

Parameter-Free Segmentation of Robot Movements with Cross-Correlation Using Different Similarity Metrics

Wendy Carvalho,* Meriem Elkoudi,* Brendan Hertel, and Reza Azadeh

Abstract— Often, robots are asked to execute primitive movements, whether as a single action or in a series of actions representing a larger, more complex task. These movements can be learned in many ways, but a common one is from demonstrations presented to the robot by a teacher. However, these demonstrations are not always simple movements themselves, and complex demonstrations must be broken down, or segmented, into primitive movements. In this work, we present a parameter-free approach to segmentation using techniques inspired by autocorrelation and cross-correlation from signal processing. In cross-correlation, a representative signal is found in some larger, more complex signal by correlating the representative signal with the larger signal. This same idea can be applied to segmenting robot motion and demonstrations, provided with a representative motion primitive. This results in a fast and accurate segmentation, which does not take any parameters. One of the main contributions of this paper is the modification of the cross-correlation process by employing similarity metrics that can capture features specific to robot movements. To validate our framework, we conduct several experiments of complex tasks both in simulation and in real-world. We also evaluate the effectiveness of our segmentation framework by comparing various similarity metrics.

I. INTRODUCTION

As robotic systems are increasingly expected to perform complex, real-world tasks, the ability to autonomously segment a long-duration task into its constituent sub-tasks is crucial for them to be replicated. In many applications, robots obtain new skills through Learning from Demonstration (LfD) [1], yet the continuous demonstration of a long task often contains multiple sub-tasks that must be accurately identified and segmented for effective learning and reproduction [2], [3]. One example is the table-setting task shown in Fig. 1. To tackle the challenge of robot learning from long and continuous demonstrations, segmentation can be applied to break up the long task into motion primitives (i.e., sub-tasks). However, manually annotating and segmenting demonstrations can be time-consuming and arduous [4]. If robots could identify distinct sub-tasks within a more complex task, they would be better able to learn and replicate complex skills [5].

Here, we propose to segment demonstrations using cross-correlation-inspired techniques. In autocorrelation and cross-correlation techniques, a signal is compared either with itself or another signal to find the correlation between the two signals [6]. This idea can be applied to trajectory

Authors are with the Persistent Autonomy and Robot Learning (PeARL) Lab, University of Massachusetts Lowell, Lowell, MA 01854, USA. Emails: {wendy_carvalho, meriem_elkoudi, brendan_hertel}@student.uml.edu, reza@cs.uml.edu

* Indicates equal contribution.

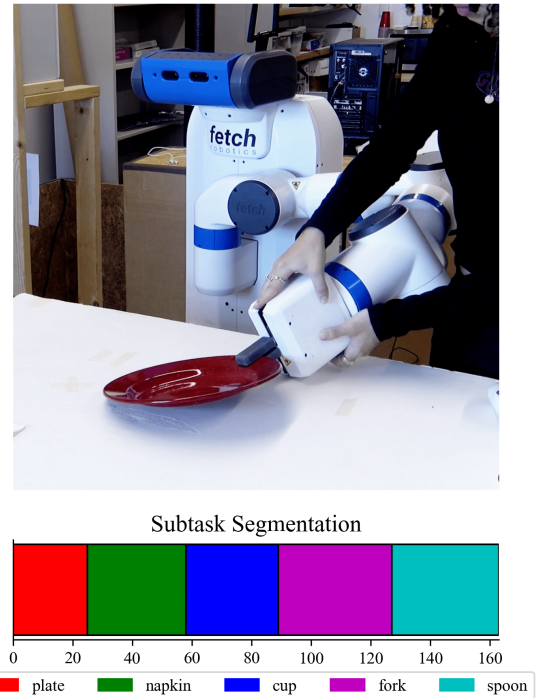


Fig. 1: A full-task demonstration of a table setting task, which contains many sub-tasks. The sequencing of sub-tasks is shown.

segmentation to find the cross-correlation between motion primitive sub-tasks and a long task. For this, a single long demonstration is compared to several pre-recorded “representative” demonstrations of sub-tasks expected to appear within it. However, we consider that there may be variability in these demonstrations, due to differences in velocity, precision, or even environmental conditions. Therefore, a robust similarity metric is essential to perform the correlation effectively. We employ several similarity metrics, including the original cross-correlation function, and compare the results across several environments. While the original cross-correlation function is useful for finding similarity between digital signals, it is not well-suited to finding similarities in robot trajectories. Other similarity metrics proposed here are related to features specific to robot movements, leading to more accurate segmentation. One of the main advantages of this method is that it only requires sub-task and full-task demonstrations, without requiring manual intervention or parameter tuning. Using this segmentation approach, robots can perform complex tasks by applying the detected motion primitives in sequence.

