Motion Learning in Variable Environments Using Probabilistic Flow Tubes

Shuonan Dong and Brian Williams

***Abstract*— Commanding an autonomous system through complex motions at a low level can be tedious or impractical for systems with many degrees of freedom. Allowing an operator to demonstrate the desired motions directly can often enable more intuitive and efficient interaction. Two challenges in the field of learning from demonstration include (1) how to best represent learned motions to accurately reflect a human's intentions, and (2) how to enable learned motions to be easily applicable in new situations. This paper introduces a novel representation of continuous actions called probabilistic flow tubes that can provide flexibility during execution while robustly encoding a human's intended motions. Our approach also automatically determines certain qualitative characteristics of a motion so that these characteristics can be preserved when autonomously executing the motion in a new situation. We demonstrate the effectiveness of our motion learning approach both in a simulated two-dimensional environment and on the All-Terrain Hex-Limbed Extra-Terrestrial Explorer (ATHLETE) robot performing object manipulation tasks.**

I. Introduction

Today, most space robots are directly operated using manual command sequences. While manageable for simple robots, this approach is tedious when controlling robots with many degrees of freedom, such as NASA JPL's ATHLETE (All-Terrain Hex-Limbed Extra-Terrestrial Explorer), which has 36 independent joints [1]. Currently, operators can either use low-level joint angle commands or choose among a handful of pre-programmed higher-level tasks. Attempting to pre-program all possible tasks is unreasonable on such complex robots. Our solution is to introduce a motion learning approach that uses teleoperated demonstration. The problem of motion learning is to summarize several demonstrated samples of a motion into a generalized representation, which then can be used to autonomously perform the motion in new situations.

Many researchers have investigated different aspects of learning from demonstration [2]. For example, learning human-taught policies has proven useful in domains of underactuated pendulum control [3], autonomous helicopters [4], and vehicle navigation [5]. We are interested in manipulation tasks, where interaction with objects in the environment becomes important. Our approach is inspired by existing work in learning manipulation tasks from demonstration. Peters and Campbell [6] taught the humanoid Robonaut to grasp a tool by time-normalizing and averaging demonstrated motions. Their work showed that learning from teleoperated demonstrations is an attractive approach to controlling complex robots. With a more robust motion representation and better adaptability to new situations, this type of approach will be compelling for a wider range of applications. Researchers at USC [7], [8] modeled learned motions as a spring system described by differential equations with parameterized start and goal locations. In contrast, Mühlig et al. [9] chose to abstract the demonstrated time-series data in task space with Gaussian Mixture Models. Calinon et al. [10] model motions as a mixture of Gaussian Mixture Models and Bernoulli distributions that captures both spatial and temporal aspects of a motion. Inspired by these approaches, our work introduces a new representation that can faithfully capture important features of a human's demonstrated motion, without prior knowledge of the motion's distinguishing characteristics.

Our approach encodes a motion using a novel representation called a *probabilistic flow tube*, defined in the next section. Constraint-based flow tubes have been used in the context of planning and execution with continuous motions, to represent sets of trajectories with common characteristics [11], [12]. A flow tube defines a state region where valid trajectories of a motion can be feasibly achieved given constraints on the system dynamics. We incorporate this concept into motion learning by introducing a *probabilistic* version of a flow tube, computed by inferring the desired state region at each time step from human demonstrations. Geometrically, the breadth of a probabilistic flow tube represents flexibility in the robot's desired movement, enabling it to optimize additional performance criteria or recover from disturbances. We choose the probabilistic flow tube representation because it models motions close to how humans do, since they are directly based on human-generated trajectories. For robots designed to work in the field with other humans nearby, such as in the case of ATHLETE, it is important that any autonomous behavior should be executed in a way humans expect, and not in a way that could cause alarm for the human, even if it means executing a less optimal behavior.

We also introduce the ability to automatically determine relevant variables of a motion directly from observed training sequences, so that the motion's relationship to these variables can be preserved when autonomously executing the motion in a new situation. For example, a motion to "put the block in the bin" is described by variables specifying the locations of the block and bin, whereas a motion to "move two feet to the left" is described by variables specifying distance and direction, unrelated to positions of objects in the environment. Our problem is similar to that of task space selection [13], although instead of the human-robot

The authors are with the Department of Aeronautics & Astronautics, Massachusetts Institute of Technology, 77 Massachusetts Ave. 32-224, Cambridge, MA 02139 {dongs, williams}@mit.edu

correspondence problem, we are more interested in detecting what relevant features, if any, exist in the environment, and when they become relevant in a motion.

