Trajectory Learning for Robot Programming by Demonstration Using Hidden Markov Model and Dynamic Time Warping

Aleksandar Vakanski, Iraj Mantegh, Andrew Irish, and Farrokh Janabi-Sharifi, *Senior Member, IEEE*

***Abstract*—The main objective of this paper is to develop an efficient method for learning and reproduction of complex trajectories for robot programming by demonstration. Encoding of the demonstrated trajectories is performed with hidden Markov model, and generation of a generalized trajectory is achieved by using the concept of key points. Identification of the key points is based on significant changes in position and velocity in the demonstrated trajectories. The resulting sequences of trajectory key points are temporally aligned using the multidimensional dynamic time warping algorithm, and a generalized trajectory is obtained by smoothing spline interpolation of the clustered key points. The principal advantage of our proposed approach is utilization of the trajectory key points from all demonstrations for generation of a generalized trajectory. In addition, variability of the key points' clusters across the demonstrated set is employed for assigning weighting coefficients, resulting in a generalization procedure which accounts for the relevance of reproduction of different parts of the trajectories. The approach is verified experimentally for trajectories with two different levels of complexity.**

***Index Terms*—Hidden Markov model (HMM), key points, programming by demonstration (PbD), robotics, robotics learning.**

I. Introduction

ROBOT programming by demonstration (PbD) refers to transferring skills to robots by providing examples of the required behavior through demonstrations. This programming technique is analogous to the way humans learn new skills, i.e., from demonstrations by a teacher. The main advantage of this approach is in the elimination of the need for extensive technical knowledge in order to program robots [1]–[5]. The intuitive programming style of PbD has the potential to facilitate robotic applications in both the manufacturing industry and the service sector (e.g., office or household environments).

Manuscript received March 30, 2011; revised August 22, 2011 and November 29, 2011; accepted January 11, 2012. Date of publication March 9, 2012; date of current version July 13, 2012. This work was supported in part by the New Initiative Funding Program at the National Research Council Canada's Institute for Aerospace Research and Ontario Partnership for Innovation and Commercialization. This work was also supported through a collaborative research and development Grant from the Natural Sciences and Engineering Research Council of Canada: NSERC (CRDPJ 350266-07). This paper was recommended by Editor E. Santos Jr.

A. Vakanski and F. Janabi-Sharifi are with the Department of Mechanical and Industrial Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada (e-mail: aleksandar.vakanski@ryerson.ca; fsharifi@ryerson.ca).

I. Mantegh and A. Irish are with the Institute for Aerospace Research, National Research Council Canada, Montreal, QC H3T 2B2, Canada (e-mail: iraj.mantegh@cnrc-nrc.gc.ca; Andrew.Irish@cnrc-nrc.gc.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2185694

The focus of this paper is on the problem of trajectory generation for transfer of knowledge between a human demonstrator and a robot. Acquiring skills at a trajectory level is considered here because of the rich information that is contained in the human trajectories. This information includes not only the execution-level kinematics data, such as velocities and accelerations, but also important task knowledge, e.g., task space constraints, obstacle avoidance requirements, position and orientation of the robot end-point for grasping a target object or for approaching a workpiece, etc. [1], [4], [6], [7].

Learning a skill at a trajectory level involves modeling the demonstrated set of trajectories and retrieving a generalized representation of the set suitable for reproduction by a robot learner. For modeling the random variations of the human demonstrated motions, many authors employed probabilistic representations of the recorded data from several repeated examples of the same skill. For instance, hidden Markov models (HMMs) [8], Gaussian mixture models (GMMs) [9], and conditional random fields [10] have been reported in the literature for modeling of demonstrated tasks at a trajectory level [1]–[6], [11]–[13]. Among these methods, HMM has been one of the most widely used in the field of robot PbD for modeling and analysis of human motions due to its robustness to spatio-temporal variations of sequential data [14]–[16].

