Trajectory Learning Using Principal Component Analysis

Asmaa A.E. Osman(✉), Reda A. El-Khoribi, Mahmoud E. Shoman, and M.A. Wahby Shalaby

Faculty of Computers and Information, Cairo University, Dr. Ahmed Zewail St. 5, Giza 12613, Egypt
{Asmaa.A.Elsayed,R.Abdelwahab,M.Essmael, M.Wahby}@fci-cu.edu.eg.com

Abstract. Robots are increasingly used in numerous life applications. Therefore, humans are looking forward to create productive robots. Robot learning is the process of obtaining additional information to accomplish an objective configuration. Moreover, robot learning from demonstration is to guide the robot the way to perform a particular task derived from human directions. Traditionally, modeling the demonstrated data was applied on discrete data which would result in learning outcome distortions. So as to overcome such distortion, preprocessing of the raw data is necessary. In this paper, trajectory learning from demonstration scheme is proposed. In our proposed scheme, the raw data are initially preprocessed by employing the principal component analysis algorithm. We experimentally compare our proposed scheme with the most recent proposed schemes. It is found that the proposed scheme is capable of increasing the efficiency by minimizing the error in comparison to the other recent work with significant reduced computational cost.

Keywords: Robots · Trajectory learning · Learning from demonstration · Principal component analysis

1 Introduction

Currently, robots are widely used by the humans in most of their daily life. Therefore, there is a continuous need to create intelligent robots. For the purpose of producing such robots, some skills should be provided for the robot such as the ability to recognize, analyze and interact with human actions. The essential part of any robotics application is to change the state or the configuration of the robot. Changing the configuration of the robot by the transition from the present state to a goal state to perform required action is called a policy. Employing the machine learning techniques can enable the robot to develop these policies [1]. Consequently, combination between the robotic and the machine learning is used in robot learning approach.

Modeling the set of demonstrations is achieved through several techniques such as Gaussian mixture models (GMMs), conditional random field, and hidden Markov models (HMMs) [2–5]. An approach presented by Calinon and Billard [3] was based on using the concept of the key points to generate a generalized trajectory. The authors only used the key points of the sequence with the highest log likelihood. The weakness of this approach is in its dependence on the key points from only one trajectory which had the highest log likelihood. Therefore, some important key points would be missed in the generalized trajectory.

In [4], Asfour et al. proposed another key points approach where the common key points among all demonstrations were used in generating the generalized trajectory. The weakness of this approach was their dependent on the common key points only. Therefore, the generalized trajectory would miss an important key point just because it was missed in one of the observation sequences. Another approach that used the concept of the key points was proposed by Vakanski et al. [5]. The authors used the key points from all demonstrations for the trajectory generalization. Set of initial key points are modeled by discrete HMM (DHMM) and the key points are temporally aligned by applying dynamic time warping algorithm (DTW). They considered the variance among the key points by setting weighting coefficient for each set of the key points. The aligned key points and the weighting coefficients are interpolated by cubic spline interpolation. The drawback of their approach was using the DHMM directly to model the initial key points which may result in local distortion in the demonstrated data.

Another approach for learning of the trajectories by demonstration was established by M. Field et al. [6] where probabilistic motion primitives were used for representing the demonstrated trajectories. Factor analyzers algorithm was used for reducing the dimensions included in the demonstrated data. The approach of learning from demonstration used in [6] was the observational learning, in which a human demonstrates the motion by his own body, then the motion is mapped to the kinematics of the robot. The observational learning is used in a lot of applications including the surgical applications [7]. The Kinesthetic teaching is another approach for learning from demonstration. In this approach, the motion is performed using the body of the robot to avoid the need to interpret the body of the human [5]. The kinesthetic is employed in many applications especially important on highly-dynamic tasks [8]. The learning from demonstration approach used in our proposed scheme is the kinesthetic teaching.