In the remainder of the paper, we describe the input data used in our approach, formalize the problem we address, and discuss the effectiveness of our approach both in a simulated environment and on the ATHLETE robot.

II. Observed State Space

In our experiments, the correspondence problem between a human operator and a robot is handled by allowing the human to demonstrate desired motions through teleoperation. However, our approach is not limited to this kind of interaction, because we choose to work in *task space*, tracking only the positions and interactions of points of interest in the task, defined as the robot end effector, objects, and other notable features in the environment. Therefore, any system with suitable sensing capabilities, such as vision or motion capture, and inverse kinematics control for trajectory following [14] can also utilize our approach.

Each point of interest has a state, which may include location $\mathbf{x}$ and orientation $\boldsymbol{\theta}$. We model the world seen by the system as the set of these states, which we call "environment states." Demonstrated motions are captured by the evolution of environment states through time, and information about when the points of interest make contact with one another. Contact information can be either obtained directly from sensing such as force feedback or derived from the positions of the points of interest, which is also applicable for non-teleoperated systems with other sensing capabilities. For the tasks discussed in this paper, it is sufficient to track only the contact variable $c \in \{0, 1\}$ between the robot end effector and other points of interest.

Given B points of interest in an environment and the state vector $\mathbf{s}_b = [\mathbf{x}, \boldsymbol{\theta}, c]$ of each point, the full recorded environment state at each time step n during demonstration is $S_n = [\mathbf{s}_1, \ldots, \mathbf{s}_b, \ldots, \mathbf{s}_B]$. An observed training sequence is then $\mathbf{S} = \left[S_1^T, \ldots, S_n^T \ldots, S_N^T\right]^T$.

III. Problem Statement

The inputs to the motion learning problem are:

- A set of D demonstrated training sequences corresponding to a desired motion, $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2 \ldots, \mathbf{S}_D\}$, which captures the environment states of B points of interest over time. No assumptions are made regarding the similarities of the initial states of the training sequences.
- A new environment state $S_{new} = [\mathbf{s}_1, \ldots, \mathbf{s}_b, \ldots, \mathbf{s}_B]$ capturing the new states of the same points of interest. This specifies the initial environment in which the robot should autonomously execute the learned motion.

The system outputs the learned motion represented as a probabilistic flow tube defined by the following:

- A trajectory sequence of the robot end effector

$$\mathbf{s}_{eff} = \begin{bmatrix} \mathbf{x}_1 & \boldsymbol{\theta}_1 & c_1 \\ \mathbf{x}_2 & \boldsymbol{\theta}_2 & c_2 \\ & \vdots & \\ \mathbf{x}_N & \boldsymbol{\theta}_N & c_N \end{bmatrix}.$$

- Covariances at each corresponding time step

$$\boldsymbol{\Sigma}_{eff} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{x}_1} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}_1} \\ \boldsymbol{\Sigma}_{\mathbf{x}_2} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}_2} \\ \vdots & \\ \boldsymbol{\Sigma}_{\mathbf{x}_N} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}_N} \end{bmatrix}.$$

IV. Motion Learning Approach

Our motion learning approach is summarized in Algorithm 1. First, it determines the important features or relations in the demonstrated motion, which we call *motion variables* ($vars$). Then it uses the training sequences $\mathcal{S}$ to create a probabilistic flow tube $\langle \mathbf{s}_{eff}, \boldsymbol{\Sigma}_{eff} \rangle$ that abides by the same relations ($vars$) in the new environment (S_{new}).

We now proceed to describe each of these steps in detail.

A. Identifying Motion Variables

A key feature of our learning system is the ability to autonomously determine what features or relations, if any, are characteristic of a particular demonstrated motion. We use the general term *motion variables* to describe qualitative features of a motion. The basic idea behind motion variable identification is that the variables relevant to the motion are preserved over different demonstrated trials, while other variables of the motion can vary due to changes in the environment or the human's movement.

