The Application of Deep Learning in Movement Segmentation

Kevin Hao, Riddhi Bhave, and Ilham Thomson

***Abstract*-Although the concept of movement primitives has been explored extensively, there is still much to uncover for movement primitive libraries. The goal of this work is to segment various demonstrations into movement primitives and create a library that can define unknown trajectories. While current work takes advantage of the expectation-maximization algorithm to build a movement primitive library, this work demonstrates the use of neural networks to predict movement segmentation. We showcase this approach with real robot applications, including assembling a sandwich with ingredients placed at different locations on a table.**

Keywords-movement segmentation; movement primitive; neural network; forward kinematics; StitchMP

I. INTRODUCTION

The focus of modern robotics is to teach robots to learn new tasks. A common approach to this issue is imitation learning, a method in which a robot is given a task that it reproduces and improves upon. Solving complex tasks in a single movement primitive can be challenging, so in order to reduce the complications of difficult tasks, a movement can be segmented into multiple movement primitives. Different combinations of these movement primitives can be used to identify and define unique demonstrations. However, a problem with this approach is building a movement primitive library without relying on hand labeled demonstrations. There is a key proposed solution to this issue by Lioutikov, Neumann, Maeda, and Peters (2017). This includes an iterative expectation-maximization algorithm that converges to a set of movement primitives. While their work takes on a similar objective, this work highlights an alternative approach: using neural networks to segment demonstrations. In addition to neural networks, this work will reference the use of probabilistic movement primitives (ProMPs) for representing and learning new movement primitives, an idea proposed by Paraschos, Daniel, Peters, and Neumann (2010). Our approach of using neural networks in conjunction with ProMPs can be useful when attempting to create movement primitive libraries from various spaces, with this work using joint space. Ideally, with the creation of this library, the goal is to design a framework that can learn the movement primitive components of unlabeled trajectories. This implementation can prove useful when training robots to complete tasks in non technical fields.

II. METHOD AND APPROACH

Symbol Definitions

- w - weights to apply to each radial basis function
- Φ - radial basis functions
- Ψ - block radial basis function
- λ - penalty coefficient
- μ_w - mean weights
- Σ_w - weight covariance
- $\mu_w^{[new]}$ - conditioned mean weight
- $\Sigma_w^{[new]}$ - conditioned weight covariance
- Y - trajectory
- t - time to condition on
- y_t^* - desired position at time t
- Σ_y^* - observation strength
- Ψ_t - block radial basis function at time t
- p - precision
- r - recall
- TP - number of true positives
- FP - number of false positives
- FN - number of false negatives

In the world of robotics, movement primitives are essential for the representation of simple movements. Complicated demonstrations can be broken down into smaller, more understandable movements, and movement primitives can be assembled into elaborate tasks. Expanding on this concept, we can use multiple demonstrations of a movement primitive in order to create a more robust representation of its movement. Hence, we use the time invariant ProMPs. Instead of representing a single trajectory, a ProMP represents a distribution over possible trajectories in joint space.

To generate these ProMPs, we first record multiple demonstrations of movements in joint space. After recording our demonstrations, we transform them into weight space using ridge regression.

$$w = (\Psi^T\Psi + \lambda I)^{-1}\Psi^T Y$$

After generating these weights for each demonstration, we generate a distribution over the weights. Now that we have a distribution in weight space, we can condition the distribution to a certain point in time with a certain observation strength.

$$L = \Sigma_w\Psi_t\,(\Sigma_y^* + \Psi_t^T\Sigma_w\Psi_t)^{-1}$$

$$\mu_w^{[new]} = \mu_w + L(y_t^* - \Psi_t^T\mu_w)$$

$$\Sigma_w^{[new]} = \Sigma_w - L\Psi_t^T\Sigma_w$$

After generating ProMPs from this data, we can create stitched movement primitives (StitchMPs) using Algorithm 1. We can transform the results of this algorithm from joint space to Cartesian space using forward kinematics. With these StitchMPs and segmentation locations, we can train our neural network to segment tasks in Cartesian space. Our neural network parameters are now trained to take in trajectories in Cartesian space and identify segments despite its primitive library existing in joint space. We then tested our neural network using trajectories outside of our training set.