The work presented here concentrates on the problem of reproduction of human motions using HMM. Among the works in the literature that tackle this problem, the approach reported in [17] used the observation sequence with the maximum likelihood of being generated by the learned model to be reproduced by a robot. However, because of the natural noise content of human movements, this trajectory is generally not suitable for execution by robots. In [18], a large number of hidden state sequences for a trained HMM were synthetically generated, and a generalized trajectory was obtained by averaging over the set of trajectories that corresponded to the output distributions of the model for the given state sequences. A similar averaging approach was used in [19]. The drawback of the averaging approaches is that the smoothness of the obtained generalized trajectory cannot be guaranteed.

Another body of work proposed to generate a trajectory for reproduction of HMM modeled skills via interpolation of *trajectory key points* (herein referred to as "key points") which describe a set of features that are the most important for reproduction [20]. In Calinon and Billard [11], to generate a trajectory for task reproduction, first, the forward algorithm

was used for the HMM to find the observation sequence with the maximum likelihood of the trained model, and then, the most probable sequence of hidden states for that observation sequence was obtained by applying the Viterbi algorithm [8]. The key points were connected by a third-order spline for the Cartesian trajectories and via a cosine fit for the joint angle trajectories. Although this approach generated a smooth trajectory suitable for robot reproduction, the resulting trajectory corresponded to only one observed demonstration. What is preferred instead is a generalized trajectory that will encapsulate the observed information from the entire set of demonstrations. The HMM-based key point approach for trajectory learning was also employed by Asfour *et al.* [21]. The authors introduced the term *common key points*, which refer to the key points that are found in all demonstrations. Thus, the generalized trajectory included the relevant features shared by the entire set of individual demonstrations. Their approach was implemented for emulation of generalized human motions by a dual arm robot with the generalized trajectory obtained by linear interpolation through the common key points.

The proposed approach in this paper is similar with those of [11] and [21] in using key points for describing the important parts of the trajectories for reproduction. However, unlike these two approaches, we utilize a clustering technique for initial identification of candidate key points prior to the HMM modeling. Discrete HMMs are then employed for modeling the demonstrated trajectories, and the key points for each observation sequence are identified at the transitions between the hidden states. We introduced the term *null key points*, which refer to the hidden states for which the transitions do not occur in the individual observation sequences. There is generally a temporal mismatch among the trajectory key points identified in any two different demonstrations. To solve this problem, the demonstrated trajectories are aligned in time domain by applying the multidimensional dynamic time warping (DTW) algorithm [22]. The demonstrated trajectory with the maximum likelihood of being generated by the trained HMM is used as a reference sequence for alignment of the demonstrations. A generalized time-frame vector for the reference trajectory is derived by computation of the average durations of all hidden states over the entire set of demonstrations. After the DTW alignment, weighting coefficients are assigned to each cluster of key points, based on key points' variability across the set of demonstrations. A generalized trajectory is produced by applying smoothing cubic spline interpolation of the key points from all demonstrations. The proposed approach was verified for tasks of painting a panel.

The main contributions of our work are the inclusion of the proposed clustering technique in the PbD process for selection of the key points and the generalization approach which employs features from the entire set of demonstrations. Hence, in comparison with the work of Calinon and Billard [11], our proposed approach uses the key points from all demonstrations for interpolation and extraction of a trajectory for task reproduction. Contrary to the work of Asfour *et al.* [21], the key points that are not repeated in all demonstrations are not eliminated from the process. Instead, by introducing the concept of null key points, all of the key points from the entire set of demonstrations are considered for reproduction. Low weights are assigned for reproduction of the key points that are not found in most of the demonstrations, thus ensuring that only the most consistent features across the set of demonstrations will be reproduced. Moreover, the weighting process of the proposed approach also serves as a mechanism for interpreting the intent of the demonstrator. Namely, for the sections of the trajectories that exhibit a considerably higher variance of the key points across the demonstrated set (when compared to the other sections), lower weights for the reproduction purposes are assigned. This step will ensure that the generated trajectory would follow the corresponding sections of the demonstrations more loosely. Conversely, low variability of the key points translates to a constrained part of the trajectories, leading to higher significance for reproduction and high values of the weighting coefficients, thus resulting in a tighter fit of the extracted key points. The advantage of such generalization approach is that the trajectory for task reproduction fits better with the demonstrated set, which has been verified here by using root-mean-square difference as a comparison metric. Additionally, the DTW alignment of the trajectories is performed after the segmentation of the trajectories with HMM. This procedure allows the temporal alignment to be performed at the key point level, which, on the other hand, maintains the shape of the velocity profile of the demonstrated trajectories.