II. RELATED WORK

A major challenge in robot task segmentation is the need to manually tune the parameters involved in segmentation algorithms. Approaches such as state clustering require extensive tuning of parameters to work effectively with different tasks [7]. For example, previous methods enhance the segmentation of tasks by implementing Gaussian Mixture Model (GMM)-based transition clustering [8]. While this improves the reliability of the segmentation by combining visual and kinematic features, the model still relies on intensive tuning of Convolution Neural Networks (CNNs) to incorporate the visual features. In fact, there are several approaches that rely on Deep Learning, which requires many demonstrations [9]. Unlike these methods, our approach only requires a single representative demonstration for each sub-task. Other methods do not take representative sub-task demonstrations, but can have a large number of parameters or parameters that may be difficult to tune. In [10], a segmentation method is proposed that uses changepoint detection, then probabilistically combines these changepoints across multiple modalities. Each modality can have its own parameters for changepoint detection, resulting in a method that is very difficult to tune, and different tunings leading to very different results. Other methods based on Gaussian Processes and Hidden-Semi-Markov Models (GP-HSMM) [11] have been proposed, which accurately segment motions. However, this approach also requires the number of classes to be specified beforehand and is computationally expensive, and, therefore, may not be the most suitable for online robot learning. Later, GP-HSMM was extended using a Hierarchical Dirichlet Process to automatically determine the number of classes, known as HDP-GP-HSMM [12]. This extension, however, exchanges one hyperparameter (number of classes) with other hyperparameters for a proposed stick-breaking process and Dirichlet process.

A parameter-free approach eliminates the need for manually setting parameters such as the number of classes or CNN hyperparameters. This results in a model that is more flexible and scalable, since it does not rely on fixed parameters that may limit its applicability to new tasks or larger datasets. This is particularly beneficial in robotic applications where task dynamics and sensor noise may vary across different scenarios. By removing the need for frequent retuning, a parameter-free model allows for consistent performance, making it well-suited for LfD frameworks and the transfer of skills between different robots or environments [13]. Additionally, reducing training time enables faster learning and execution of motion tasks. This contributes to the growing body of work aiming to make robotic learning more broadly applicable.

III. METHODOLOGY

In this paper, we propose a framework that performs cross-correlation [14] between a long-duration task and its sub-tasks, shown in Fig. 2.¹ Cross-correlation essentially

measures the similarity between observations of a signal at varying points in time. It is useful for detecting patterns and changes in time-series data, making it appropriate for determining transition points between different robot tasks, identifying repetitive patterns, and segmenting said tasks.

When a robot performs a complex task, the motion of its joints and end-effector follows a structured pattern that consists of repetitive movements. Therefore, by observing how data changes over time during a long task and computing its cross-correlation in relation to specific sub-tasks, we can determine the segments of the time-series data that have the highest similarity to each other. However, there are several challenges in developing correlation segmentation, including (i) handling variability in demonstrations, and (ii) finding the most appropriate motion features. To address these challenges, we apply cross-correlation to analyze temporal variations in the robot’s end-effector position for a high-level view of the robot’s spatial trajectory. This end-effector position data forms the basis of our time-series input and is used consistently across both full-task and sub-task demonstrations for computing similarity. We employ several different similarity metrics for calculating this correlation, which enables different and more robust features to be captured in segmentation. Although we investigate three different similarity metrics, others could be used. While we use end-effector data in our experiments, this method could be generally applied to joint data or other modalities as well.