For example, if the desired task is "move box to bin," the demonstrated sequences will reveal a pattern where the robot end effector first moves to the location of the box and makes contact with it, then moves to the location of the bin and breaks contact with the box. The system will learn that the distance between the robot effector and the box is a relevant motion variable indicating when to close the gripper, and that the distance between the robot effector and the bin is a relevant motion variable indicating when to release the gripper. The system will also learn that the positions of any other objects known in the environment are not relevant to this motion.

In the implementation described here, we consider as candidate motion variables the (nontrivial) absolute or pairwise relative positions or orientations of all points of interest at all changes of contact. As an example, a motion that moves an object two feet to the right has two relevant variables: the initial location of the object (the absolute state at which the robot effector makes contact), and the relative movement

Algorithm 1 LearnPFT $(\mathcal{S},\ S_{new})$

1: $vars \leftarrow$ IdentifyMotionVars $(\mathcal{S})$
2: $\langle \mathbf{s}_{eff}, \boldsymbol{\Sigma}_{eff} \rangle \leftarrow$ MakePFT $(\mathcal{S},\ S_{new}, vars)$

vector (the relative displacement from the initial state at which the effector breaks contact).

After identifying locations of candidate motion variabes, we use clustering to determine if patterns exist across different training samples for each candidate. A narrow spread in the values of a motion variable across many training samples indicates that the variable is relevant to the motion. Algorithm 2 describes the details of the approach.

The input is a set of D demonstrated training sequences corresponding to a motion, $\{\mathbf{S}_1, \mathbf{S}_2 \ldots, \mathbf{S}_D\}$. Lines 1-5 then determine the key time steps in the motion: in our examples, these are time steps when the robot makes or breaks contact with an object. Figure 1 illustrates two demonstrations of a motion where the robot effector makes contact with an object and breaks contact some time later, as might occur in a pick-up and drop-off action. For illustration purposes, suppose the contact changes at time steps 2 and 8 in demonstration 1, and at time steps 3 and 12 in demonstration 2. Assume demo 1 has a total of 20 time steps, and demo 2 has 25. Then according to our algorithm, $keyTimes = \begin{bmatrix} 1 & 2 & 8 & 20 \\ 1 & 3 & 12 & 25 \end{bmatrix}$

We next determine if there are patterns in the extracted environment states at key time steps that indicate relevant motion variables. In our examples, we restrict candidate motion variables to relative positions $\mathbf{x}$ and orientations $\boldsymbol{\theta}$ between an object (denoted obj) and the robot end effector (denoted eff) at key time steps (denoted α, β).

Using data gathered from the demonstrated sequences, our

Algorithm 2 IDENTIFYMOTIONVARS $(\mathcal{S})$

Input:
$\mathcal{S}$, set of demonstrated sequences $\{\mathbf{S}_d : d = 1..D\}$
Output:
$vars$, set of motion variable record entries (initially $\emptyset$)

1: **for** $d = 1$ to D **do**
2: $keyTimes\,(d) \leftarrow \{1, length\,(\mathbf{S}_d)\}$
3: **for** $n = 2$ to $length\,(\mathbf{S}_d)$ **do**
4: **if** $\mathbf{S}_d\,(n)\,.contact \neq \mathbf{S}_d\,(n-1)\,.contact$ **then**
5: Add n to $keyTimes\,(d)$
6: **for** $i, j \in cols\,(keyTimes)$ and $obj \in$ all objects **do**
7: $\alpha = \{keyTimes\,(d, i) : d = 1..D\}$
8: $\beta = \{keyTimes\,(d, j) : d = 1..D\}$
9: $A = \{\mathbf{S}_d\,(\alpha_d) : d = 1..D\}$
10: $B = \{\mathbf{S}_d\,(\beta_d) : d = 1..D\}$
11: $\langle \mu_{abs}, \Sigma_{abs} \rangle \leftarrow$ FITGAUSS $(A.\mathbf{x}_{eff})$
12: **if** $max\,(eig\,(\Sigma_{abs})) < \epsilon$ **then**
13: Add ["abs", μ_{abs}, i] to $vars$
14: $\langle \mu_{obj}, \Sigma_{obj} \rangle \leftarrow$ FITGAUSS $(A.\mathbf{x}_{eff} - A.\mathbf{x}_{obj})$
15: **if** $max\,(eig\,(\Sigma_{obj})) < \epsilon$ **then**
16: Add ["relobj", μ_{obj}, i, obj] to $vars$
17: $\langle \mu_{rel}, \Sigma_{rel} \rangle \leftarrow$ FITGAUSS $(A.\mathbf{x}_{eff} - B.\mathbf{x}_{eff})$
18: **if** $max\,(eig\,(\Sigma_{rel})) < \epsilon$ **then**
19: Add ["relmot", μ_{rel}, i, j] to $vars$
20: {and similarly for orientation $(\boldsymbol{\theta})$}