The rest of this paper is organized as follows. Section II discusses the initial steps of the proposed technique related to selection of key points for the demonstrated trajectories. In Section III, the details of initialization and implementation of HMM are addressed. Retrieving a generalized trajectory from the set of key points using DTW and computation of weighting coefficients for the key points are discussed in Section IV. The simulation and reproduction results for two experiments of painting are reported in Section V, while a comparison with the similar PbD approaches is presented in Section VI. Section VII briefly summarizes the proposed approach and the results.

II. Initial Trajectory Labeling

A. Data Acquisition

The proposed approach focuses on learning the trajectories of a manipulated tool. Depending on the task, trajectories of the demonstrator's hand can be tracked and learned without loss of generality. Perception of the demonstrated trajectories is performed with an optical tracking system which measures the positions of infrared optical markers attached to the tool. Based on the markers' locations with respect to a fixed reference frame, position and orientation (pose) of predefined rigid bodies are inferred over a set of discrete time instants. The measured trajectories from the perception phase are denoted by $\{X_m = (\mathbf{x}_{1,m}, \mathbf{x}_{2,m}, \ldots, \mathbf{x}_{N_m,m})\}_{m=1}^{M}$, where m is used for indexing the demonstrations, M refers to the total number of demonstrations, and N_m denotes the number of measurements of the demonstration X_m. Each measurement is a D-dimensional vector $\mathbf{x}_{n,m} = (x_{n,m}^{(1)}, x_{n,m}^{(2)}, \ldots, x_{n,m}^{(D)})$, for $n = 1, 2, \ldots, N_m,\ m = 1, 2, \ldots, M$. We consider 6-D measurements consisting of Cartesian coordinates for the tool's position and Euler's roll–pitch–yaw angles for the tool's orientation.

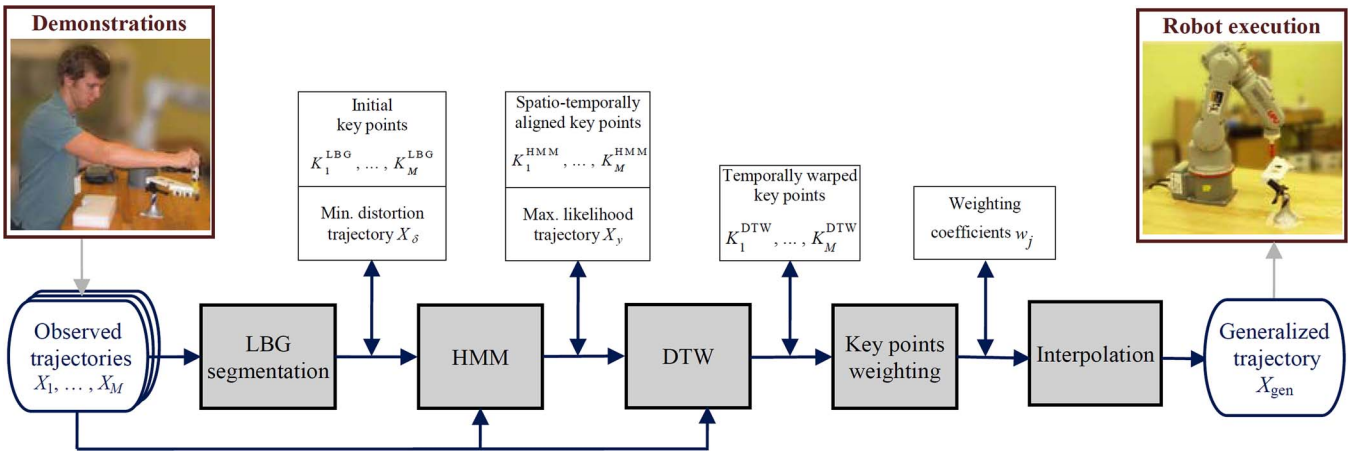


Fig. 1. Diagram representation of the proposed approach for robot PbD. The solid (blue) lines depict automatic steps in data processing. For a set of observed trajectories $X_1, \ldots, X_M$, the algorithm automatically generates a generalized trajectory X_{gen}, which is transferred to a robot for task reproduction.

