Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full length article

# Trajectory learning using posterior hidden Markov model state distribution

Asmaa A.E. Osman [*], Reda A. El-Khoribi, Mahmoud E. Shoman, M.A. Wahby Shalaby

*Department of Information Technology, Faculty of Computers and Information, Cairo University, Egypt*

A R T I C L E  I N F O

*Article history:*
Received 16 August 2016
Revised 26 November 2016
Accepted 22 December 2016
Available online 11 January 2017

*Keywords:*
Robots
Trajectory learning
Learning from demonstration
Hidden Markov models
Posterior state distribution of HMMs

A B S T R A C T

Many life applications are extremely depending on using the robots, thus the human are seeking to develop efficient robots. Robot learning is to acquire extra knowledge in order to achieve objective configuration. In addition, robot learning from demonstration is about teaching the robot how to do specific task by the guidance of the human. Till now, learning from demonstration depends on discrete data which may cause distortion in the learning outcome. So that, preprocessing phase for the data is necessarily to handle this distortion. In this paper, we propose a new scheme for generating a generalized trajectory by employing set of demonstrated trajectories. Such that preprocessed data is used initially instead of the raw data, the preprocessing is done using posterior hidden Markov model state distribution. The rest of the model is based on set of key points identified for each demonstration. Our proposed scheme is experimentally compared to the previous works. The results show that our proposed scheme is able to reduce the error in comparison to other recent schemes with insignificant added computational cost.

## 1. Introduction

Nowadays, the needs of the human being are highly relying on using robots in many life applications. Accordingly, the need to produce intelligent robots is drastically increasing. Robot learning is considered to be a combination between the robotics and the machine learning fields. The transition from a current state of the robot to another to execute new action (i.e. policy) is the core of robotics applications. The machine learning techniques are traditionally used to develop new polices [1].

The traditional technique for programming the robot is to have a human programmer. The programmer needs to code the required task and make the robot adapted to any new situation. In this technique, the programmer needs to break down the task for many steps then perform testing for each step. If the robot meets unexpected conditions after coding the task and deploying the robot, the coding procedure may need a lot of modifications or to be repeated. In contrast to this traditional learning technique, robot learning from demonstration does not require an expert user, and hence it eliminates the need to go deeply in the technical details [1–3]. Robot programming by demonstration enables the development of new medical applications for using the robots as surgical assistants [4].

Among the learning from demonstration techniques, the commonly used approach is developed such that a robot learns how to do a task using its own sensors instead of interpreting the body of a demonstrator or even another robot. The demonstrator teaches the robot in an iterative manner using multiple demonstrations, thus the robot is enabled to produce a generalized demonstration. If any error found during performing the task, the demonstrator will just provide other demonstrations. The objective is to expand the ability of the robot to handle new conditions and to be adjusted to those conditions.

In learning from demonstration, the mapping between the states and actions should be done from a state experienced to an action already taken. Learning from demonstration is divided into two phases, first collecting the demonstrations then developing new policy according to the collected demonstrations [1].

* Corresponding author.
*E-mail address:* asmaa.a.elsayed@fci-cu.edu.eg (A.A.E. Osman).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Many techniques have been applied to model the demonstrated data in the trajectory level. It has been found in the literature that hidden Markov models (HMMs), Gaussian mixture models (GMMs) and conditional random fields have been applied successfully for modeling the demonstrated data [3,5–9]. Alizadeh et al. [3] employed an approach that focus on the problem that happens when there are some missed parameters for the task reproduction. In their proposed approach, they used the Gaussian mixture model for the trajectory reproduction. Another work was presented by Vakanski et al. [8], the authors employed an approach for trajectory reproduction. In their approach, Conditional Random Field was used for modeling the demonstrations and for finding the most important features that are needed for reproducing a generalized trajectory.

HMM is the most widely used technique compared to the others found in the literature because of its capability to model spatial and temporal variations in the data. Calinon and Billard [6] employed key points approach based on HMM to generate a trajectory. The authors used the forward algorithm for HMM to find a set of observation sequences. Afterwards, the Viterbi algorithm was used to find the most likely sequence of hidden states for the sequence with the highest log likelihood. Third order spline fit was applied to find a generalized trajectory. The drawback of this approach is that they used the key points from only one demonstration. The generalized trajectory may miss some important key points found in the other demonstrations.

Asfour et al. [7] employed a similar HMM-based key point approach for trajectory learning. The authors used the term common key points to refer to the key points that are found in all demonstrations. The Viterbi algorithm was also used to find the most likely sequence of hidden states for each demonstration. Then, linear interpolation was used between the common key points to find a generalized trajectory. The drawback of this approach is that they used only the common key points in the interpolation phase. Thus, it would eliminate an important key point because it was missed in one of the demonstrations.

Vakanski et al. [9] employed an approach such that the key points from all demonstrations are used in the trajectory reproduction. The authors used the Linde–Buzo–Gray (LBG) algorithm for the selection of key points. Afterwards, Discrete Hidden Markov model was used to model the set of key points. Dynamic time warping (DTW) was applied to align the key points in time. Weighting coefficients are assigned to each set of key points to consider the variance of the key points among the demonstrated data. Finally, Cubic spline interpolation was used to generate a generalized trajectory. The drawback of their approach is that they used discrete HMM (DHMM) to model the demonstrated trajectories and this would lead to local distortion in the demonstrated data. Hence, there is a need to preprocess the trajectory raw data to overcome such a distortion problem.

Calinon and Billard [10] presented an approach for gesture recognition and reproduction. Their approach is based on two main stages: (1) Using Principal component analysis (PCA) or Independent component analysis (ICA) for the decomposition of the data as a preprocessing stage, (2) Applying hidden Markov models (HMMs) for the encoding of the gestures. Another approach used for the dimensionality reduction is the factor analyzers [11]. The authors proposed an approach for trajectory learning from demonstration based on using probabilistic motion primitives for representing the demonstrated trajectories. In this approach, the motion was recorded by the body of the human then correspondence map between the human and the robot was achieved. This approach can be used in many applications such as surgical applications [4]. Another approach for learning from demonstration, which avoids interpreting the body of the human to the kinematics of the robot, is called Kinesthetic teaching, in which the motion is recorded using the body of the robot [9]. This approach is especially important on highly-dynamic tasks [12]. The Kinesthetic teaching approach is employed in our proposed scheme.