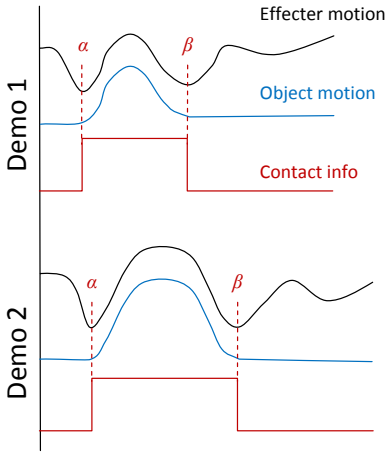

Fig. 1. Motion variable identification

Algorithm 2 checks for consistencies, over all the sequences, of the following types of candidate motion variables:

- Absolute motions (lines 11-13), such as "move to the origin," or "orient gripper down,"
- Motions relative to an object (lines 14-16), such as "move to the box," or "point camera at the box,"
- Motions relative to the robot (lines 17-19), such as "move two feet to the right," or "rotate 30° clockwise."

We determine which of the parameters listed above are statistically similar over the different training sequences by fitting a Gaussian $\langle \mu, \Sigma \rangle$ for each parameter at corresponding key time steps over all the trials. Resulting Gaussians with very narrow spread, i.e. $max\,(eig\,(\Sigma)) \leq \epsilon$, indicate that the motion variable in question is relevant. Figure 2 visualizes the process of determining the relevance of the variable $(\mathbf{x}_{eff} - \mathbf{x}_{bin})$ at different key time steps for a "move object to bin" motion.

B. Data Processing and Flow Tube Generation

After identifying the motion variables, we know which points of interest in the environment are relevant to the motion in the new situation. The next step is to process the

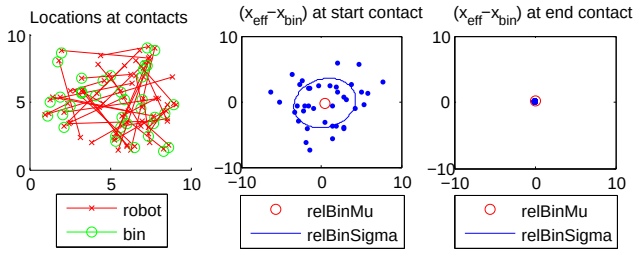

Fig. 2. Clustering identifies motion variables for a "move object to bin" motion. Left: robot and bin locations at the change-of-contact points for each demonstration. Middle: values of the variable $\left(\mathbf{x}^{\alpha}_{eff} - \mathbf{x}^{\alpha}_{bin}\right)$ at the start of contact—the large spread indicates low relevance. Right: values of $\left(\mathbf{x}^{\beta}_{eff} - \mathbf{x}^{\beta}_{bin}\right)$ at the end of contact—the narrow spread indicates that this variable is relevant to this motion.

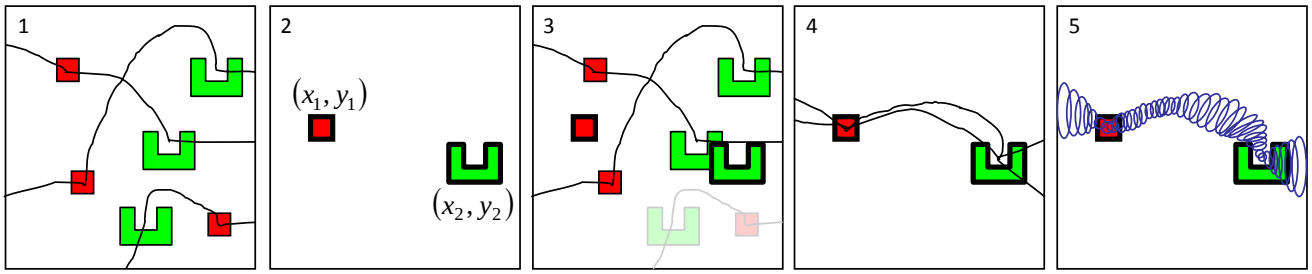

Fig. 3. Illustrated steps of our approach with three demonstrations of the "move the box to the bin" task in our two-dimensional simulation environment

training data into a format with which we can create the probabilistic flow tube. Algorithm 3 describes this process.