A schematic diagram depicting the data processing flow for the proposed approach is shown in Fig. 1. A detailed description of the approach follows in the next sections.

B. Selection of Candidate Key Points

Analysis of the recorded trajectories initially involved identification of a set of candidate key points. Sets of criteria for selection of candidate key points for continuous trajectories were proposed in [11] and [21]. In this paper, we introduce a different technique for selecting candidate key points, which is based on classification of discretely labeled data using normalized position and velocity feature vectors. This process results in segmentation of the trajectories based on significant changes in position and velocity. Linde–Buzo–Gray (LBG) algorithm [23] was used here for this purpose. Using such an approach for trajectory segmentation was motivated by the works for human motion identification and classification based on clustering techniques [24], [25].

The LBG algorithm is a variant of the k-means clustering method, i.e., it is used to partition a set of input data points into a number of clusters. A distinct property of the LBG algorithm is the partitioning procedure, which starts with one data cluster followed by iteratively doubling the number of clusters until satisfaction of termination conditions. A short description of the algorithm is given in Table I.

For the problem at hand, the velocity values for each dimension of the recorded trajectories were calculated using an empirically chosen delay value of p sampling periods, i.e., $v_{n,m}^{(d)} = (x_{n+p,m}^{(d)} - x_{n-p,m}^{(d)})/(t_{n+p,m} - t_{n-p,m})$, for $n = 1, \ldots, N_m$ and $d = 1, \ldots, D$, where d denotes the dimensionality of the sequences and t denotes the time instants. The introduced delay smoothes the velocity data in order to prevent the next key points to be selected too close to the previous key points (note that selection of values of the parameters, e.g., p, and other implementation details are discussed in Section V). Velocities of the first and last p sampling periods were assigned to the constant vectors $v_{p+1,m}^{(d)}$ and $v_{N_m-p,m}^{(d)}$, respectively.

To avoid biased results caused by different scaling in the numerical values of the positions, orientations, and velocities,

TABLE I
LBG ALGORITHM

The input data consists of N multidimensional vectors $\mathbf{x}_n \in \mathrm{R}^D$, for $n = 1, 2, ..., N$.

1. Initialization: the number of partitions is set to $k = 1$, and the centroid μ_1 is set equal to the mean of the data vectors $\mathbf{x}_n$, for $n = 1, 2, ..., N$.

2. Splitting: each centroid $\{\mu_j\}_{j=1}^{k}$ is replaced by $\mu_j + \varepsilon$ and $\mu_j - \varepsilon$, where ε is a fixed perturbation vector.

3. Assignment: each vector $\mathbf{x}_n$ is assigned to the nearest centroid, generating a set of clusters $\{C_j\}_{j=1}^{k}$. The Euclidean distance from each data point to the cluster centroids is used as assignment criterion.

4. Centroids updating: the centroids are set equal to the mean of all data vectors assigned to each cluster, i.e., $\mu_j = \text{mean}\left(\mathbf{x}_n, \text{for } \forall \mathbf{x}_n \in C_j\right)$, for $j = 1, 2, ..., k$.

5. Termination: if the Euclidean distance for the data points to the assigned centroid is less than a given threshold, then stop. Otherwise, set $k = 2k$ and return to step 2.