It is found from the literature that the stage of modeling the demonstrated data was applied by discretizing the raw data which may lead to distortion in the learning outcome. Therefore, a preprocessing stage of the data, prior to identifying the initial key points, is required to solve such distortion. In this paper, we propose a new scheme for trajectory learning from demonstration by employing the principal component analysis. PCA is used for preprocessing the raw data. The preprocessed data is used in identifying the initial key points instead of using the raw data. HMM is used to model the set of the initial key points. Afterwards, dynamic time warping is applied to the temporal alignment of the key points. Each set of the key points are assigned a weighting coefficients to consider the variance among the different parts of the trajectories. Finally, cubic spline interpolation is used to interpolate the set of the key points and the assigned weights to generate a generalized trajectory.

The remainder of this paper is organized as follows: the proposed scheme is discussed in Sect. 2. In Sect. 3 the experimental work is shown, and hence in Sect. 4 the results, comparison and discussion are reported. Finally, the work presented in this paper is concluded in Sect. 5.

2 Proposed Scheme

As mentioned earlier, a new trajectory learning model is proposed using PCA in a preprocessing stage. The aim is to produce a generalized trajectory using set of demonstrated trajectories. The input data for our proposed model is principally acquired by teaching the robot the way to do particular task. The data represents a set of demonstrated trajectories each one of them X_m where m $\in \{1, 2, \ldots,$ M$\}$ has 6-D measurements for the position and orientation data.

2.1 PCA Based Preprocessing

For the PCA based preprocessing stage, raw demonstrated data is considered to be the input, and the reconstructed data is the output of this stage. Before reconstructing the data using PCA, the data points which include NANs are first interpolated from the raw data. A block diagram of the proposed model is shown in Fig. 1.

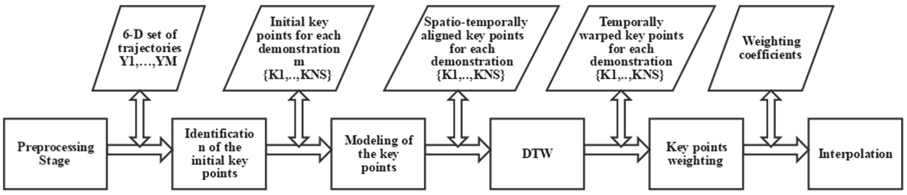
6-D set of trajectories Y1,...,YM
Initial key points for each demonstration m {K1,..,KNS}
Spatio-temporally aligned key points for each demonstration {K1,..,KNS}
Temporally warped key points for each demonstration {K1,..,KNS}
Weighting coefficients
Preprocessing Stage
Identification of the initial key points
Modeling of the key points
DTW
Key points weighting
Interpolation

Fig. 1. Block diagram of the proposed scheme.

PCA is applied on the original data set to analyze it and accounts for the variation included in the original variables. The PCA is used to represent the most important information included in a data set into another set of uncorrelated variables called principal components, each of which is a linear combination between the original variables [9]. For raw data set under consideration, a set of demonstrated trajectories are represented. Each trajectory contains a group of points. Such that each trajectory point is represented in 6-dimensional space: the location (x, y, z) and orientation (roll, pitch, yaw). In our proposed scheme, PCA technique is applied on this raw data as follows:

Adjusting the data. The data are adjusted by subtracting the mean across each dimension. The mean $\bar{x}$ is the average of a given data set x. All x values are subtracted from $\bar{x}$ which is the average of this dimension. Also all y values are subtracted by $\bar{y}$ and so on for all the other 4 dimensions. The adjusted data sets are referred to as X^{AD}.

Calculating the covariance matrix. The covariance matrix is a quantity to measure the variation from the mean across the data set. The covariance is used for the 2-dimensional data but if we have 3-dimensional (x,y,z) so we have to calculate cov(x, y), cov(x,z), and cov(y,z). Covariance matrix is used for the data with 3-dimensional and above. Since we have 6-D data sets for the position and orientation data, we should have 6×6 covariance matrix.

Calculating the eigenvectors and eigenvalues. The eigenvector is non-zero vector which its direction isn't affected by applying any linear transformation and can be calculated only for squared matrices. The eigenvectors are scaled to have the same length which means that each element is divided by the length of the vector. Also they are sorted according to the eigenvalues descending. The eigenvector with the highest eigenvalue is the most important and the eigenvector with the least eigenvalue is the least important.