It is seen from the literature that the learning from demonstration depends mainly on discrete data which may cause distortion in the learning outcome. Thus, a preprocessing phase for the data is necessarily to handle this kind of distortion. In this paper, a new scheme for robot trajectory learning is developed by employing the posterior HMM state distribution [13] as a preprocessing step. Posterior HMM state distribution is used in reconstructing the trajectory data using set of hidden states. The reconstructed data is used instead of the raw data to identify the positions of the initial key. And hence, this would lead to enhancement in identifying the positions of the initial key points. After identifying the initial key points, HMM is used to model the set of key points. Afterwards, DTW is used for the temporal aligning of the key points. Weighting coefficients are assigned such that the low variance parts can have higher weights than the high variance parts. Finally, the last step deals with interpolating the set of aligned key points and its weights using cubic spline interpolation to find a generalized trajectory.

The rest of the paper is organized as follows: data preprocessing phase is presented in Section 2, in Section 3 the initial key points selection is discussed, modeling of the key points is addressed in Section 4, trajectory generalization is presented in Section 5, the experimental work is shown in Section 6 and finally the results, comparisons and discussions are reported in Section 7.

## 2. Posterior HMM state distribution

The input for the proposed scheme consists of the raw demonstrated trajectories and the output of this stage is the preprocessed trajectories. Two steps are performed within this stage: (1) the missed fields consisting of NaNs are interpolated from the rest of the data; (2) after interpolating the NaNs fields, the interpolated data are then reconstructed using posterior HMM state distribution.

### 2.1. Data gathering

The work presented in this paper is mainly based on data got from teaching the robot how to do specific task. The demonstrator teaches the robot by moving the robot to do the required task so that the robot will not need to interpret the movement of the teacher or even another robot. The data contains set of M demonstrated trajectories, each trajectory $X_m$ contains 6-D measurements for the position and Euler's roll–pitch–yaw angles for the tool's orientation data where $m \in \{1, 2, \ldots, M\}$. We proposed a model for generating a generalized trajectory from a set of key points which define the most important features in each of the demonstrated trajectories.

A block diagram of the proposed scheme is shown in Fig. 1. We have used the posterior HMM state distribution to interpolate the data initially using set of states defined in each trajectory.

### 2.2. HMM parameters initialization

A set of hidden states and observation symbols are needed to employ the posterior HMM state distribution. Thus, the normalized positions and velocities [9] of each trajectory are combined and clustered using K-means algorithm. The clustering phase is used twice: first to represent each trajectory in a set of observation symbols and second for having set of hidden states. The combined data are mapped into set of discrete observation symbols (i.e. $O_m$) and into clusters labels (i.e. $C_m$) too. Afterwards, the parameters needed

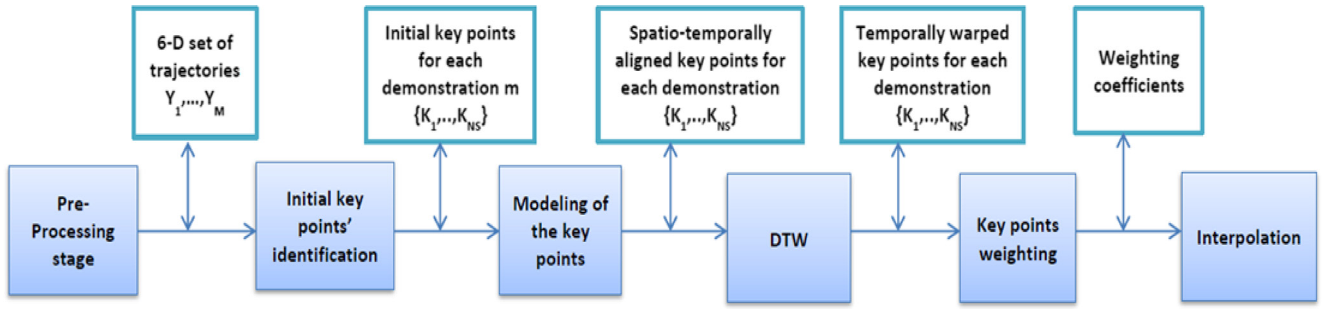
6-D set of trajectories Y1,...,YM
Initial key points for each demonstration m {K1,..,KNS}
Spatio-temporally aligned key points for each demonstration {K1,..,KNS}
Temporally warped key points for each demonstration {K1,..,KNS}
Weighting coefficients
Pre-Processing stage
Initial key points' identification
Modeling of the key points
DTW
Key points weighting
Interpolation

**Fig. 1.** Block diagram of the proposed scheme.

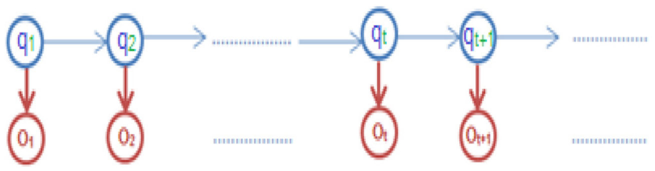

**Fig. 2.** Graphical representation of an HMM.

for applying posterior HMM state distribution are estimated using the clusters labels and the observation symbols. A graphical representation of HMM is shown in Fig. 2, where $q_t \in \{s_1, s_2, \ldots, s_T\}$ denotes the hidden state and $o_t$ denotes the observation symbol at time index $\{1, 2, \ldots, t, t+1, \ldots\}$. A HMM can be represented as: $\lambda = \{A, B, \pi\}$.

- Transition Probabilities (A)

The probability of the transition from state $i$ at time t to state $j$ at time $t+1$ is denoted by $a_{ij} = P[s_{t+1} = j | s_t = i]$, form the state transition matrix $A = \{a_{ij}\}$ where $i, j = \{1, 2, \ldots, N_s\}$, and $N_s$ is the number of states in the model. The state transition diagram is shown in Fig. 3. The state transition matrix is calculated as follows:

$$a_{i,i} = (1 - 1/\tau_i)(1/Z) \tag{1}$$

$$a_{i,i+1} = (1/\tau_i)(1/Z) \tag{2}$$

$$a_{i,i+2} = (1/4\tau_i)(1/Z) \tag{3}$$

where $\tau_i$ is the duration of state $i$ in the demonstration and $Z$ is constant for normalization.