these data sequences were first normalized to sequences with zero mean and unit variance for each dimension of the trajectories. The normalized position, orientation, and velocity vectors $\hat{\mathbf{x}}_{n,m}$ and $\hat{\mathbf{v}}_{n,m}$ were vertically concatenated into a combined $2D$-dimensional vector $\boldsymbol{\alpha}_{n,m} = [\hat{\mathbf{x}}_{n,m}\ \hat{\mathbf{v}}_{n,m}]^T$. Afterward, the vectors from all trajectories were horizontally concatenated into a set of vectors U with the length equal to the total number of measurements from all trajectories, i.e., $\sum_{m=1}^{M} N_m$. Next, the LBG classification algorithm was run on the set U to group all $\boldsymbol{\alpha}_{n,m}$ vectors into k clusters, by assigning each vector $\boldsymbol{\alpha}_{n,m}$ to the label of the closest centroid. The candidate key points for the trajectories were associated with the transitions between the clusters labels. The resulting sequences of key points after the initial segmentation with LBG algorithm were denoted by $K_m^{\text{LBG}} = \{\kappa_{j,m}^{\text{LBG}} = (t_{j,m}, \mathbf{x}_{j,m})\}$, for $j = 1, 2, \ldots$, where t and $\mathbf{x}$ represent the time instances and tool's pose of the jth key point of demonstration m, respectively.

A major challenge in automatic identification of the key points is to determine the number of key points that is sufficient

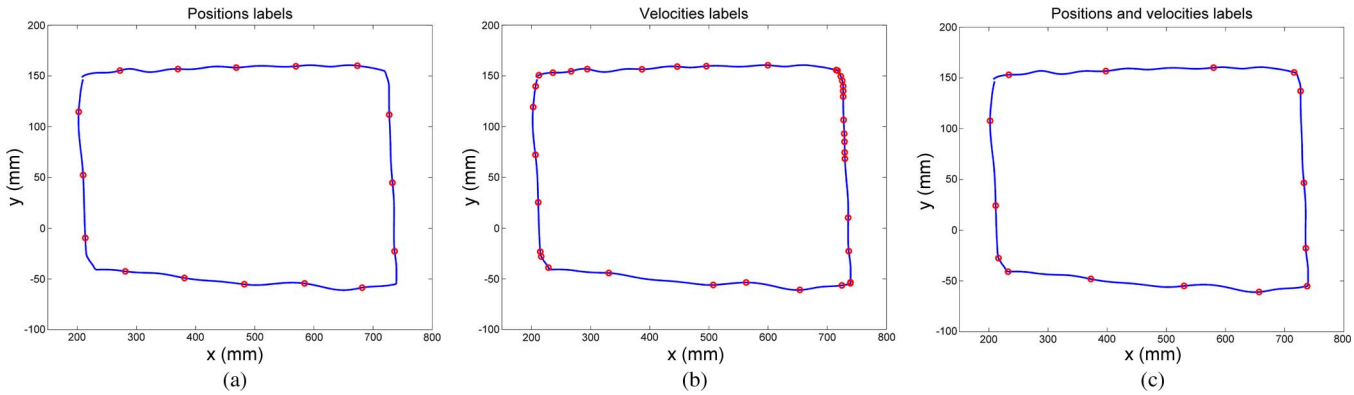


Fig. 2. Example of initial selection of trajectory key points with the LBG algorithm. The key points are indicated with circles. The following input features are used for clustering: (a) normalized positions coordinates $(\hat{x}, \hat{y})$; (b) normalized velocities $(\hat{v}_x, \hat{v}_y)$; and (c) Normalized positions and velocities $(\hat{x}, \hat{y}, \hat{v}_x, \hat{v}_y)$.

to describe the recorded trajectories. Similar to many learning algorithms, this involves a tradeoff between generalization and overfitting. In cases where the number of identified key points is low, some relevant features of trajectories may be excluded from the generalization process. On the other hand, extraction of too many key points may result in overfitting of the demonstrated trajectories. For our case, the number of key points per trajectory is indirectly controlled by prespecifying the number of clusters k based on the complexity of the trajectories.

To illustrate the LBG segmentation, an example of assigning key points for simple 2-D trajectories with rectangular shape is shown in Fig. 2 (the trajectories correspond to painting the contour of a rectangular panel). The number of clusters was empirically set to 16. Fig. 2(a) shows the results when only the normalized position coordinates x and y are used for clustering. The key points, represented by circles in the figure, are associated with changes in x and/or y position coordinates. The selected set of key points missed the corners of the trajectories, which, on the other hand, represent important features of the demonstration, since they convey information about the spatial constraints of the task. Labeling with the LBG algorithm for the same trajectories based solely on the velocity features is shown in Fig. 2(b). For this case, there are significant changes in velocities around the corners of the trajectories (due to the slowing down of the tool motions around the corners of the panel). Fig. 2(c) shows the LBG clustering results for combined positions and velocity data, which yielded improved segmentation of the trajectories, when compared to the results in Fig. 2(a) and (b) for separate positions and velocity features.