Forming the feature vector. In this step, either all eigenvectors are used for forming the feature vector, or only subset of the eigenvectors, the selected eigenvectors are called principal components (i.e. fV). In the case of selecting all eigenvectors, the new dataset will be of the same dimensions of the original data and there will be no loss of information. However, in the case of selecting only subset of the eigenvectors, there will be loss of information because of reducing the dimensionality of the data.

Calculating the new dataset. The new dataset which is called the scores or here we call it preprocessed data are derived by multiplying the adjusted data with the feature vector as follows:

$$Y = X^{AD} * fV. \tag{1}$$

The preprocessed data are of the same lengths of the raw trajectories and it is used in this paper instead of the raw data to apply the clustering and find optimum position for the initial key points. The preprocessed data is named Y and it will be used in identifying the initial key points instead of X.

2.2 Selection and Modeling of the Key Points

The initial key points are assigned using the preprocessed data derived by applying the PCA (i.e., Y). The preprocessed data is first clustered using K-means algorithm [10]. After the clustering, each transition among the clusters labels is assigned as an initial key point. Discrete HMM (DHMM) is used for modeling the key points. Set of observation symbols are needed for each trajectory to apply such model. So that the preprocessed data are clustered using the K-means algorithm to map the data into discrete observation symbols. Now we have the trajectories mapped to discrete observation symbols and for each trajectory we have set of key points. The Bakis left-right topology [11] is used to calculate the needed parameters. The initialization of the HMM parameters i.e. (A, B, π) is done using minimum distortion trajectory X_σ which is selected according to the criteria stated in [12]. Number of key points in this minimum distortion trajectory plus one is used as number of the hidden states, and number of the observation symbols is used as Q. After initializing the parameters, DHMM is employed to model the observation sequences. For each observation sequence, the Viterbi algorithm is applied to achieve the most likely sequence of hidden states. The new sequences, which resulting from applying the Viterbi algorithm, were used to modify the set of the key points. The new values of the key points are the positions and orientations from the trajectory that are equivalent to transitions between the states in the new sequences.

However the key points are precisely modeled, they are corresponding to different time indices as they are belonging to different trajectories. Therefore, the Dynamic Time Warping (DTW) is applied to handle such issue.

2.3 Dynamic Time Warping (DTW)

As mentioned in [11] a reference sequence is required for applying the DTW algorithm so that the sequence with highest log-likelihood is selected to be the reference sequence. The average duration of the hidden states among all sequences is used to adjust the reference sequence, such that it is referred to the average as $\overline{\tau_i}$ for $i = 1, \cdots, N_s$. The updated time indices are put such that $t_{K_1} = 1$ and j$^{\text{th}}$ key point time index is put as mentioned in [5]. After selecting and updating the reference sequence, this sequence is used to align the other sequences using DTW algorithm. In order to have sequence of the same length, shape preserving constraint is used such that the new length is put as the average of the sequences lengths. After Applying DTW, the time indices of the key points are modified to be from the warped time sequences.

2.4 Weighting Coefficients and Interpolation of the Key Points

Since there are different variances among the same trajectory because the begging and the leaving parts of the trajectory have greater variance than the middle part. So there is a need to have weighting coefficients for each key point. The weighting coefficients are set by measuring the root mean square error in order to consider the closeness of each set of the key points. The RMSE and the weighting coefficients are computed as mentioned in [9]. After aligning the key points and assigning the weighting coefficients, a generalized trajectory has to be produced. Cubic spline interpolation [13] is applied to interpolate the set of the key points with the assigned weights to generate the new trajectory.

3 Experimental Work

With the purpose of indicating the efficiency of our proposed scheme, we have used the data set presented by Vakanski et al. [5] in our experimental work. The aim of the demonstrator was to draw a panel virtually in a dry form, by using an operator to move the hand tool which paints the panel, in which the tool was used as a spray gun. The optical marker attached to the hand tool is tracked using the optical tracking system Optotrak Preseon. The data set was created by recording the pose of the tool with respect to the reference frame of the panel. Set of demonstrated trajectories were resulted from conducting two experiments with different complexity. A simple trajectory motion was recorded in the first experiment. A complex motion is recorded in the second experiment, where different amplitudes were used in the waving motions.