- Observation probabilities (B)

The observation probability of a symbol $q_k$ while the model is at state $i$ is denoted by $b_i(k) = P[q_k \ at \quad t | s_t = i]$, form the observation probability matrix. It is calculated as follows:

$$b_i(k) = n_i(q_k)/\tau_i \tag{4}$$

where $n_i(q_k)$ is how many times the symbol $q_k$ is appeared with the state $i$.

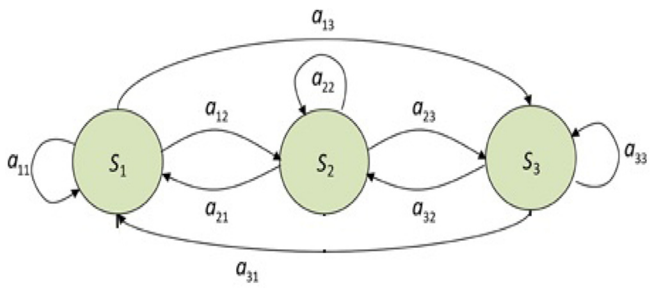

**Fig. 3.** State transition diagram.

- Initial state probabilities ($\pi$)

The probability of being at state $i$ at time $t$ is denoted by $\pi_i$ and it is initialized as $\pi = [10 \ldots 0]$.

### 2.3. Trajectories reconstruction

After identifying the parameters of the HMM, the delta matrix probabilities is calculated "$\delta$". In our proposed scheme, delta matrix is used to reconstruct the trajectory data. The probability $\delta_t(i)$ for any sequence to have the highest probability path for the first T observations is defined for the state $S_i$,

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_t} p(q_1, q_2, .., q_t = i, o_1, o_2, .., o_t | \lambda) \tag{5}$$

$$\delta_t(i) = P(q_1^* q_2^* .. q_t^* = i, o_1 o_2 .. o_t | \lambda) \tag{6}$$

We will consider that:

$$P(q_t^* = i | o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1}) = \frac{P(q_t^* = i, o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1} | \lambda)}{\sum_j P(q_t^* = j, o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1} | \lambda)} \tag{7}$$

$$P(q_t^* = i | o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1}) = \frac{\delta_t(i)}{\sum_j \delta_t(j)} \tag{8}$$

$\delta_t(i)$is calculated as follows:

- Initialization

$$\delta_t(i) = \pi_i b_i(O_1), 1 \leqslant i \leqslant N \tag{9}$$

- Recursion

$$\delta_t(j) = \max_{1 \leqslant i \leqslant N} [\delta_{t-1}(i) a_{ij}] b_j(t), 2 \leqslant t \leqslant T, 1 \leqslant j \leqslant N \tag{10}$$

Numerical example for the calculation of the delta matrix is shown in Table 1. After calculating the delta matrix, the new trajectory data $Y_m$ is calculated as follows:

$$Y_m(t) = E[C_i^m(t) | o_1 o_2 \cdots o_t] = \int_{-\infty}^{\infty} C_i^m(t) P(C_i^m(t) | o_1 o_2 \cdots o_t) d_t \tag{11}$$

This satisfies the minimum mean square error (MMSE) criteria since $C_i^m(t)$ is a function of the state, we can use $P(q_t^* = i | o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1})$.

$$Y_m(t) = \sum_{i=1}^{N_s} C_i^m(t) P(q_t^* = i | o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1}) \tag{12}$$

$$Y_m(t) = \sum_{i=1}^{N_s} C_i^m(t) \frac{\delta_t(i)}{\sum_j \delta_t(j)} \tag{13}$$

Then the reconstructed data $Y_m$ is used instead of $X_m$ to find the position of the initial key points.

**Table 1**
Numerical example for the calculation of the delta matrix.

| |
|---|
| Assuming we have those probability matrices for the transition and observation probabilities, |
| A = [0.95, 0.05; 0.10, 0.90], B = [1/6, 1/6, 1/6, 1/6, 1/6, 1/6; 1/10, 1/10, 1/10, 1/10, 1/10, 1/2;] |
| For the following sequence: |
| [1, 5, 2, 4, 1, 4, 2, 4, 5, 5, 3, 1, 2, 6, 1, 5, 4, 6, 1, 3] |
| The delta matrix is: |
| [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1; 1, 1, 1, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2] |
| After normalization the delta matrix is: |
| [0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185, 0.0185; 0.0185, 0.0185, 0.0185, 0.0370, 0.0370, 0.0370, 0.0370, 0.0370, 0.0185, 0.0370, 0.0370, 0.0370, 0.0370, 0.0370, 0.0185, 0.0185, 0.0370, 0.0370, 0.0370, 0.0370] |

## 3. Initial key points selection

After the reconstruction of the data using posterior HMM state distribution, the new data will be clustered using K-means algorithm [14] and the initial key points will be defined at the transition between the clusters labels in each demonstration.

K-means algorithm starts with initialization of a set of K means $m_1^1, \ldots, m_k^1$ then the algorithm alternates between the following two steps repeatedly until the clusters assignments do not change.

- Assignment step: each point is assigned to the nearest cluster.

$$S_i^{(t)} = \left\{ y_p : \left\| y_p - m_i^{(t)} \right\|^2 \leqslant \left\| y_p - m_j^{(t)} \right\|^2 \forall j, 1 \leqslant j \leqslant k \right\} \tag{14}$$

- Update step: the centroid of each cluster is updated to be the mean of the new cluster points.

$$m_i^{(t+1)} = \frac{1}{\left| S_i^{(t)} \right|} \sum_{y_j \in S_i^{(t)}} y_j \tag{15}$$

## 4. Modeling of the key points

After identifying the initial key points for each demonstration, the reconstructed data are mapped into set of discrete observation symbols. Discrete HMM is used for modeling the demonstrated trajectories. The parameters needed for applying HMM (a,b,π) are initialized using the minimum distortion trajectory and calculated according to Bakis left-right topology [15]. The minimum distortion trajectory $X_\sigma$ is selected using the criteria mentioned in [16] and [17].

$$\sigma = \arg\min_{1 \leqslant m \leqslant M} \frac{\sum_{n=1}^{N_m} (\alpha_{n,m} - \mu(l_{n,m}))}{N_m} \tag{16}$$

Number of hidden states is set equal to number of key points in this minimum distortion trajectory plus 1 and Q is set equal to number of observation symbols. Discrete HMM is trained on each observation sequence using the initialized parameters. The Viterbi algorithm is used to find the most likely sequence of hidden states for each observation sequence. Afterwards, the key points are modified to be at the position and orientation values of trajectory that correspond to transition between the hidden states. Until this step, there are set of key points for each demonstration but they are not temporally aligned. Dynamic time warping algorithm is used to solve this issue.