Typically, autocorrelation is used to find a signal obscured by noise, performed by correlating a signal with itself [14]. Essentially, given some time signal f , we wish to compute

$$a^{ff} = f * f, \quad (1)$$

where a^{ff} is the autocorrelated signal and $*$ indicates the correlation operator. This autocorrelated signal shows where f shows the most similarity with itself. When the two signals are different, this is known as cross-correlation [6]. For example, given some noisy time signal g , we can look for f in g by finding where

$$c^{fg} = f * g, \quad (2)$$

is maximized, where c^{fg} is the cross-correlated signal. This can be applied to robot demonstrations, as full-task demonstrations are given as some time signal $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T] \in \mathbb{R}^{T \times d}$, with individual d -dimensional points \mathbf{x}_i . We assume we are also given demonstrations of M sub-tasks in the set $\mathcal{S} = \{\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_M\}$ with an individual sub-task demonstration $\mathbf{Y}_i = [\mathbf{y}_{i,1}, \mathbf{y}_{i,2}, \dots, \mathbf{y}_{i,t_i}] \in \mathbb{R}^{t_i \times d}$ and $t_i < T \forall i = 1, \dots, M$. These “time signals” could be joint-space trajectories, task-space (end-effector) trajectories, or some other mode. We assume that the set of sub-task demonstrations \mathcal{S} is known in advance. If the sub-tasks are not provided, an additional method for their discovery must be applied prior to segmentation, such as those applied in Visual Imitation Learning (VIL) [15]. We wish to find the cross-correlation between each sub-task demonstration and the full-task demonstration, which will reveal where the sub-task demonstration may appear in the

¹<https://github.com/PeARL-robotics/PFCS>

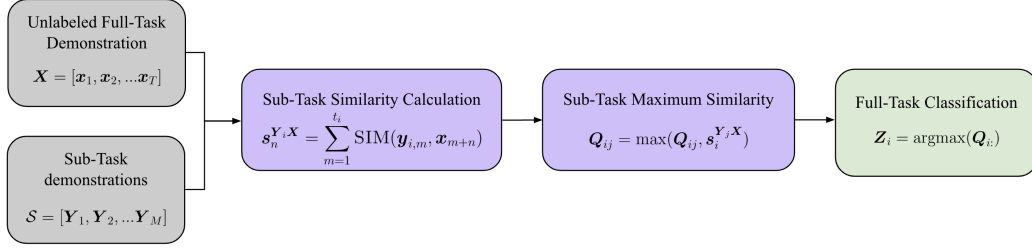


Fig. 2: The proposed framework for our parameter-free similarity-based segmentation approach.

full-task demonstration. For two finite discrete functions, the cross-correlation is defined as

$$c_n^{fg} = (f * g)_n = \sum_{m=1}^N \overline{f(m)} g(m+n), \quad (3)$$

where $\overline{f(m)}$ represents the complex conjugate of f [16]. In the case of robot demonstrations, we only work within the real domain, so $\overline{\mathbf{X}} = \mathbf{X}$. Therefore, we apply this to full-tasks and sub-tasks as

$$c_n^{\mathbf{Y}_i \mathbf{X}} = (\mathbf{Y}_i * \mathbf{X})_n = \sum_{m=1}^{t_i} \mathbf{y}_{i,m} \cdot \mathbf{x}_{m+n}, \quad (4)$$

where $c_n^{\mathbf{Y}_i \mathbf{X}}$ is the cross-correlation between sub-task demonstration \mathbf{Y}_i and the full-task demonstration \mathbf{X} at some point along the demonstration n , and \cdot represents the dot product of two vectors. By doing this for all possible points along the demonstration, we find $c^{\mathbf{Y}_i \mathbf{X}} = [c_0^{\mathbf{Y}_i \mathbf{X}}, c_1^{\mathbf{Y}_i \mathbf{X}}, \dots, c_{T-t_i}^{\mathbf{Y}_i \mathbf{X}}]$, which we can use to find the most likely position of the start of sub-task \mathbf{Y}_i as

$$P_i = \arg \max_n (c^{\mathbf{Y}_i \mathbf{X}}), \quad (5)$$

where P_i is the predicted start of the sub-task.