The algorithm can be summarized intuitively in the following steps, with numbers corresponding to the illustrations in Figure 3:

1) *Given the relevant motion variables*: In Figure 3, we have identified that the locations of the box and bin are relevant.
2) *Extract relevant states in new environment* (line 1): During autonomous execution of the demonstrated task in the new environment, we assume the system can use sensing to determine the new environment states. This step simply extracts those states that are relevant based on the identified motion variables. In the example, the system will examine the new scene and extract the locations of the box and bin using sensing.
3) *Gather similarly initialized demonstrations* (line 2): Identify a subset of the demonstrated data sequences that have a relevant initial environment most similar to the new situation. In the example, the system will select a subset of demonstrations in which the relative initial positions of the box and bin are most similar to those in the new situation.
4) *Normalize demonstration sequences* (line 3): Normalize the selected subset of demonstrated data sequences to fit the values of the motion variables for the new situation. In the example, the system will scale and rotate the data sequences so that the robot end effector location upon initial contact with the box matches the box location in the new situation, and the effector location upon releasing contact with the box matches the bin location in the new situation.
5) *Generate flow tube* (lines 4-13): Temporally match all the space-normalized sequences, and create a probabilistic flow tube to be used for autonomous execution. This is accomplished through the use of dynamic time warping, which we discuss next.

Algorithm 3 MAKEPFT $(\mathcal{S},\ S_{new}, vars)$

Input:

$\mathcal{S}$, set of demonstrated sequences $\{\mathbf{S}_d : d = 1..D\}$

S_{new}, a new environment state

Output:

$\mathbf{s}_{eff}$, trajectory sequence of robot end effector

$\mathbf{\Sigma}_{eff}$, covariances at each corresponding time step

1: $S'_{new} \leftarrow$ EXTRACTRELEVANTSTATES $(S_{new}, vars)$
2: $\mathcal{S}' \subseteq \mathcal{S} \leftarrow$ SIMILARINITCONDSEQS $(\mathcal{S}, S'_{new})$
3: $\mathcal{S}'' \leftarrow$ NORMALIZESCALEROTATE $(\mathcal{S}', vars, S'_{new})$
4: $\mathbf{s}_{eff} \leftarrow \mathcal{S}''(1).\mathbf{s}_{eff}$
5: **for** $d = 2$ to D **do**
6: $\quad \mathbf{w} \leftarrow$ DYNAMICTIMEWARP $(\mathbf{s}_{eff}, \mathcal{S}''(d).\mathbf{s}_{eff})$
7: $\quad \mathbf{s}_{eff} \leftarrow \frac{1}{d}\left[(d-1)\,\mathbf{s}_{eff}(\mathbf{w}_{:,1}) + \mathcal{S}''(d).\mathbf{s}_{eff}(\mathbf{w}_{:,2})\right]$
8: **for** $d = 1$ to D **do**
9: $\quad \mathbf{w} \leftarrow$ DYNAMICTIMEWARP $(\mathbf{s}_{eff}, \mathcal{S}''(d).\mathbf{s}_{eff})$
10: $\quad \mathbf{s}_d \leftarrow \mathcal{S}''(d).\mathbf{s}_{eff}(\mathbf{w}_{:,2})$
11: $\quad \mathbf{s}'_d \leftarrow$ INTERPOLATE $(\mathbf{s}_d, length(\mathbf{s}_{eff}))$
12: **for** $n = 1$ to $length(\mathbf{s}_{eff})$ **do**
13: $\quad \mathbf{\Sigma}_{eff}(n) \leftarrow$ COVARIANCE $\{\mathbf{s}'_d(n) : d = 1..D\}$

We use dynamic time warping (DTW) [15], [16] to temporally match observed sequences. Intuitively, dynamic time warping temporally deforms two sequences to minimize the overall difference between them. The basic algorithm takes two recorded state sequences $\mathbf{R} = [R_1, R_2 \ldots, R_m]^T$ and $\mathbf{S} = [S_1, S_2 \ldots, S_n]^T$, and creates an $m \times n$ local cost matrix with entries containing the pairwise distances between all the data points in both sequences, $c_{ij} = |R_i - S_j|$, where $i \in \{1 \ldots m\}$ and $j \in \{1 \ldots n\}$.