## 5. Trajectories generalization

### 5.1. Dynamic time warping (DTW)

The reference time sequence is selected by finding the observation sequence with the highest log-likelihood [15]. The time sequence of the reference trajectory is modified using the average durations of the hidden states $\overline{\tau_1}$ for $i = 1, \cdots, N_s$. The new time index of the key points is assigned such that $t_{K_1} = 1$ and the time index of the jth key point is set as

$$t_{k_j} = 1 + \sum_{i=1}^{j-1} \overline{\tau_i} \quad for \quad j = 2, \ldots, N_s \tag{17}$$

This modified time sequence is used to align the rest of the trajectories using the dynamic time warping algorithm. The description of dynamic time warping algorithm to align two time sequences (A of length I) & (B of length J) is done as follows:

- Initial condition: g(1,1) = 2d(1,1).
- Dynamic programming equation:

$$g(i,j) = \min \begin{bmatrix} g(i,j-1) + d(i,j) \\ g(i-1,j-1) + 2d(i,j) \\ g(i-1,j) + d(i,j) \end{bmatrix} \tag{18}$$

- Restriction condition (adjustment window):

$j - r \leqslant i \leqslant j + r$

- Time-normalized distance:

$$D(A,B) = \frac{1}{N} g(I,J) \tag{19}$$

Shape preserving constraint is applied so that all sequences would have a length equal to the average of all sequences lengths [18]. The new time index of each key point is assigned from the warped time sequences.

### 5.2. Weighting coefficients

Weighting coefficients are assigned to the key points because the approaching and departing parts have higher variance than the inner parts of the trajectories [19,20]. The weighting coefficients are measured using the root mean square error to check the nearness of each set of key points. The following equations are used to calculate the RMSE and the weighting coefficients:

$$RMSE(k_j) = \sqrt{\sum_{m=1}^{M} \left( k_{j,m}^{DTW} - \overline{k_j^{DTW}} \right)^2} \tag{20}$$

$$w_j = \left\{ \begin{array}{l} 0, \ for \quad RMSE(k_j) \geqslant \varepsilon_{\max} \\ \frac{\varepsilon_{\max} - RMSE(k_j)}{\varepsilon_{\max} - \varepsilon_{\min}}, \ for \quad \varepsilon_{\min} \leqslant RMSE(k_j) \leqslant \varepsilon_{\max} \\ 1, for \quad RMSE(k_j) \leqslant \varepsilon_{\min} \end{array} \right\} \tag{21}$$

### 5.3. Interpolation of the key points

After assigning the weights, cubic spline interpolation is used to produce the generalized trajectory [21].

The pseudo-code algorithm of the proposed model is shown in Table 2.

**Table 2**
Pseudo-code algorithm of the proposed model.

```
Begin
*Preprocessing*
For (# of trajectories)
  1st clustering using K-means Eqs. (14) and (15)(14) and (15)(14) and (15): output clusters labels
  2nd clustering using K-means Eqs. (14) and (15)(14) and (15)(14) and (15): output observation symbols
  Compute HMM parameters:
    Compute "a" matrix using Eqs. (1)–(3)(1)–(3)(1)–(3)(1)–(3)
    Compute "b" matrix using Eq. (4)(4)
    Set $\pi = [10 \ldots 0]$.
  Calculate the reconstructed data
    Compute delta matrix using Eqs. (9) and (10)(9) and (10)(9) and (10)
    Compute reconstructed data using Eq. (13)(13)
End_For

*Initial identification of the key points*
For (# of trajectories)
  Clustering for the reconstructed data using K-means Eqs. (14) and (15)(14) and (15)(14) and (15)
End_For

*Modeling of the key points*
The minimum distortion trajectory is selected using Eq. (16)(16)
The reconstructed data is clustered twice using K-means to output the states and observation symbols by Eqs. (14) and (15)(14) and (15)(14) and (15)
Compute HMM parameters as declared above for the minimum distortion trajectory
For (# of trajectories)
  Apply Viterbi algorithm to find the most likely sequence of hidden states
End_For

*Dynamic time warping*
Find the sequence with the highest log-likelihood
Modify the time of the key points using Eq. (17)(17)
Apply dynamic programming Eq. (18)(18)
Update the time index of the key points according to the warped time sequences

*Key points weighting*
For (# of key points)
  Calculate the RMSE of the key point
  Calculate the weighting coefficient of the key point
End_For
*Interpolation*
Apply cubic spline interpolation to get the generalized trajectory

End
```

## 6. Experimental work

In order to show the effectiveness of the proposed scheme, the data set in [9] have been used in our experimental study. An operator was used to move a hand tool for painting the panel virtually where the tool works as a spray gun. The paintings have been applied in a dry form for some practical reasons. An optical tracking system Optotrak Preseon was used to track the optical marker attached on the hand tool shown in Fig. 4b. The data set resulted from recording the pose of the tool with respect to the reference frame of the panel, shown in Fig. 4a. The data set contains group of demonstrated trajectories resulted from conducting two experiments. The first data set represents a simple trajectory case. However, a complex trajectory was followed in the second experiment, such that it includes waving motions of the painting tool with different amplitudes.

### 6.1. First experiment with simple trajectory

In the first experiment, the task of the demonstrator was to draw a panel by starting at initial position, moving to the upper left corner, drawing the contour of the panel clockwise, drawing its inner part by moving left to right, and return back to the initial position. This task was demonstrated four times by four different operators resulting in sixteen demonstrated trajectories. Then, by eliminating the inconsistent trajectories, the resultant data set consisted of twelve trajectories. Each trajectory contained 6-D measurements for the position and orientation data. The positions of demonstrated data were shown in Fig. 4c.