In the first experiment, the demonstrated data include twelve trajectories with 6-dimensional measurements for the positions (X, Y, Z) and orientation (Roll, Pitch, Yaw). Missed fields representing NANs points were handled by interpolating the whole trajectory to get values for these missed points. Afterwards the data were preprocessed by applying PCA. PCA has been applied on each trajectory as mentioned in Sect. 2.1 through adjusting the data, calculating the covariance matrix, calculating the eigenvectors and eigenvalues, forming the feature vector by choosing the principal components (i.e. we have chosen all the eigenvectors as principal components), then reconstruct the data by multiplying the feature vector and the adjusted data. After applying the PCA, the preprocessed data has been clustered using K-means algorithm by 64 clusters to identify the initial key points. The actual data representing the position and orientation of the key points were determined by using only the index from the PCA domain but the values were put from the original domain. That's because we use all the six eigenvectors so there is no loss of the information.

Afterwards, the set of key points have been modeled by employing the DHMM. To apply the DHMM, the data were mapped into discrete observation symbols by clustering it using 256 clusters as number of the observation symbols. The parameters of the HMM was set using the minimum distortion trajectory which was X_{12}. Number of the key points in the minimum distortion plus one for the first state was used as the number of the hidden states and Q was set to number of the observation symbols which was 256. After training the parameters of the HMM, the Viterbi algorithm was employed for each observation sequence to get the most likely sequence of hidden states. Then the key points were altered to be at the index each transition included between the hidden states in the new sequences resulting from the Viterbi algorithm. The position and orientation of the key points were set from the trajectory that equivalent to the new indices. The key points were aligned in the time domain using DTW algorithm. The sequence which had the greatest log likelihood was O_3 and it was selected as the reference sequence. The average duration of the key points among all trajectories were calculated to edit the reference time sequence. This reference sequence was used to align the other time sequences. The time indices of the key points were updated to be from the aligned sequences. Then each key point had a weighting coefficient according to the RMSE values. Finally, the set of the aligned key points with the weighting coefficients were interpolated using cubic spline interpolation by smoothing factor 0.975 and a length equal to the average of the trajectories lengths. The generalized trajectory of this experiment is shown in Fig. 2.

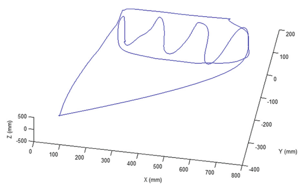

Fig. 2. The x-y-z coordinates for the generalized trajectory of the first experiment.

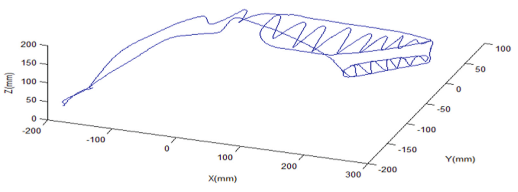

Fig. 3. The generalized trajectory of the second experiment.

In the second experiment, the panel was painted in a complex geometry. The task was performed five times by only one demonstrator resulting in five demonstrated trajectories (i.e. $X_1, \ldots, X_5$). We have applied the same steps of the first experiment with the same parameters. The generalized trajectory is shown in Fig. 3.

4 Experimental Results and Comparison

For the purpose of evaluating our proposed scheme's performance, our results were compared to the previous work using two metrics. First, the accuracy of the generalized trajectory is measured by the RMSE metric [14]. The second metric is the computational cost which is measured in terms of processing times. In our comparison, we have compared our proposed scheme's performance with the work presented in [5], because the authors have already mentioned that their results were better than the results obtained from using the state-of-the-art approaches presented in [3, 4] in terms of the RMSE and the computational cost. RMSE have been used because its ability to show the similarity between the trajectories. Minimum RMSE means that the selected trajectory is the most similar one with respect to the other trajectories, so it ought to be the best one that the robot should follow. For applying the RMSE, the trajectories were first scaled using linear scaling to have the same length. The new length was equals to the average of all trajectories lengths.