While cross-correlation does find the likely position of a sub-task within a full-task demonstration, it does not include the ability to be customized to different robot tasks, which may present certain spatial or geometric features [17]. Therefore, we propose a generalization of (4), where an arbitrary similarity metric is used. Different similarity metrics can lead to different results, depending upon the importance of that metric with respect to the task [18]. Cross-correlation with an arbitrary similarity metric takes the form

$$s_n^{\mathbf{Y}_i \mathbf{X}} = \sum_{m=1}^{t_i} \text{SIM}(\mathbf{y}_{i,m}, \mathbf{x}_{m+n}), \quad (6)$$

where $s_n^{\mathbf{Y}_i \mathbf{X}}$ is the *similarity* correlation between sub-task demonstration \mathbf{Y}_i and the full-task demonstration \mathbf{X} at some point along the demonstration n , and SIM is a similarity metric. From this similarity correlation, the predicted start of the sub-task can be found using the same method as presented in (5).² In this paper, we investigate three similarity metrics: cross-correlation similarity (CCS), negative Sum of

Squared Errors (SSE), and tangent cosine similarity (COS). Cross-correlation similarity is the same as cross-correlation presented in (4), as

$$\text{SIM}_{\text{CCS}}(\mathbf{y}_{i,m}, \mathbf{x}_{m+n}) = \mathbf{y}_{i,m} \cdot \mathbf{x}_{m+n}, \quad (7)$$

and gives a higher-is-better similarity metric. SSE is defined as

$$\text{SIM}_{\text{SSE}}(\mathbf{y}_{i,m}, \mathbf{x}_{m+n}) = -\|\mathbf{y}_{i,m} - \mathbf{x}_{m+n}\|_2^2, \quad (8)$$

where $\|\cdot\|_2$ represents the L^2 -norm. Normally, using SSE results in a lower-is-better similarity metric, but we apply a negative to result in a higher-is-better similarity metric. Finally, the tangent cosine similarity is defined as

$$\text{SIM}_{\text{COS}}(\mathbf{y}_{i,m}, \mathbf{x}_{m+n}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|_2 \|\mathbf{b}\|_2}, \quad (9)$$

where $\mathbf{a} = \mathbf{y}_{i+1,m} - \mathbf{y}_{i,m}$ and $\mathbf{b} = \mathbf{x}_{m+n+1} - \mathbf{x}_{m+n}$, or the tangent of $\mathbf{y}_{i,m}$ and \mathbf{x}_{m+n} , respectively. This similarity measures the difference in angle between two vectors, which, when comparing the tangent vectors, results in a comparison of shape over the demonstrations.

The result of (5) only gives the predicted indices of each sub-task. This does not address possible gaps or overlaps in the predictions, which can be addressed in different ways. For some full-task demonstrations, gaps may be necessary, as not every part of the demonstration belongs to one of the sub-tasks. However, it may be important to fully segment a demonstration, and gaps must be filled [10]. Overlaps occur when one sub-task is predicted to start before the previous sub-task has finished. The portion where the overlap occurs could belong to either sub-task, and it should be determined which sub-task is a better fit. To record the best fit for each sub-task, we introduce a sub-task similarity matrix $\mathbf{Q} \in \mathbb{R}^{T \times M}$, which contains the similarity of a full-task demonstration point \mathbf{x}_i with sub-task \mathbf{Y}_j at entry Q_{ij} . This matrix is iteratively updated starting with the first index of \mathbf{X} , as

$$Q_{ij} = \max(Q_{ij}, s_i^{\mathbf{Y}_j \mathbf{X}}), \quad (10)$$

where Q_{ij} will hold the maximum similarity for corresponding full-task demonstration points and sub-task demonstrations. This can then be used to find the sub-task class of a point as

$$\mathbf{Z}_i = \arg \max(\mathbf{Q}_{i,:}), \quad (11)$$

where $\mathbf{Q}_{i,:}$ is the i -th row of the \mathbf{Q} matrix. Note that this means sub-task demonstrations given in \mathcal{S} may not

²here we use higher-is-better similarity metrics which are compatible with arg max, for lower-is-better, arg min should be used.

Algorithm 1: Segmentation via Cross-Correlation with Arbitrary Similarity Metrics