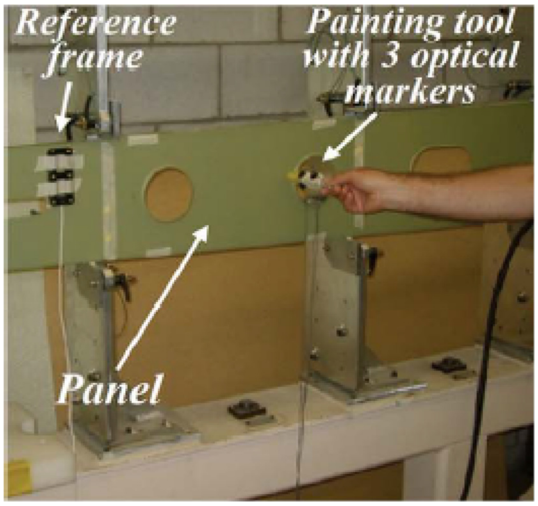

**Fig. 4a.** Experimental setup at NRC-IAR's laboratories for the experiment presented in [9].

The missed fields consisting of NaNs were interpolated from the rest of the data. The posterior HMM state distribution was applied to interpolate each trajectory from a set of hidden states. As mentioned in Section 2.3, to employ the posterior HMM state distribution, the normalized positions and velocities of the raw data are clustered using K-means algorithm twice: first with 256 observation symbols and second with 64 clusters as the number of the

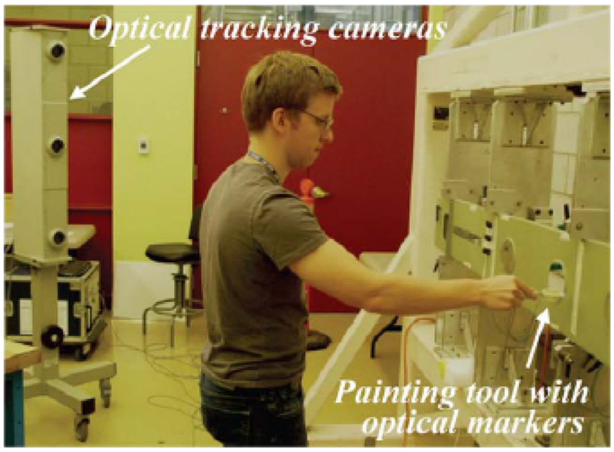

**Fig. 4b.** Perception of the demonstrations with the optical tracking system.

states. Each trajectory $X_m$ was mapped into the observation symbols (i.e. $O_m$) and into clusters labels (i.e. $C_m$) where m = {1,2,...,12}. Afterwards, the parameters (A,B,π) are estimated using the clusters labels and the observation symbols as explained in Section 2.2. The delta matrix "δ" is calculated for each demonstration as explained in Section 2.3 and then the data are reconstructed using Eq. (13). After reconstructing the data, the reconstructed data were used to find the initial key points. The new data were clustered using K-means algorithm with 64 clusters then the initial key points were assigned at the transitions between the clusters labels. The initial selection of key points in the minimum distortion trajectory is shown in Fig. 5.

After identifying the initial key points for each demonstration, the recorded data were mapped into set of discrete observation symbols then the discrete HMM was used for modeling the demonstrated trajectories. Number of observation symbols was set equal to 256 exactly as used in [9]. The parameters needed for applying HMM (a,b,π) were initialized using the minimum distortion trajectory and were calculated according to forward algorithm [15]. The minimum distortion trajectory was selected using the criteria mentioned in [16] and [17]. The selected trajectory using the criteria was $X_{12}$.

Number of hidden states was set equal to number of key points in this minimum distortion trajectory plus 1, and Q was set equal to number of observation symbols. Discrete HMM was trained on each observation sequence using the initialized parameters. Then, the Viterbi algorithm was used to find the most likely sequence of hidden states for each observation sequence. Afterwards, the key points were modified to be at the position and orientation values of trajectory that corresponds to transition between the hidden states.

Dynamic time warping algorithm was used to warp the key points in time. $O_3$ was used as reference because it had the highest log likelihood. The time sequence of the reference trajectory was modified using the average durations of the hidden states. Then this modified time sequence was used to align the rest of the trajectories. The new time index of each key point was assigned from the warped time sequences. Afterwards, weighting coefficients were assigned to the key points as explained in Section 5.2. After assigning the weights, cubic spline interpolation was used to produce the generalized trajectory using smoothing factor equals to 0.975. The length of the generalized trajectory was set equal to the average of all lengths of the trajectories. The generalized trajectory obtained from using our approach is shown in Fig. 6.

### 6.2. Second Experiment with complex trajectory

In the second experiment, a complex geometry was used for painting the panel. The task of the demonstrator was to paint the top and right side of the panel shown in Fig. 7. The reference coordinate system of the panel was defined using the three optical markers shown in the figure. One demonstrator performed the task five times and the demonstrated trajectories (i.e. X1, ..., X5) are shown in Fig. 8. The range of the trajectories lengths was between 4028 and 4266. The same steps applied for the first experiment have been applied in this experiment. The raw trajectories were preprocessed using the posterior HMM state distribution. Number of observation symbols was set to 256 and number of the clusters was 64. The initial key points were selected from the reconstructed data by applying K-means algorithm with 64 clusters. The transitions between the clusters labels were assigned as the initial key points. HMM was used for modeling the set of the observation symbols. DTW was applied to align the demonstrations with the length of 4141 which is the average of the lengths. Weighting