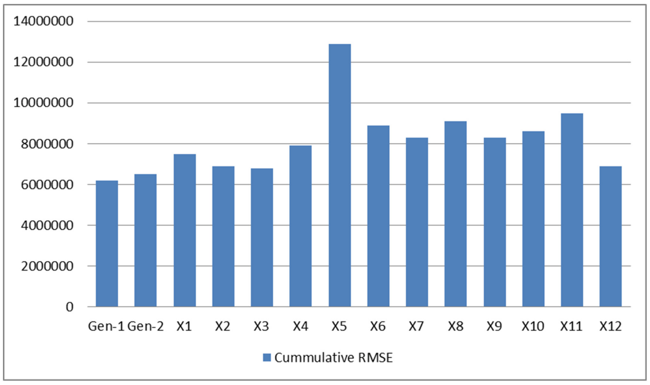

Fig. 4. The cumulative sum of the RMSE for the first experiment.

In the first experiment, Fig. 4 shows the cumulative sum of the RMSE. The cumulative sum of the generalized trajectory resulted from our approach (i.e. Gen-1), the generalized trajectory resulted using the approach presented in [5] (i.e. Gen-2), and the original trajectories (i.e. $X_1, \ldots, X_{12}$) are shown in the figure. The cumulative sum of the RMSE at Gen-1 means the sum of the RMSE between Gen-1 and the original trajectories (i.e. $X_1, \ldots, X_{12}$), the cumulative sum at X_1 is the sum of the RMSE between X_1 and $(X_2, \ldots, X_{12})$, while the cumulative sum at Gen-2 is the value reported in [5]. The figure shows that our proposed scheme's performance has the minimum cumulative RMSE with respect to the previous work and the original trajectories. We had the minimum error because instead of identifying the initial key points from the raw data, we first applied the preprocessing step using PCA algorithm which leads to enhancement in the index of the initial key points. Such enhancement was expected because the PCA analyzes the data and consider for the most important variation included in the raw data. Thus, this proposed scheme would lead to generating a generalized trajectory which is more accurate.

Similarly, in the second experiment, Fig. 5 shows the cumulative sum of the RMSE. The figure shows that the generalized trajectory resulted from applying our proposed scheme (i.e. Gen-1) has the minimum error with respect to the generalized trajectory obtained in [5] and the original trajectories (i.e. X_1, …, X_5).

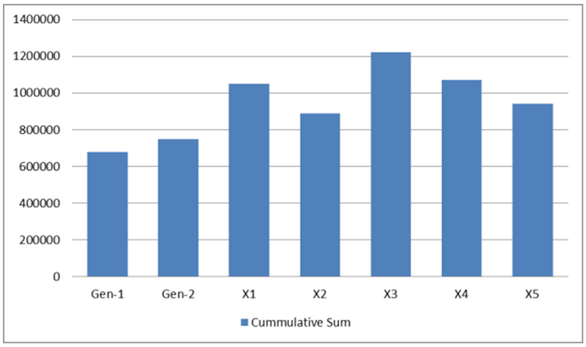

Fig. 5. The cumulative sum of the RMSE for the second experiment.

For having a complete comparison and discussion, the MATLAB has been used for implementing our model. The codes were run ten times on 1.8 GHz INTEL core i5 CPU with 4 GB RAM on windows 7. MEX file of the DTW has been used to increase DTW speed as indicated in [5]. The mean and standard deviations of the processing times for each module in our proposed scheme for both experiments are shown in Table 1. The experimental work in [5] was performed using 2.1 GHz dual core CPU with 4 GB RAM, the total processing times of the scheme presented in [5] are 404.045 s and 245.726 s, respectively.

Table 1. The mean and standard deviations values of the computation time by our approach in the two experiments.