Input: Full-task data \mathbf{X} , Sub-tasks demonstrations \mathcal{S}
Output: Sub-task classes \mathbf{Z}

```

1  $\mathbf{Q} = [-\infty]_{T \times M}$ 
2 for  $i = 1 : M$  do
3    $\mathbf{s}^{Y_i \mathbf{X}} = [0]$  for  $1 : T - t_i$ 
4   for  $j = 1 : T - t_i$  do
5      $\mathbf{s}_j^{Y_i \mathbf{X}} \leftarrow \sum_{m=1}^{t_i} \text{SIM}(\mathbf{y}_{i,m}, \mathbf{x}_{m+j})$ 
6     for  $k = j : j + t_i$  do
7        $\mathbf{Q}_{ki} \leftarrow \max(\mathbf{Q}_{ki}, \mathbf{s}_j^{Y_i \mathbf{X}})$ 
8  $\mathbf{Z} = [-1]_T$ 
9 while  $\max(\mathbf{Q}) > -\infty$  do
10   $i \leftarrow \arg \max(\max(\mathbf{Q}, \text{axis} = 0))$ 
11   $j \leftarrow \arg \max(\mathbf{Q}_{:,i})$ 
12   $k \leftarrow 0$ 
13  while  $\mathbf{Z}_{j+k} == -1$  and  $k < t_i$  and  $j + k < T$  do
14     $k \leftarrow k + 1$ 
15     $\mathbf{Z}_{j:j+k} \leftarrow i$ 
16     $\mathbf{Q}_{:,i} \leftarrow -\infty$ 
17     $\mathbf{Q}_{j:j+k,:} \leftarrow -\infty$ 

```

necessarily show up in the segmentation, as the segmentation will only choose the best sub-tasks.

Alternatively, we can allow gaps in the segmentation, leaving some parts of the demonstration unclassified. In order to do this, we assume each sub-task appears at most once in the full-task demonstration. Then, we iteratively classify points in \mathbf{Z} , starting from the sub-task with the highest similarity across all sub-tasks. For example, if the similarity for sub-task Y_i (stored as $\mathbf{Q}_{:,i}$) contains $\max(\mathbf{Q})$, then we set $\mathbf{Z}_{j:j+t_i} = i$, where $j = \arg \max(\mathbf{Q}_{:,i})$. This ensures that the most likely sub-task is classified first. Then we perform the same operation for the sub-task with the next-highest maximum value, but take care not to overwrite any classes in \mathbf{Z} which have already been assigned. This procedure prioritizes sub-tasks with maximum similarity, but allows for gaps in the segmentation. The pseudocode for this process is shown in Algorithm 1. The time complexity of this algorithm is $\mathcal{O}(MT\bar{t}_i d)$, where \bar{t}_i is the average value for all t_i and assumes the SIM operation is $\mathcal{O}(d)$. Overall, this results in a computationally fast segmentation of the task.

IV. EXPERIMENTS

In this section, we validate the proposed cross-correlation-based segmentation. We first use our method on a simulated handwriting task, then on a real-world table setting task with a Fetch Robotics Fetch Mobile Manipulator as seen in Fig. 1.

A. Segmentation on a Handwriting Task

We tested the proposed method on a simulated 2D trajectory of the word “dog” written in cursive as seen in Fig. 3. The goal of the experiment was to segment the trajectory into sub-task movements corresponding to individual letters (“d,” “o,” and “g”) which were written individually. Here, the full-task demonstration “dog” is the continuously written word.

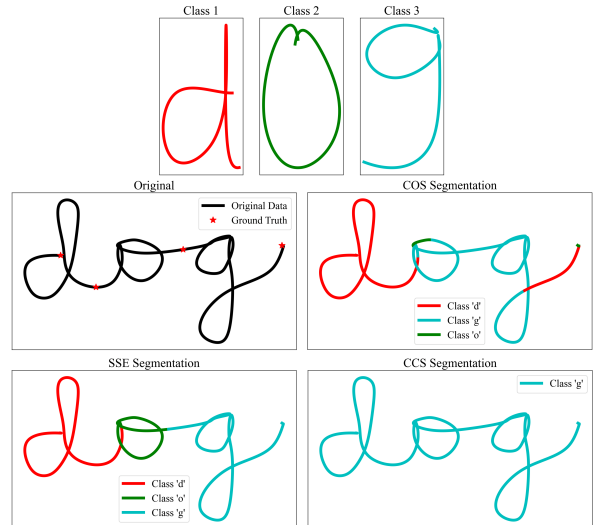


Fig. 3: Segmentation results of the handwriting task using COS, SSE, and CCS as similarity metrics. The representative tasks are displayed in the top row.

TABLE I: Accuracy Results for SSE, COS, and CCS Predicting Each Class and Overall Accuracy

Task	Class 1 (“d”)	Class 2 (“o”)	Class 3 (“g”)	Overall Accuracy
CCS	0.00%	0.00%	100.00%	40.38%
SSE	100.00%	80.50%	100.00%	94.80%
COS	100.00%	13.38%	85.14%	71.25%

The trajectory was captured using a screen capture interface, where a user writes on a screen using a computer mouse. All demonstrations were pre-processed using smoothing and resampling from univariate splines [19].