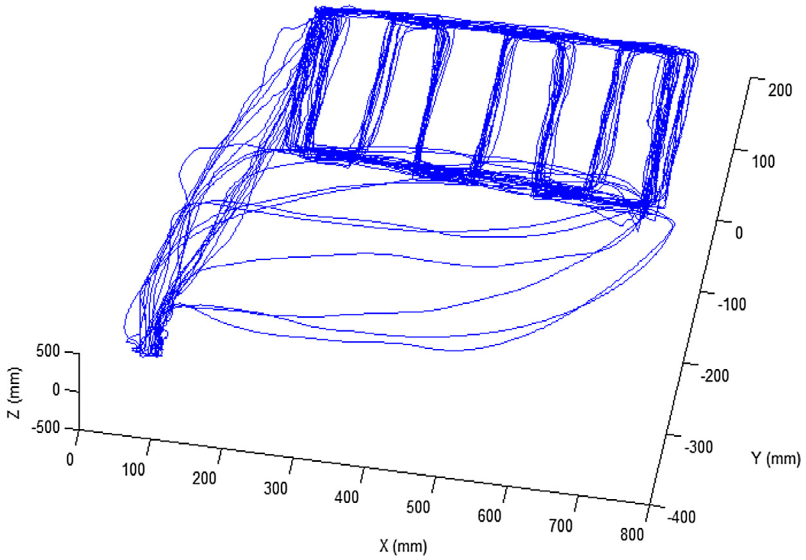

**Fig. 4c.** The x-y-z position coordinates for the demonstrated trajectories in the first experiment.

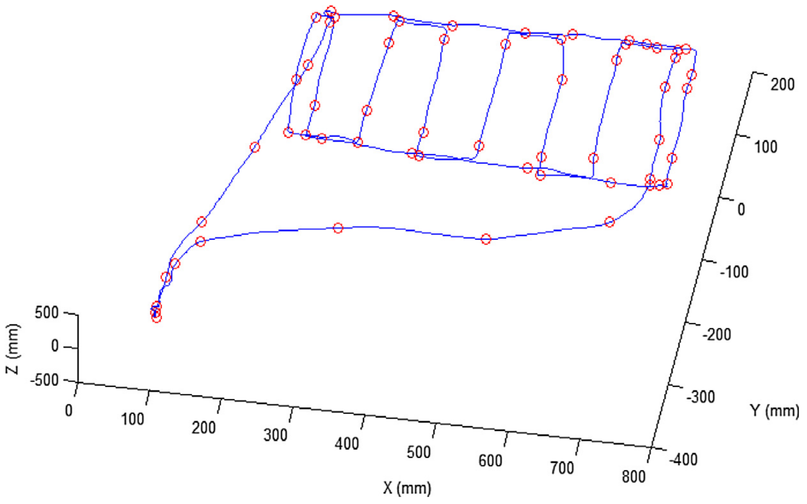

**Fig. 5.** Initial selection of key points in the minimum distortion trajectory.

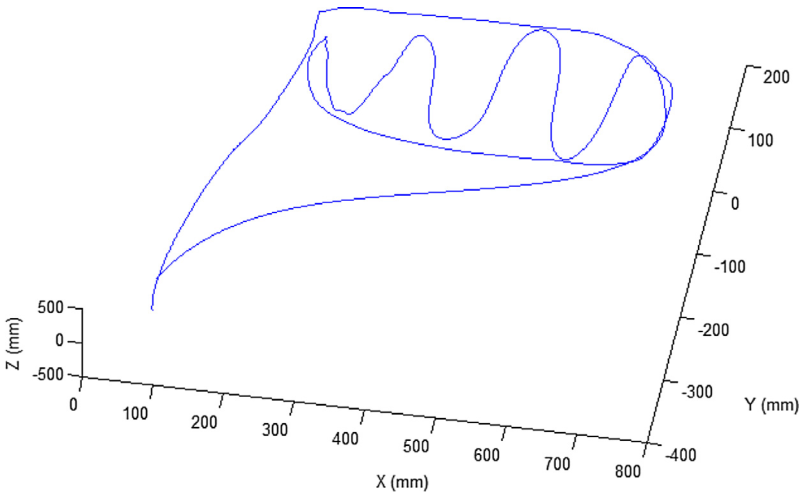

**Fig. 6.** The generalized trajectory for x-y-z position coordinates in experiment-1.

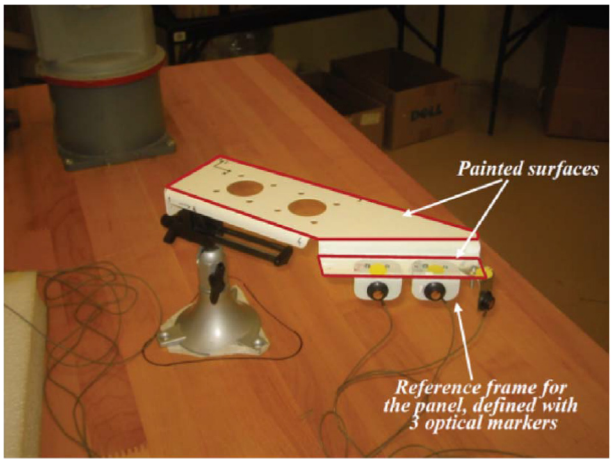
Painted surfaces
Reference frame for the panel, defined with 3 optical markers

**Fig. 7.** The panel used in the second experiment.

coefficients were assigned to the set of the key points. The generalized trajectory shown in Fig. 9 was generated by interpolating the key points from the five trajectories.

## 7. Experimental results and comparisons

In this section, two metrics are used to present the experimental results and to compare the proposed scheme's performance to the previous work. First, RMSE metric [22] is used to measure the accuracy of the generalized trajectory. Second, the computational cost of different learning schemes are measured in terms of processing times. In this comparative study, the proposed scheme's performance is compared only to the approach presented by Vakanski et al. [9]. It has been mentioned in their paper that the RMSE obtained using their approach was less than the RMSEs obtained using the two state-of-the-art approaches presented in [6] and [7]. In addition, it has been shown that the scheme in [9] reduces also the overall computational cost in comparison to state-of-the-art approach presented in [6].

The reason for using the RMSE metric is that it measures the similarity between each two trajectories. Having minimum error means that the selected trajectory is the most similar one to all demonstrated trajectories and it should be the best one for the robot to follow. Thus, the original trajectories were scaled using linear scaling to demonstrations with the same length which equals the average of the lengths of the trajectories. Afterwards,