Any temporal matching of the two sequences corresponds to a traversal of the cost matrix from the element matching the origin of the two sequences, c_{11}, to the opposite corner, c_{mn}. Thus the problem of finding the best temporal matching reduces to finding the traversal of the cost matrix that results in the least total cost. Dynamic programming is employed to find this optimal matching by computing the minimal cumulative cost:

$$C_{i,j} = \begin{cases} \infty, & \text{if } i = 0 \text{ or } j = 0 \\ c_{ij}, & \text{if } i, j = 1 \\ c_{ij} + \min \begin{Bmatrix} C_{i-1,j-1} \\ C_{i-1,j} \\ C_{i,j-1} \end{Bmatrix}, & \text{otherwise} \end{cases}$$

The minimal cumulative cost at the last entry, $C_{m,n}$, is the minimal total cost, and the path taken to achieve it reflects the best matching between the two sequences. If the two sequences are very similar, the traversal of the cost matrix

will be near diagonal. This optimal matching is represented as a $p \times 2$ matrix $\mathbf{w}$ containing the indices of $\mathbf{R}$ and $\mathbf{S}$ such that $\mathbf{R}(\mathbf{w}_{i,1})$ is aligned with $\mathbf{S}(\mathbf{w}_{i,2})$, where $p \geq m, n$ is the number of elements along the matched path.

Using dynamic time warping, we can determine the mean of two trajectories $\mathbf{R}$ and $\mathbf{S}$ as $\frac{1}{2}(\mathbf{R}(\mathbf{w}_{i,1}) + \mathbf{S}(\mathbf{w}_{i,2}))$. Referring back to our set of normalized demonstrated sequences $\mathcal{S}''$ in Algorithm 3, we can iteratively compute a representative mean sequence using this procedure (lines 4-7). This is the output trajectory sequence of the robot end effector $\mathbf{s}_{eff}$.

The demonstrated sequences may have different numbers of data entries, so we use dynamic time warping again to temporally match each of the normalized demonstrated sequences in $\mathcal{S}''$ to the mean sequence $\mathbf{s}_{eff}$, and interpolate so that all have the same number of data entries (lines 8-11). Finally, we compute covariances at each corresponding time step across the temporally matched normalized demonstrated sequences (lines 12-13).

V. Experimental Results

We first tested our approach using a two-dimensional simulated environment, and later demonstrated the system working on the All-Terrain Hex-Limbed Extra-Terrestrial Explorer (ATHLETE) robot. For simplicity, we employed only the position component of each environment state.

A. Two-dimensional Simulation Results

In our simulated environment, there are three entities: a red box, a green bin, and a stationary location marked x. The box and bin are positioned at random locations, while the x always marks the fixed location $(5, 5)$. A user can teach a motion by moving the mouse around in the region. Pressing and releasing the mouse button simulates gripping and releasing an object if one is present. The user labels each demonstrated sequence with the name of the motion.

In our first experiment, we taught the system three different motions: "move the box to the bin," "move the box left one unit," and "move the box to x." During each demonstrated trial, the box and bin locations were randomly generated. Fifty demonstrations generated by two users were recorded for each motion. In our experiment, 30 randomly chosen demonstrations of each motion were used for training, and the remaining demonstrations were used for testing. Figure 4 shows some example results.

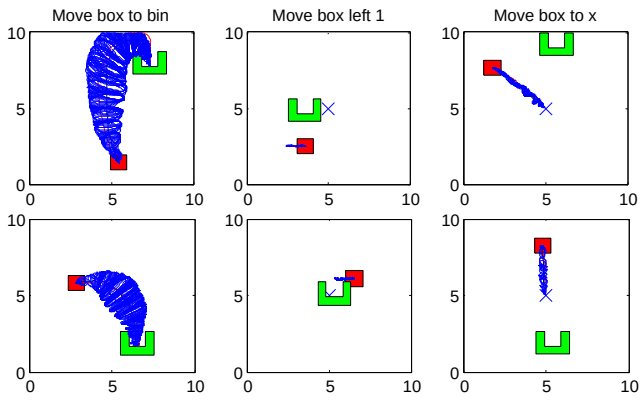

Fig. 4. Example output for three different types of motions. Black lines show mean flow tube trajectory and blue lines show covariances. Compare with red lines that show demonstrated trajectories for these test cases.