To perform the segmentation, we applied our approach using all three similarity metrics (CCS, SSE, and COS). The results of the segmentation can be seen in Fig. 3 as well as Table I. In Fig. 3, the original trajectory is plotted in black, while the detected letter segments are displayed in different colors. Visually, the segmentation using SSE intuitively produces correct results for this experiment, as each letter was properly segmented, with only minor errors observed at the endpoints of some letters. The COS metric performed worse than SSE but better than the CCS metric. COS was able to accurately segment the “d” class and only a portion of the “o” and “g” classes. CCS performed significantly worse than the other similarity metrics, where it predicted the “g” class for all data points. This suggests CCS failed to distinguish between the segments effectively. Quantitatively, the accuracy found for each similarity metric further exhibits the discrepancy, as seen in Table I. Here, SSE achieved the best segmentation with an overall accuracy score of 94.79%, followed by COS at 71.25%, while CCS had a much lower overall accuracy score of 40.38%. It is also evident in the individual accuracy scores for each sub-task class where SSE had an accuracy of 100% for class 1 and class 3, and 80.50% for class 2. In comparison, COS achieved 100% accuracy for class 1, 13.38% for class 2 and

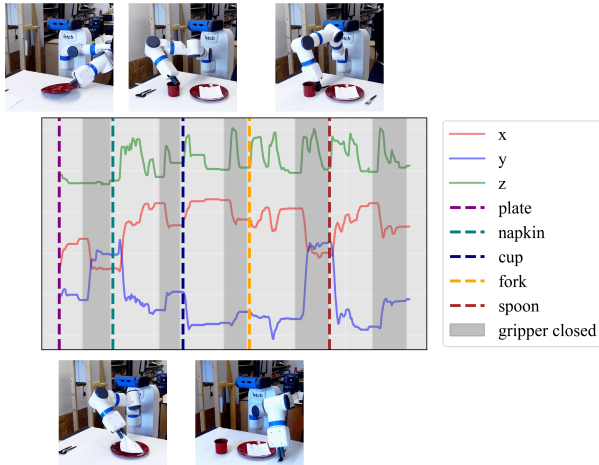


Fig. 4: End-effector position and gripper data for full-task with ground truth indices.

85.14% for class 3. Finally, CCS, which had performed the worst, had accuracy scores of 0% for both classes 1 and 2, and a 100% accuracy for class 3 (this is due to classifying the entire trajectory as class 3). Overall, CCS was less suitable for our 2D task, while both SSE and COS were better at identifying the correct sub-task classes.

B. Segmentation of a Table Setting Task

We also use a real-world table setting task to validate our approach. In this task, several objects must be placed on the table, some of which are ordered (i.e., the plate must be placed before the napkin). First, a demonstration of the table setting task was taken using the Fetch robotic platform, as seen in Fig 1. In the demonstration, a plate, a napkin, a cup, a fork, and a spoon were placed in various locations throughout the robot’s workspace. Each of these object placements is equivalent to a sub-task in our experiments. The robot is guided using kinesthetic teaching to pick and place each of these objects in the defined sequence to set the table. The demonstrations and the ground truth segments, found manually from human labeling, can be seen in Fig. 4. Here, the end-effector position data used for the experiment (x , y , and z) is shown. The ground truth shows five segments, one corresponding to each object that is picked and placed. For each sub-task, a single demonstration is recorded and used as the representative demo for that sub-task. For this experiment, the segmentation is performed in position space.

We also experiment with pre-processing the full-task and sub-task demonstration data through smoothing to mitigate noise and outliers. This ensures consistency across the data, and we find that our method performs better under such conditions. It is important to note that although pre-processing techniques are applied to improve the quality of the data, our segmentation does not rely on such techniques. The pre-processing technique involves the selection of parameters that best optimize the data, distinguishing it from our method, which operates without the need for parameter selection. However, pre-processing steps such as smoothing and time

alignment are ubiquitous across robotics, and we believe it is important to show results considering such techniques.