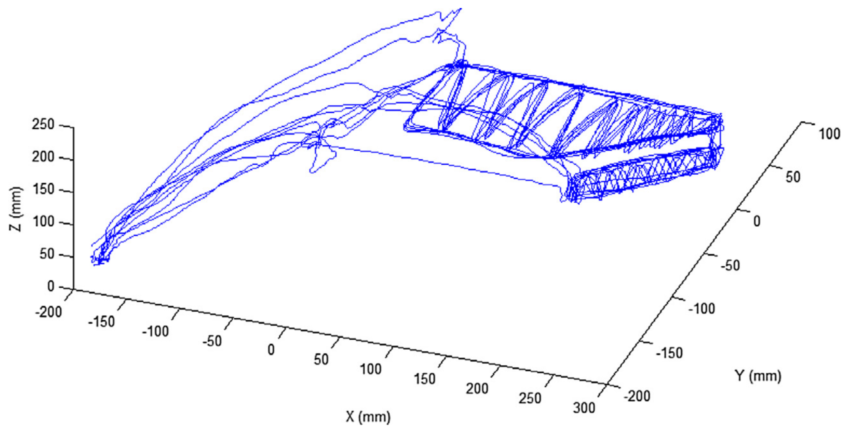

**Fig. 8.** The demonstrated trajectories of the second experiment.

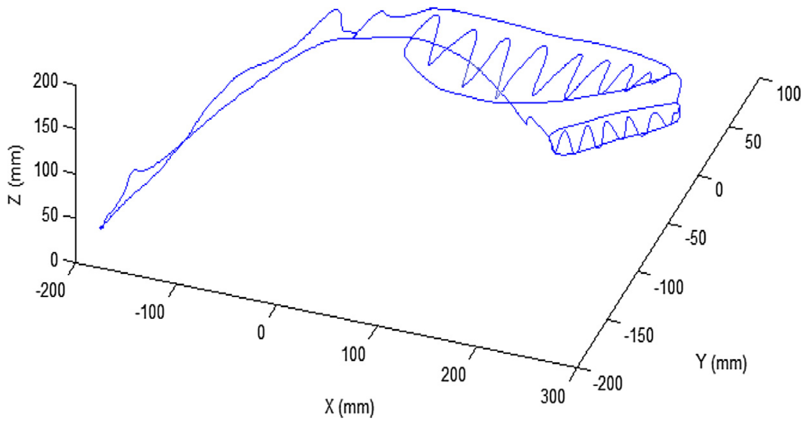

**Fig. 9.** The generalized trajectory of the second experiment.

the root mean square error between two trajectories was measured using the following equation:

$$e(m_1, m_2) = \sum_{t=1}^{N_m} \|X_{m_1,t} - X_{m_2,t}\| \qquad (22)$$

For the first experiment, the cumulative sum of the RMSEs are shown in Fig. 10. Gen-1 is the generalized trajectory obtained using our approach, Gen-2 is the generalized trajectory obtained using the approach presented in [9], and $X_1$ to $X_{12}$ are the original trajectories. 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 ($X_2, \ldots, X_{12}$) and so on. The cumulative sum value at Gen-2 is the value reported in [9]. It is seen that "Gen-1" which is the generalized trajectory obtained using our approach has the minimum error in comparison to others. The reason for having minimum error is that instead of clustering the normalized position

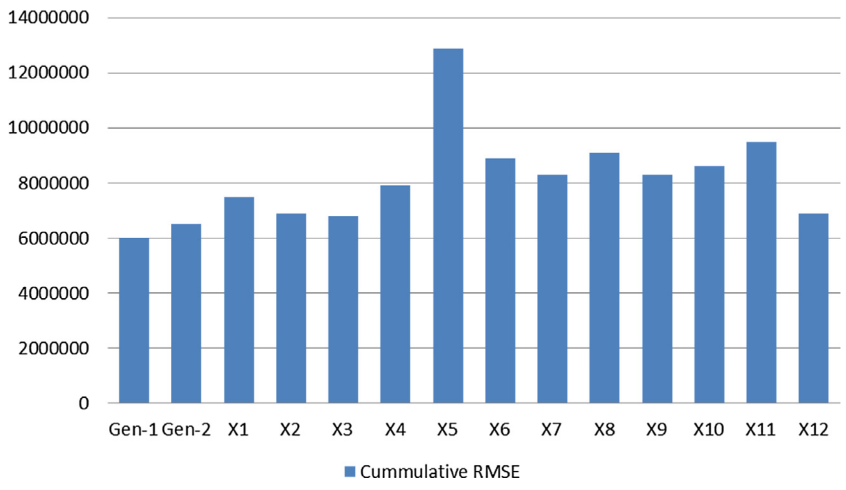

**Fig. 10.** The cumulative sum of RMSE for the generalized trajectory (Gen-1) obtained by using our approach in the first experiment and the demonstrated trajectories (X1-X12).

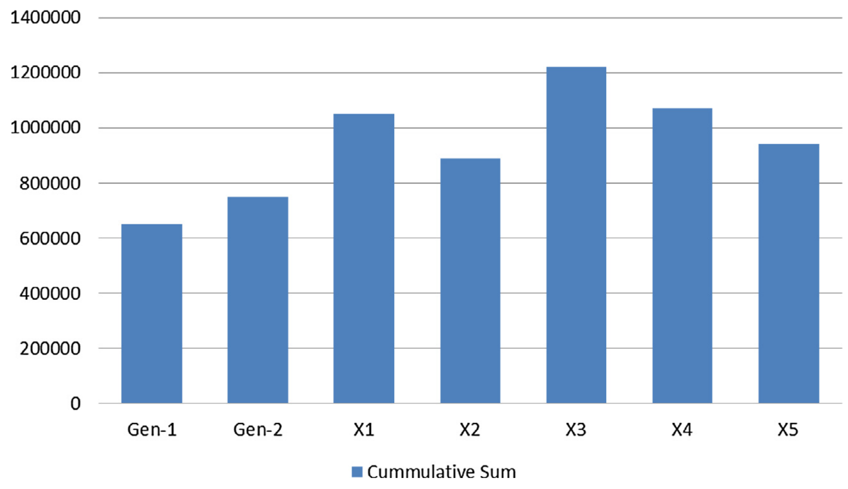

**Fig. 11.** The cumulative sum of RMSE for the generalized trajectory (Gen-1) obtained by using our approach in the second experiment and the demonstrated trajectories (X1-X5).

**Table 3**
The mean and standard deviations values of the computation time.

| Code steps | Time in seconds | |
|---|---|---|
| | Experiment-A | Experiment-B |
| 1. Preprocessing: Posterior HMM state distribution | 133.548(±33.728) | 131.254(±7.623) |
| 2. Initial key points | 2.707(±0.737) | 1.096(±0.218) |
| 3. HMM discretization | 3.272(±0.924) | 1.839(±0.361) |
| 4. HMM training and inference | 11.329(±4.68) | 9.498(±1.171) |
| 5. DTW alignment | 81.709(±19.377) | 24.92(±1.775) |
| 6. Weighting and interpolation | 0.772(±0.275) | 0.761(±0.204) |
| Total: | 233.337 | 169.37 |

and velocity of the raw data to identify the initial key points as done in [9], Posterior HMM state distribution was first used to reconstruct the data to avoid the local distortion in the demonstrated data in the proposed scheme. The new dataset were reconstructed using set of clusters and probability of each cluster at each point. The set of clusters consider the most significant changes in the demonstrated data which would lead to significant identification of the initial key points. This reconstruction helped in finding an optimum index for the initial key points, and hence, the generalized trajectory of the proposed scheme is more accurate.

Similarly, in the second complex-based geometry experiment, the cumulative sum of the RMSEs are shown in Fig. 11. Gen-1 which is the generalized trajectory for experiment-2 using our approach has the minimum error with respect to Gen-2 and the original five trajectories.