The criteria for initial selection of the key points reported in [11] suggest assigning a key point if there is a change in direction for one of the position coordinates greater than a given threshold or if the distance to the previous key point is greater than a given threshold. The authors of [21] proposed an extended set of criteria for key point selection, which among others included the following: 1) the angle between two position vectors of the trajectories is less than a given threshold; 2) the norm between two position vectors is greater than a given threshold; and 3) the number of time frames between the previous key point and the current position is greater than a given threshold. This set of criteria is more complete since key points can be identified even if there is no significant change in any of the individual coordinates, but there is an overall change in direction of the demonstrated trajectory. The drawback of these techniques for key point selection is the requirement for manual adjustment of the threshold parameters for different trajectories. In [21], the recorded trajectories consisted of 6-D positions and orientations of the hand trajectories and 7-D joint angle trajectories. Therefore, the process of selection of candidate key points required to experimentally determine the values of 15 threshold parameters. Our proposed method assigns the key points using only the number of clusters k in the LBG algorithm as a parameter, thus avoiding the manual tuning of a large set of parameters. The number of clusters k here is selected empirically based on the prior knowledge of the trajectory geometry. Note that the selected value of the parameter k can be suited for segmentation purposes with a family of trajectories with similar level of complexity. In addition, the number of clusters k could also be automated by employing the traditional LBG approach and by setting a threshold distance value, below which centroid splitting would cease. However, in this case, the results for key point labeling could suffer from overfitting or missing important trajectory features.

In summary, the stage of selecting candidate key points resulted in different numbers of key points per trajectory, along with variations in spatial position and temporal alignment of the key points. Thus, HMM was used for modeling the trajectories due to its ability in handling spatiotemporal differences in the observed sequences of data. The trajectory that was best described by its set of discrete labels from the LBG algorithm was then used for initialization of parameters of the HMM (Section III-A).

III. Key Point Identification By HMM

This section provides a brief introduction to HMM and the notation used in this paper. The interested reader is referred to [8] for a detailed tutorial on HMM.

An HMM is a directed graphical model that is characterized by a Markov chain of a sequence of (unobserved) hidden state variables and a corresponding sequence of observation variables. This method has been used in speech recognition [26], [27], DNA sequence analysis [28], handwriting recognition [29], etc.

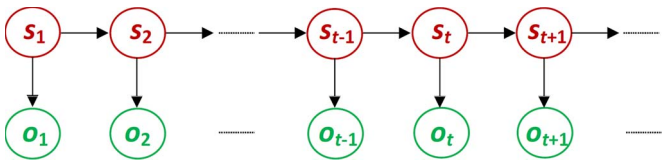


Fig. 3. Graphical representation of an HMM.

A graphical representation of an HMM is shown in Fig. 3, where the hidden states are denoted with s_t and the observed variables are denoted by o_t at time instants $\{1, 2, \ldots, t-1, t+1, \ldots\}$. The probabilities of being in state j at time $t+1$ given that the model was in state i at time t, denoted here by $a_{ij} = P(s_{t+1} = j | s_t = i)$, form the state transition matrix $\mathbf{A} = \{a_{ij}\}$, for $i, j = \{1, 2, \ldots, N_s\}$, where N_s is the number of states of the model. Probabilities of the model being at state i at time $t = 1$ form a vector of initial state probabilities $\boldsymbol{\pi} = \{\pi_i = P(s_1 = i)\}$, for $i = \{1, 2, \ldots, N_s\}$. Probabilities of observing a symbol q_k at time t given the model is in state i, i.e., $b_i(k) = P$ (q_k at $t | s_t = i$), form the observation probability matrix $\mathbf{B} = \{b_i(k)\}$, for $i = \{1, 2, \ldots, N_s\}$ and $k = \{1, 2, \ldots, Q\}$, where Q denotes the number of observation symbols. An HMM is fully described by the set of matrices $\boldsymbol{\lambda} = \{\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}\}$.