We incorporate a smoothing step using the Savitzky-Golay filter [20], which is applied separately to each dimension of the data (x , y , and z of the end-effector position). The filter is applied to both the full-task and sub-task demonstration data before performing the cross-correlation-based segmentation. We empirically determined a window size of 301 and polyorder of 2 were the best parameters for this filter.

To evaluate the performance of our method, we find the accuracy of the segmentation. In the ground truth, each point is given a true class Z_{true} . This can be compared with the predicted class Z for each point to find the number of accurately predicted points. We also report the prediction accuracy for each sub-task, as some sub-tasks may have higher performance. We report the accuracy for each of the three similarity metrics with and without smoothing in Table II. In both the raw and smoothed cases, CCS leads to the worst results, SSE finds the second-best results, and COS performs the best. CCS is only ever able to accurately detect some of class 2 and class 4, but fails to find the other classes. These results indicate CCS is not useful for this segmentation method. Since CCS is not designed with robotic trajectories in mind, it cannot leverage properties such as spatial, temporal, or angular similarities that may be present in such trajectories. SSE, on the other hand, can detect most of the classes somewhat accurately, but cannot detect class 5 in both the raw and smoothed cases. This could be due to small shifts in position between the sub-task and full-task demonstrations within the environment, which introduce bias in the positional error SSE is designed to measure. Finally, COS leads to the best results for both the raw and smoothed cases, shown in Fig. 5. The COS metric relies on finding similarities between the shape of demonstrations, which is consistent across environments. Additionally, the performance of COS is greatly increased with smoothing, as the overall accuracy increases from 74.49% to 91.80%. Segmentation with the COS metric performs well across almost all sub-tasks, but there is some error in overwriting of class 1 (plate) early in favor of class 2 (napkin). Additionally, in the raw data case, the segmentation does not detect class 3 (cup), due to temporal differences between demonstrations. This shows how pre-processing can eliminate some errors, leading to better results.

V. CONCLUSIONS & FUTURE WORK

In this paper, we have proposed a segmentation method inspired by the ideas of autocorrelation and cross-correlation. In correlation, a signal is “passed over” another signal to find the similarities or time-shift between the two signals. We apply this idea to find a parameter-free segmentation approach. Additionally, we propose several similarity metrics, which our results indicate perform better than the original cross-correlation similarity metric. We validate our method with a simulated handwriting task as well as a table setting task using a Fetch mobile manipulator in the real-world.

TABLE II: Results from segmenting the table setting task, with raw and smoothed data, across all three metrics. For both raw and smoothed data, the COS metric outperforms the other metrics.

Smoothing	Metric	Class 1 (plate)	Class 2 (napkin)	Class 3 (cup)	Class 4 (fork)	Class 5 (spoon)	Overall Accuracy
Raw	CCS	0.00%	89.15%	0.00%	72.20%	0.00%	34.34%
	SSE	89.51%	26.33%	79.50%	83.81%	0.00%	53.28%
	COS	71.82%	100.00%	0.00%	94.82%	95.63%	74.49%
Smoothed	CCS	0.00%	87.79%	0.00%	71.01%	0.00%	33.80%
	SSE	89.51%	26.94%	79.57%	83.87%	0.00%	53.43%
	COS	72.62%	96.25%	95.05%	94.52%	95.33%	91.80%

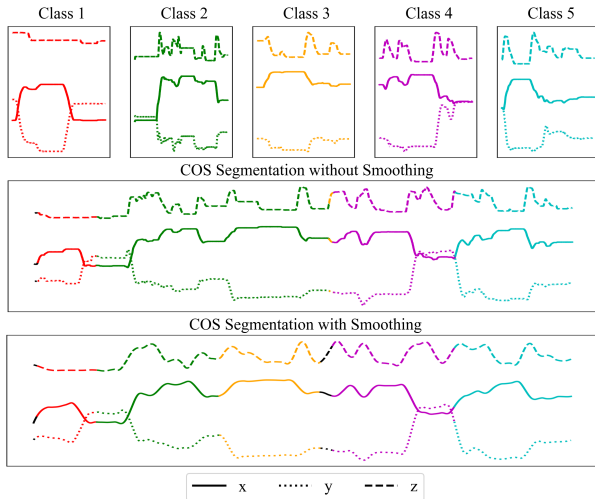


Fig. 5: COS Segmentation of the full-task, with the sub-tasks listed in order, with and without smoothing. The representative tasks are displayed in the top row.