As seen in Table I (top), of the 60 test scenarios, our system correctly classified the human input with 93% success. This demonstrates the ability of our approach to recognize motions without prior knowledge of the motion type.

TABLE I

Classification of 20 input test sequences using probabilistic flow tubes (PFT) and Gaussian mixture models (GMM).

		User inputs		
	Classified as	Move to bin	Move left	Move to x
PFT	Move to bin	20	0	3
	Move left	0	19	0
	Move to x	0	1	17
GMM	Move to bin	15	4	0
	Move left	3	11	13
	Move to x	2	5	7

B. Comparison with Prior Art

We also compared our approach to a recent approach by Mühlig et al. [9]. Mühlig's approach also uses dynamic time warping to temporally match demonstrated trajectories, but uses Gaussian mixture models (GMM) to describe learned motions. These GMMs are generated using Expectation Maximization with a Bayesian Information Criterion to determine the optimal number of Gaussians. Mühlig's approach assumes prior knowledge of the type of motion; for comparison purposes, we used our algorithm for motion variable identification, and then normalized all trajectories to the appropriate start and end positions before applying the GMM. Table I (bottom) shows the results of this approach using the same motion data as in the previous section. This approach correctly classified 75% of the human's "move box to bin" input sequences, and averaged 55% success over all three motions. These results indicate that probabilistic flow tubes can offer a more useful representation than GMMs for recognizing these types of motions.

The GMM representation suffers particular drawbacks in cases where temporal ordering of a motion is important, since it is computed based on spatial coordinates alone. For instance, GMMs cannot distinguish a clockwise circular motion (such as winding a cable) from a counter-clockwise motion (unwinding), since they occupy the same spatial

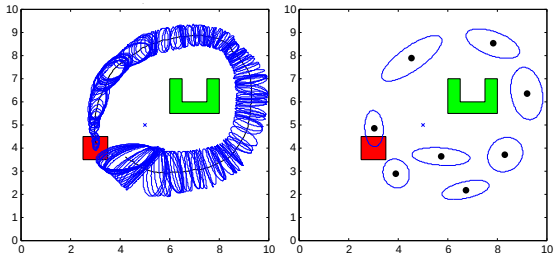

Fig. 6. Example output of an "encircle bin clockwise with box" motion. Left: our probabilistic flow tube approach. Right: GMM approach.

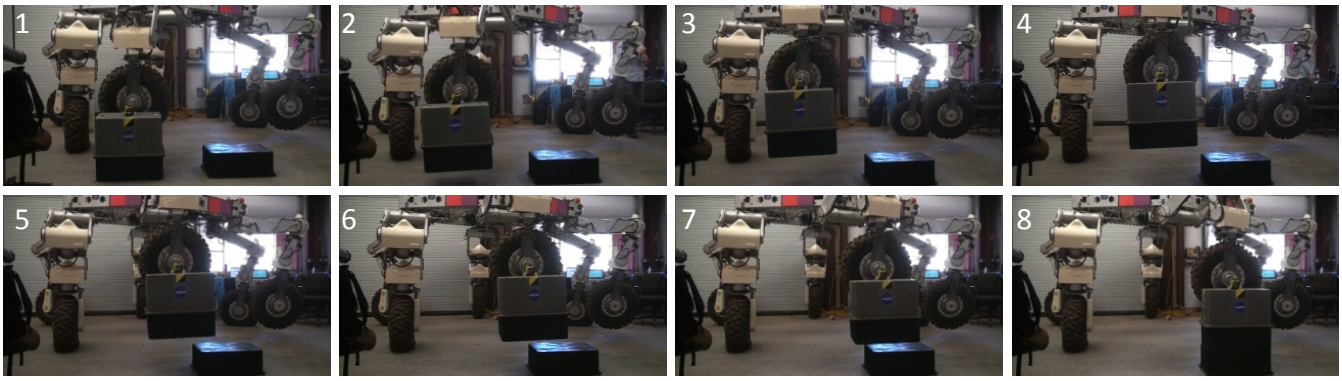

Fig. 5. ATHLETE executing a learned motion in a new situation

region. Figure 6 compares the GMM representation of such a learned motion with our probabilistic flow tube representation, which incorporates temporal ordering information. In this example, our approach recognized the direction of motion in all test cases, while the GMM approach performed only slightly better than random chance.