A. HMM Codebook Formation and Initialization

In this paper, a discrete HMM was used for segmentation of the observed trajectories which requires the recorded continuous trajectories to be mapped into a codebook of discrete values. The technique applied in Section II was employed here for vector quantization purposes, i.e., the continuous data for positions and velocities were normalized, clustered via LBG algorithm, and associated with the closest cluster centroid. The vector quantization requires us to utilize greater number of clusters k in this case in order to preserve most of the continuous trajectory features. Thus, for a total number of Q clusters used, each continuous measurement of the tools position and orientation $\mathbf{x}_{n,m}$ is mapped into a discrete symbol $o_{n,m}$ from a finite codebook of symbols $\{q_1, q_2, \ldots, q_Q\}$, i.e., $o_{n,m} \in \{q_1, q_2, \ldots, q_Q\}$. These observation sequences form the set $\mathbf{O} = \{O_m = (o_{1,m}, o_{2,m}, \ldots, o_{N_m,m})\}_{m=1}^{M}$, where M is the total number of demonstrations.

It should be noted here that several other types of strategies for codebook formation were attempted. The multidimensional short time Fourier transform [14] was implemented by using the "spectrogram" function in MATLAB. The frequency spectra of each dimension of the data were concatenated to form the prototype vectors, which were afterward clustered into a finite number of classes. Another vector quantization strategy proposed in [11] and [30] was examined, which was based on homogenous (fixed) codebooks. This approach employs linear discretization in time of the observed data. Also, a vector quantization based on the LBG clustering of the multidimensional acceleration, velocity, and magnitude data was tested. The best labeling results were achieved by using the normalized velocity and position approach, which is probably due to hidden dependence of position and velocity features for human trajectories.

The efficiency of learning with HMM in a general case depends directly on the number of available observations. Since in robot PbD the number of demonstrations is preferred to be kept low, a proper initialization of the HMM parameters is very important. Here, initialization is based on a demonstration X_δ selected via the minimum distortion criterion [17], [23], calculated as the time-normalized sum-of-squares error

$$\delta = \arg\min_{1 \le m \le M} \frac{\sum_{n=1}^{N_m} (\boldsymbol{\alpha}_{n,m} - \boldsymbol{\mu}(l_{n,m}))^2}{N_m} \tag{1}$$

where $\boldsymbol{\alpha}_{n,m}$ is the $2D$-dimensional position/velocity vector of time frame n in demonstration m and $\boldsymbol{\mu}(l_{n,m})$ is the centroid corresponding to the label $l_{n,m}$.

Accordingly, the number of hidden states N_s of HMM was set equal to the number of regions between the key points of the sequence X_δ (i.e., number of key points of X_δ plus one). The HMM configuration known as a *Bakis left–right topology*[8] was employed to encode the demonstrated trajectories. The self-transition and forward-transition probabilities in the state transition matrix were initialized as follows: $a_{i,i} = (1 - 1/\tau_{i,\delta})(1/Z)$, $a_{i,i+1} = (1/\tau_{i,\delta})(1/Z)$, and $a_{i,i+2} = (1/4\tau_{i,\delta})(1/Z)$, where $\tau_{i,\delta}$ is the duration (in time steps) of the state i in demonstration X_δ and Z is a stochastic normalizing constant ensuring $\sum_j a_{i,j} = 1$. The transitions $a_{i,i+2}$ allow states to be skipped in the case when key points are missing from some of the observed sequences. All other state transition probabilities were assigned to 0, making transitions to those states impossible. The output probabilities were assigned according to: $b_i(k) = n_{i,\delta}(q_k)/\tau_{i,\delta}$, where $n_{i,\delta}(q_k)$ represents the number of times the symbol q_k is observed in state i of the observation sequence O_δ. The initial state probabilities were set equal to $\boldsymbol{\pi} = [1\ 0\ \ldots\ 0]$. A more detailed explanation on HMM parameter initialization can be found in [8].