Algorithm 1. Creating StitchMPs

Input: Movement primitives recorded in joint space

Output: Stitched together tasks in Cartesian space

Segmentation locations

Use euclidean distance to generate a mapping between movement primitives to identify which ones can fit together sequentially.

N: random length of stitched together task

Randomly generate N movement primitives that fit together sequentially

Randomly generate a length for each movement primitive

For each random movement primitive

Condition the beginning of the ProMP with the previous movement primitive

Condition the end of the ProMP with the next movement primitive

Sample from the conditioned ProMP using the corresponding number of time steps

Append to entire task

Transform from joint space to Cartesian space

We evaluated the accuracy of our neural network by using F-score. Segmentations within a neighborhood of the ground truth are considered true positives, while more than one determined segmentation within the neighborhood of the ground truth are considered false positives. In addition, a false negative is the lack of segmentation within the neighborhood of the ground truth.

$$p = \frac{TP}{TP + FP}$$

$$r = \frac{TP}{TP + FN}$$

$$\text{F-score} = \frac{2pr}{p + r}$$

III. Evaluation

Evaluation was carried out through the use of joint space robot data. Collection of data was done using shadowing, wherein a human operator performed actions which were recorded on the mimicking robot, as seen in Figure 1. The data collected was designed to represent larger trajectories capable of being split into smaller movement primitives, in order to make it compatible with the StitchMP generator. For our purposes, we strung together distinct picking up and putting down motions: actions associated with putting together a sandwich. Evaluation of the StitchMP generator was based on plausibility of the results. If the StitchMP resembled a reasonable trajectory made up of the chosen movement primitives, it was considered correct.

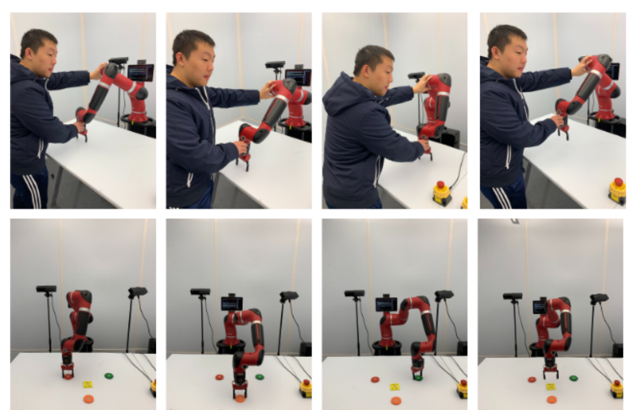
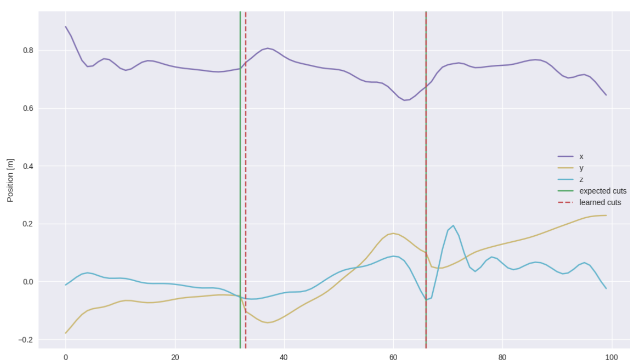


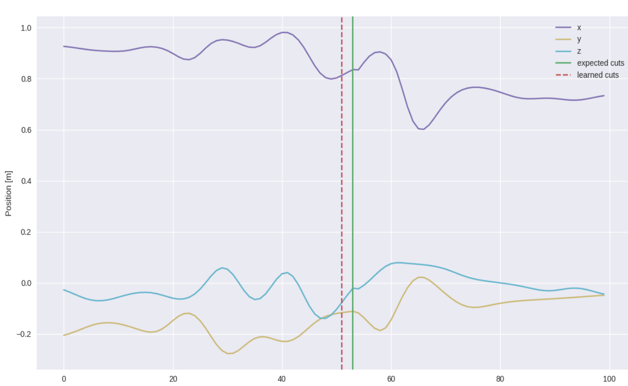
Figure 1. Human demonstration of movement primitives in joint space. Demonstrated the same movements 15 times in order to create a library of movement primitives. These movement primitives were then stitched together to create training data for the neural network.

