel, and Kenneth Y. Goldberg. Learning accurate kinematic control of cable-driven surgical robots using data cleaning and gaussian process regression. In *2014 IEEE International Conference on Automation Science and Engineering, CASE 2014, New Taipei, Taiwan, August 18-22, 2014*, pages 532–539, 2014.

San Mohan, Volker Krüger, Danica Kragic, and Hedvig Kjellström. Primitive-based action representation and recognition. *Advanced Robotics*, 25(6-7):871–891, 2011.

Teodor Mihai Moldovan, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. Optimism-driven exploration for nonlinear systems. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 3239–3246, 2015.

Adithyavairavan Murali, Siddarth Sen, Ben Kehoe, Animesh Garg, Seth McFarland, Sachin Patil, W. Douglas Boyd, Susan Lim, Pieter Abbeel, and Kenneth Y. Goldberg. Learning by observation for surgical subtasks: Multilateral cutting of 3d viscoelastic and 2d orthotropic tissue phantoms. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 1202–1209, 2015.

Scott Niekum, Sarah Osentoski, George Konidaris, and Andrew G. Barto. Learning and generalization of complex tasks from unstructured demonstrations. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 5239–5246, 2012.

Nicolas Padoy, Diana Mateus, Daniel Weinland, M-O Berger, and Nassir Navab. Workflow monitoring based on 3d motion features. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 585–592. IEEE, 2009.

Gwénolé Quellec, Mathieu Lamard, Béatrice Cochener, and Guy Cazuguel. Real-time segmentation and recognition of surgical tasks in cataract surgery videos. *IEEE Trans. Med. Imaging*, 33 (12):2352–2360, 2014.

Carol E Reiley, Erion Plaku, and Gregory D Hager. Motion generation of robotic surgical tasks: Learning from expert demonstrations. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 967–970. IEEE, 2010.

Guy Rosman, Mikhail Volkov, Dan Feldman, John W Fisher III, and Daniela Rus. Coresets for k-segmentation of streaming data. In *Advances in Neural Information Processing Systems*, pages 559–567, 2014.

Ardavan Saeedi, Matthew Hoffman, Matthew Johnson, and Ryan Adams. The segmented ihmm: A simple, efficient hierarchical infinite hmm. *arXiv preprint arXiv:1602.06349*, 2016.

Cynthia Sung, Dan Feldman, and Daniela Rus. Trajectory clustering for motion prediction. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1547–1552. IEEE, 2012.

Hao Tang, Mark Hasegawa-Johnson, and Thomas S Huang. Toward robust learning of the gaussian mixture state emission densities for hidden markov models. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 5242–5245. IEEE, 2010.

Ajay Kumar Tanwani and Sylvain Calinon. Learning robot manipulation tasks with task-parameterized semi-tied hidden semi-markov model. 2015.

Lingling Tao, Luca Zappella, Gregory D Hager, and René Vidal. Surgical gesture segmentation and recognition. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*, pages 339–346. Springer, 2013.

Aleksandar Vakanski, Iraj Mantegh, Andrew Irish, and Farrokh Janabi-Sharifi. Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 42(4):1039–1052, 2012.

Balakrishnan Varadarajan, Carol Reiley, Henry Lin, Sanjeev Khudanpur, and Gregory Hager. Data-derived models for segmentation with application to surgical assessment and training. In *Medical Image Computing and Computer-Assisted*

Intervention–MICCAI 2009, pages 426–434. Springer, 2009.

Mikhail Volkov, Guy Rosman, Dan Feldman, John W Fisher, and Daniela Rus. Coresets for visual summarization with applications to loop closure. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 3638–3645. IEEE, 2015.

Alan S Willsky, Erik B Sudderth, Michael I Jordan, and Emily B Fox. Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems*, pages 549–557, 2009.

Chenxia Wu, Jiemi Zhang, Silvio Savarese, and Ashutosh Saxena. Watch-n-patch: Unsupervised understanding of actions and relations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4362–4370, 2015.

Luca Zappella, Benjamín Béjar Haro, Gregory D. Hager, and René Vidal. Surgical gesture classification from video and kinematic data. *Medical Image Analysis*, 17(7):732–745, 2013.