Code steps:	Time in seconds	
	Using our approach (Experiment-1)	Using our approach (Experiment-2)
1. Preprocessing	11.607(±1.759)	4.274(±0.648)
2. Initial key points	9.434(±2.016)	3.537(±1.048)
3. HMM discretization	29.189(±3.66)	7.274(±1.107)
4. HMM training and inference	19.325(±8.706)	15.556(±13.64)
5. DTW alignment	84.854(±22.954)	21.167(±4.89)
6. Weighting and interpolation	0.789(±0.342)	0.788(±0.209)
Total:	155.198	52.597

The approach presented by Vakanski et al. [5] included the preprocessing step only for applying smoothing and removing of NANs. However, the preprocessing step included in our approach represents removing NANs and applying PCA for initial reconstruction of data. Table 1 shows that our proposed scheme reduces the total computation time, such that the total processing time of our two experiments are 155.198 and 52.597, respectively. However, applying our proposed scheme increases the time of the preprocessing module with respect to the time of the same module in [5], but this increasing in the time affects on the next module of identifying the initial key points which becomes faster than the same module in [5]. The results show that, however the computational cost of the added preprocessing stage is significant, the other remaining modules requires less computational cost in comparison to the state-of-the-art schemes presented in [5] and therefore it requires less total processing time.

5 Conclusion

In this paper, a new scheme for robots trajectory learning has been proposed. The proposed scheme starts with a preprocessing step by employing the PCA algorithm. PCA algorithm has been employed on the raw data to represent the important variation contained in the raw data. The initial key points have been identified after preprocessing the data which resulted in enhancements in the positions of the key points, and as a result it has enhanced the modeling phase and the phase of generating the generalized trajectory. We have used RMSE metric to compare our results with the previous work. The results indicated that the generalized trajectory generated by applying our scheme have the ability to minimize the error with respect to the previous work. The reason for this error minimization is using the preprocessed data in identifying the initial key points rather than using the raw data. In addition, it has been shown also that the overall computational cost is much smaller than the state-of-the-art approaches. We are working on having extra improvement by utilizing another clustering algorithm rather than the K-means algorithm in the phase of identifying the initial key points.

References

1. Argall, B., Chernova, S., Veloso, M., Browning, B.: A survey of learning from demonstration. Robot. Auton. Syst. **57**(5), 469–483 (2009)
2. Alizadeh, T., Calinon, S., Caldwell, D.G.: Learning from demonstrations with partially observable task parameters. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, pp. 3309–3314 (2014)
3. Calinon, S., Billard, A.: Stochastic gesture production and recognition model for a humanoid robot. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, pp. 2769–2774 (2004)
4. Asfour, T., Gyarfas, F., Azad, P., Dillmann, R.: Imitation learning of dual-arm manipulation tasks in a humanoid robot. In: Proceedings of the 6th IEEE RAS International Conference on Human. Robots, Genoa, Italy, pp. 40–47 (2006)
5. Vakanski, A., Mantegh, I., Irish, A., Janabi-Sharifi, F.: Trajectory learning for robot programming by demonstration using Hidden Markov Model and Temporal Dynamic Warping. Robot. Auton. Syst. **42**(4), 1039–1052 (2012)

6. Field, M., Stirling, D., Pan, Z., Naghdy, F.: Learning trajectories for robot programing by demonstration using a coordinated mixture of factor analyzers. IEEE Trans. Cybern. **46**(3), 706–717 (2015)
7. van den Berg, J., et al.: Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations. In: Proceedings of the 2010 International Conference on Robotics and Automation (ICRA), Anchorage, AK, USA, pp. 2074–2081 (2010)
8. Kormushev, P., Calinon, S., Caldwell, D.G.: Robot motor skill coordination with em-based reinforcement learning. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2010)
9. Smith, L.I.: A tutorial on principal components analysis, February 2002
10. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
11. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE **77**(2), 257–286 (1989)
12. Tso, S.K., Liu, K.P.: Demonstrated trajectory selection by hidden Markov model. In: Proceedings of the IEEE International Conference on Robotics and Automation, Albuquerque, NM, pp. 2713–2718 (1997)
13. Rice, J., Rosenblatt, M.: Smoothing splines: regression, derivatives and deconvolution. Ann. Stat. **11**(1), 141–156 (1983)
14. Calinon, S., D'halluin, F., Sauser, E.L., Caldwell, D.G., Billard, A.G.: Learning and reproduction of gestures by imitation: an approach based on hidden Markov model and Gaussian mixture regression. IEEE Robot. Autom. Mag. **17**(2), 44–54 (2010)