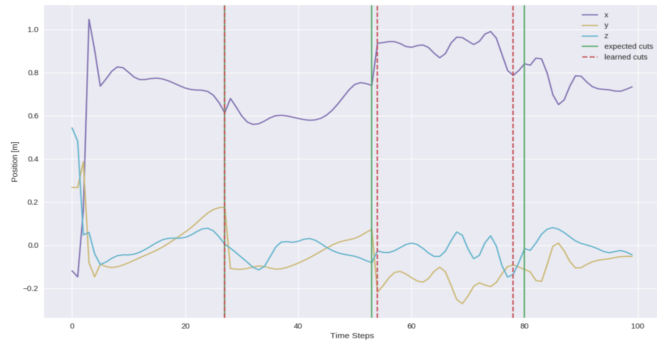
The final step of evaluating the framework was feeding the transformed Cartesian trajectories into our neural network in order to see if it could learn the appropriate segmentation points. The preliminary results in Table 1 show that the neural network was able to learn segmentation points to a certain degree. The general trend found was that the larger the training and testing data sets, the more accurate the neural network became. In addition to looking at accuracy values, this stage was evaluated by comparing the true segmentations with the learned segmentations, as in Figure 2. The components observed were the x, y, and z values of the Cartesian trajectories, with both the expected and learned segmentations being displayed in order to observe their closeness. The relative similarity between the two types of segmentations demonstrates that the network was able to identify many of the segmentations correctly, as well as being relatively close otherwise.

Table 1. Relationship between sizes of data sets used by the neural network and the resulting accuracy

Size of Trained Data Set (number of trajectories)	Size of Tested Data Set (number of trajectories)	Accuracy (f-score)
10,000	100	0.908
10,000	1,000	0.915
1,000	100	0.795







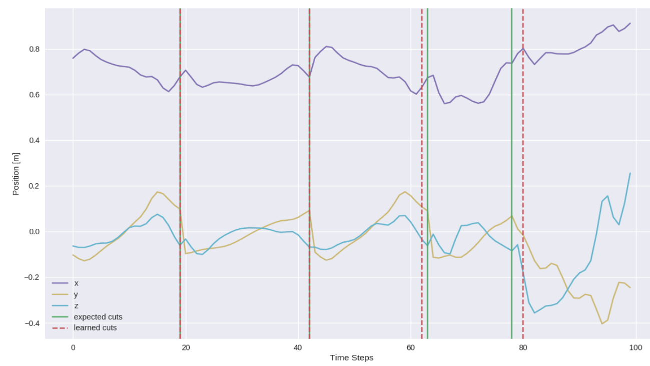


Figure 2. Comparison of ground truth (expected segmentations) and output of neural network (learned segmentations).

IV. Discussion and Conclusion

The use of ProMPs alongside a neural network in order to segment trajectories is a flexible approach to an often variable problem. Due to the neural network being the basis for segmentation, it can easily be tweaked through its parameters, data given, and results expected. Furthermore, our framework enables movement segmentation when we don't have the needed movement primitives in the given space. On the other hand, this flexibility comes along with a lack of complete understanding: sometimes results from the neural network can be questionable, with no real explanation. In addition, the preparation for this framework is somewhat extensive, as a large amount of specific data must be collected to begin the creation of the movement primitive library. Future work would include expanding the starting data set, evaluating if we can reliably return to the pre-transformation space, and attempting new transformations such as between joint space and video space.

Acknowledgment

We would like to thank Dr. Rudolf Lioutikov for his continued support and guidance throughout our project.

References

[1] Lioutikov, Rudolf, et al. "Learning Movement Primitive Libraries through Probabilistic Segmentation." *The International Journal of Robotics Research, vol*. 36, no. 8, 2017, pp. 879-894., doi:10.1177/0278364917713116.

[2] Paraschos, Alexandros, et al. "Using Probabilistic Movement Primitives in Robotics." *Autonomous Robots*, vol. 42, no. 3, 2017, pp. 529-551., doi:10.1007/s10514-017-9648-7.