C. Hardware Demonstration

We demonstrated our motion learning capability on the ATHLETE robot at NASA JPL. Designed to support future planetary surface missions, ATHLETE can roll or walk over rough terrain, and load, transport, manipulate, and deposit payloads. While manually commanding ATHLETE through complex tasks can be extremely tedious, our approach will allow operators to teach the robot desired motions. While teaching, an operator uses an interface device called the Telerobotic ATHLETE Controller for Kinematics (TRACK) [1] to teleoperate the robot.

The task in our experiment was to pick up a large box and move it to the top of a platform. We demonstrated the motion 5 times, each with different initial positions of the box and platform. Figure 5 shows the robot autonomously executing the learned motion given new initial positions.

VI. Conclusion

We have presented an approach to learning complex physical motions from human demonstration using two key ideas: (1) the relevant variables of a motion can be automatically determined through statistically analyzing demonstrated sequences, and (2) a probabilistic flow tube representation enables a more useful description of these demonstrations.

In future extensions of this work, we will investigate more complex spatial properties of demonstrated motions; for instance, how to determine if a trajectory is "avoiding" an object. Using only contact information as the cue to determining motion variables does limit automatic variable identfication to motions where the robot has clearly changing interactions with the environment. Exploring other attributes of a motion that can augment variable identification is an interesting area for future research.

VII. Acknowledgement

This research was supported by a National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a. Additional support was provided by a NASA JPL Strategic University Research Partnership.

References

[1] D. Mittman, J. Norris, M. Powell, R. Torres, and C. McQuin, "Lessons learned from All-Terrain Hex-Limbed Extra-Terrestrial Explorer robot field test operations at Moses Lake Sand Dunes, Washington," in *AIAA SPACE*, 2008.

[2] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 3, pp. 469–483, 2009.

[3] C. G. Atkeson and S. Schaal, "Robot learning from demonstration," in *Fourteenth International Conference on Machine Learning*, 1997, pp. 12–20.

[4] A. Coates, P. Abbeel, and A. Ng, "Apprenticeship learning for helicopter control," *Communications of the ACM*, vol. 52, no. 7, pp. 97–105, July 2009.

[5] P. Abbeel, D. Dolgov, A. Y. Ng, and S. Thrun, "Apprenticeship learning for motion planning with application to parking lot navigation," in *International Conference on Intellegent Robots and Systems*, 2008.

[6] R. A. Peters and C. L. Campbell, "Robonaut task learning through teleoperation," in *IEEE International Conference on Robotics & Automation*, 2003.

[7] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *ICRA*, 2009.

[8] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance," in *ICRA*, 2009.

[9] M. Mühlig, M. Giengerand, S. Hellbachand, J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *ICRA*, 2009.

[10] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 37, pp. 286 – 298, 2007.

[11] A. Hofmann and B. Williams, "Exploiting spatial and temporal flexibility for plan execution of hybrid, under-actuated systems," in *AAAI*, 2006.

[12] H. Li and B. Williams, "Generative planning for hybrid systems based on flow tubes," in *ICAPS*, 2008.

[13] M. Mühlig, M. Gienger, J. J. Steil, and C. Goerick, "Automatic selection of task spaces for imitation learning," in *IROS*, 2009.

[14] M. Hersch and A. G. Billard, "Reaching with multi-referential dynamical systems," *Autonomous Robots*, vol. 25, pp. 71 – 83, 2008.

[15] C. S. Myers, L. R. Rabiner, and A. E. Rosenberg, "Performance trade-offs in dynamic time warping algorithms for isolated word recognition," *Journal of the Acoustical Society of America*, vol. 66, no. S1, pp. S34–S35, 1979.

[16] P. Senin, "Dynamic time warping algorithm review," University of Hawaii at Manoa, Tech. Rep., 2008.