We plan to further refine and extend our approach for robotic sub-task identification in several ways. One possible extension is to improve the similarity measures used. In this work, we propose the use of SSE and COS, which are compared with the original cross-correlation metric. While these similarity metrics improve results, neither is consistently the best metric. Spatial and temporal errors between complex and representative demonstrations can lead to issues in segmentation with these metrics. Finding a metric that can accurately segment all tasks remains a challenge. Further, it may be interesting to develop a way to automatically detect skills that are not present in a sub-task, but are part of the complex task demonstration.

ACKNOWLEDGMENTS

This research is supported in part by the National Science Foundation (FRR-2237463).

REFERENCES

- [1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, “Recent advances in robot learning from demonstration,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, 2020.
- [2] A. H. Kordia and F. S. Melo, “An end-to-end approach for learning and generating complex robot motions from demonstration,” in *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2020, pp. 1008–1014.

- [3] T. Eiband, J. Liebl, C. Willibald, and D. Lee, “Online task segmentation by merging symbolic and data-driven skill recognition during kinesthetic teaching,” *Robotics and Autonomous Systems*, vol. 162, p. 104367, 2023.
- [4] M. Chi, Y. Liu, Q. Zhang, and C. Zeng, “A demonstration trajectory segmentation approach for wheelchair-mounted assistive robots,” *CAA Transactions on Intelligence Technology*, 2024.
- [5] W. Li, Y. Wang, Y. Liang, and D. T. Pham, “Learning from demonstration for autonomous generation of robotic trajectory: Status quo and forward-looking overview,” *Advanced Engineering Informatics*, vol. 62, p. 102625, 2024.
- [6] L. R. Rabiner, B. Gold, and C. Yuen, “Theory and application of digital signal processing,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 2, pp. 146–146, 1978.
- [7] X. Fan, Y. Yue, P. Sarkar, and Y. R. Wang, “On hyperparameter tuning in general clustering problems,” in *International conference on machine learning*. PMLR, 2020, pp. 2996–3007.
- [8] Y. Yamada, J. Colan, A. Davila, and Y. Hasegawa, “Task segmentation based on transition state clustering for surgical robot assistance,” in *2023 8th international conference on control and robotics engineering (ICCRE)*. IEEE, 2023, pp. 260–264.
- [9] S. L. B. Sørensen, T. R. Savarimuthu, and I. Iturrate, “Robot task primitive segmentation from demonstrations using only built-in kinematic state and force-torque sensor data,” in *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2023, pp. 1–7.
- [10] B. Hertel, N. Tran, M. Elkoudi, and R. Azadeh, “A framework for learning and reusing robotic skills,” in *21st International Conference on Ubiquitous Robots (UR)*, 2024.
- [11] T. Nakamura, T. Nagai, D. Mochihashi, I. Kobayashi, H. Asoh, and M. Kaneko, “Segmenting continuous motions with hidden semi-markov models and gaussian processes,” *Frontiers in neurobotics*, vol. 11, p. 67, 2017.
- [12] M. Nagano, T. Nakamura, T. Nagai, D. Mochihashi, I. Kobayashi, and M. Kaneko, “Sequence pattern extraction by segmenting time series data using gp-hsmm with hierarchical dirichlet process,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4067–4074.
- [13] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, “A survey of robot learning from demonstration,” *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.
- [14] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
- [15] A. Jonnavittula, S. Parekh, and D. P. Losey, “View: Visual imitation learning with waypoints,” *Autonomous Robots*, vol. 49, no. 1, 2025.
- [16] C. Wang, “Kernel learning for visual perception,” Ph.D. dissertation, Nanyang Technological University, Singapore, 2019.
- [17] H. Ravichandar, S. R. Ahmadzadeh, M. A. Rana, and S. Chernova, “Skill acquisition via automated multi-coordinate cost balancing,” in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7776–7782.
- [18] B. Hertel and S. R. Ahmadzadeh, “Similarity-aware skill reproduction based on multi-representational learning from demonstrations,” in *20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021.
- [19] P. Dierckx, *Curve and surface fitting with splines*. Oxford University Press, 1995.
- [20] M. J. Niedźwiecki, M. Ciołek, A. Gańcza, and P. Kaczmarek, “Application of regularized savitzky–golay filters to identification of time-varying systems,” *Automatica*, vol. 133, p. 109865, 2021.