In order to measure the computational cost of the proposed scheme accurately, we run the codes ten times using 1.8 GHz INTEL core i5 CPU with 4 GB RAM on Windows 7 and using MATLAB. The MEX files are also used for the DTW alignment as indicated in [9] to increase its speed. Table 3 contains the average and standard deviation of the processing times needed for each module of our proposed scheme for both experiments. From this table, it is clearly seen that the preprocessing step (i.e., Posterior HMM state distribution) and DTW alignment phase represent almost 92% of the overall computational cost in both experiments. Meanwhile, the preprocessing step, individually, represents almost 57% in the first experiment and up to 77% in the second one from the total computational cost.

In comparison to scheme presented in [9], the preprocessing step includes only smoothing and removal of NANs. On the other hand, the preprocessing step of the proposed approach includes removing NANs and applying Posterior HMM state distribution for the initial reconstruction of the data. Since the experimental work done in [9] was reported using 2.1 GHz dual core CPU with 4 GB RAM, the total processing times of the presented scheme in [9] are rescaled to correspond their results to the 1.8 GHz (Quad core) i5 CPU with 4 GB RAM of our proposed scheme. Therefore, the total processing times of the scheme in [9], for both experiments, are found to be almost 235.6 s and 143.3 s, respectively. It is seen from Table 3 that the total processing times of the proposed scheme for both experiments are 233.33 s and 169.37 s, respectively. Hence, it can be seen that the proposed approach requires almost the same computational cost in the first experiment, which contains larger number of trajectories. It can also be seen that in second experiment, the proposed approach requires insignificant increase in the overall computational cost when the number of trajectories are smaller. These 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 [9].

## 8. Conclusion

In this paper, we have presented a new scheme for robots trajectory learning from demonstration. A preprocessing stage has been proposed to avoid any distortion that may happen due to using the raw discrete data directly. The preprocessed data considers for the most significant points in each demonstration and so it is better than the raw data in identifying the initial key points. First, the preprocessed data has been used instead of the raw data. Then, a set of key points were identified for each demonstration and HMM was used to encode the key points. The key points were warped in time domain then interpolated using cubic spline interpolation. RMSE has been used as a metric to compare the generalized trajectory using our approach to the others in the related work. The results showed that our approach has the capability to achieve the minimum error in comparison with the previous work. The reason for having minimum error is that the preprocessed data has been used instead of the raw data. In addition, it has been shown also that the overall computational cost is almost similar to the state-of-the-art approaches. Currently, we are working on further enhancement of the proposed scheme by employing other clustering techniques instead of the K-means in identifying the key points.

## References

[1] Argall B, Chernova S, Veloso M, Browning B. A survey of learning from demonstration. Robot Auton Syst 2009;57(5):469–83.
[2] Billard A, Calinon S, Dillmann R, Schaal S. Robot programming by demonstration. In: Handbook of robotics. New York, NY, USA: Springer; 2008. ch. 59.
[3] Alizadeh T, Calinon S, Caldwell DG. Learning from demonstrations with partially observable task parameters. In: Proc of the IEEE intl conf on robotics and automation (ICRA), Hong Kong, China. p. 3309–14.
[4] van den Berg J et al. Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations. In: Proc 2010 int conf robot autom (ICRA), Anchorage, AK, USA. p. 2074–81.
[5] Aleotti J, Caselli S. Robust trajectory learning and approximation for robot programming by demonstration. Robot Auton Syst 2006;54(5):409–13.
[6] Calinon S, Billard A. Stochastic gesture production and recognition model for a humanoid robot. In: Proc IEEE/RSJ int conf intell robots syst, Sendai, Japan. p. 2769–74.
[7] Asfour T, Gyarfas F, Azad P, Dillmann R. Imitation learning of dual-arm manipulation tasks in a humanoid robot. In: Proc 6th IEEERAS int conf human robots, Genoa, Italy. p. 40–7.
[8] Vakanski A, Janabi-Sharifi F, Mantegh I, Irish A. Trajectory learning based on conditional random field for robot programming by demonstration. In: Proc IASTED int conf robot appl, Cambridge, MA; 2010. p. 401–8.
[9] 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 2012;42(4):1039–52.
[10] Calinon S, Billard A. Recognition and reproduction of gestures using a probabilistic framework combining pca, ica and hmm. In: ICML '05: proceedings of the 22nd international conference on machine learning. ACM; 2005. p. 105–12.
[11] Field M, Stirling D, Pan Z, Naghdy F. Learning trajectories for robot programing by demonstration using a coordinated mixture of factor analyzers. IEEE Trans Cybernetics 2015.
[12] Kormushev P, Calinon S, Caldwell DG. Robot motor skill coordination with em-based reinforcement learning. In: Proc IEEE/RSJ intl conf on intelligent robots and systems (IROS).
[13] El-Khoribi Reda A, Hamza Haitham S, Hammad MA. Indoor localization and tracking using posterior state distribution of hidden markov model. In: 8th International conference on communications and networking, China. p. 557–62.
[14] MacQueen J. Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability 1967;1:281–97.
[15] Rabiner L. A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 1989;77(2):257–86.
[16] Tso SK, Liu KP. Demonstrated trajectory selection by hidden Markov model. In: Proc IEEE int conf robot autom, Albuquerque, NM. p. 2713–8.
[17] Linde Y, Buzo A, Gray RM. An algorithm for vector quantizer design. IEEE Trans Commun 1980;COM-28(1):84–95.
[18] Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust, Speech, Signal Process 1978;ASSP-26 (1):43–9.
[19] Calinon S, Billard A. Learning of gestures by imitation in a humanoid robot. In: Dautenhahn K, Nehaniv CL, editors. Imitation and social learning in robots, humans and animals: social and communicative dimension. Cambridge, U. K.: Cambridge Univ. Press; 2007. ch. 8.
[20] Billard A, Epars Y, Calinon S, Schaal S, Cheng G. Discovering optimal imitation strategies. Robot Auton Syst Jun. 2004;47(2/3):69–77.
[21] Rice J, Rosenblatt M. Smoothing splines: regression, derivatives and deconvolution. Ann Stat 1983;11(1):141–56.
[22] Calinon S, D'halluin F, Sauser EL, Caldwell DG, Billard AG. Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression. IEEE Robot Autom Mag 2010;17 (2):44–54.