B. HMM Training and Segmentation

After the parameters $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ had been initialized, an HMM was trained on all observed sequences $O_1, \ldots, O_M$, using the Baum–Welch algorithm. Afterward, Viterbi algorithm was used to find the most likely hidden state sequence $S_m = s_{1,m}, s_{2,m}, \ldots, s_{N_m,m}$ for a given sequence of observations O_m and a given model. The transitions between states were identified as key points.

Start and end points of the trajectories were also designated as key points because these points are not associated with state transitions in an HMM. Also, due to the adopted Bakis topology of the HMM state flow structure, certain hidden states (and key points) may have not been found in all observed trajectories. Those key points are referred to as *null key points* in this paper. Introducing the concept of null key points ensures that the length of the key points' sequences of all demonstrations would be the same and equal to $N_s + 1$ (recall that N_s denotes the number of states in the HMM). Thus, as an output of the segmentation of the demonstrated trajectories with HMM, there is a set of sequences of key points $K_m^{\mathrm{HMM}} = \{\kappa_{j,m}^{\mathrm{HMM}}\}$, for $j = 1, 2, \ldots, N_s + 1$, that correspond to the same

hidden states (i.e., identical spatial positions) across all trajectories. Generalization of the demonstrations involves clustering and interpolation of these points, as explained in the next section.

IV. Generalization of the Demonstrated Trajectories

A. Temporal Normalization via DTW and Key Point Alignment

Since the demonstrated trajectories differ in the lengths and velocities of the recorded positions, the extracted trajectories' key points by HMM correspond to different time frames of the trajectories. In order to generate a reproduction trajectory, the set of key points needs to be aligned along a common time vector. This will ensure tight parametric curve fitting in generating a generalized trajectory through interpolation of the extracted key points. Otherwise, the temporal differences among the key points within the clusters could cause spatial distortion of the trajectories.

The temporal aligning of the clusters of key points is performed here through alignment of the entire trajectories with a reference trajectory while storing in a memory the newly obtained time stamps of the key points. The DTW algorithm is often used for aligning time series, where the optimal alignment of sequences is based on nonlinear time warping by using a dynamic programming technique. The main advantage of the DTW over linear time scaling of the signals is that it handles the spatial distortion of the signals more efficiently.

DTW sequence alignment is based on forming a matrix of distances between two time series and finding an optimal path through the matrix that minimizes the distance between the sequences. For a given reference sequence $r_1, r_2, \ldots, r_R$ of length R and a test sequence $f_1, f_2, \ldots, f_F$ of length F, the distance matrix is formed as follows:

$$H(u,v) = \|r_u - f_v\|_2, \qquad \text{for } u = 1,\ldots,R,\ v = 1,\ldots,F. \tag{2}$$

In this paper, Euclidean l_2-norm was used as a distance measure, although other norms can also be used (e.g., l_1-norm in [31]). The optimal alignment path $G = \{g_w = (u,v)_w\}$ is calculated by minimizing the cumulative sum of the distances $H_{u,v}$ and the minimum distance between the test and reference sequences of the neighboring cells, i.e.,

$$g(u,v) = H(u,v) + \min\{g(u-1,v), g(u-1,v-1), g(u,v-1)\}. \tag{3}$$

The warped path aligns the elements in the reference and test sequences based on boundary conditions: $g_1 = (1,1)$ and $g_{\text{end}} = (R,F)$.

One-dimensional DTW refers to alignment of time series based on a single dimension of the data, and when dealing with multidimensional data, it might result in suboptimal alignment across the rest of the dimensions. Therefore, for this paper, a multidimensional DTW approach [31] was used, with the distance matrix having the form

$$H(u,v) = \sum_{d=1}^{D} \left\| r_u^{(d)} - f_v^{(d)} \right\|_2 \qquad \text{for } u = 1,\ldots,R,\ v = 1,\ldots,F \tag{4}$$

where D denotes the dimensionality of the sequences.

To select a reference sequence for time warping of the demonstrations, the forward algorithm [8] was used for finding the log-likelihood scores of the demonstrations with respect to the trained HMM, i.e. $\ell(m) = \log P(O_m|\boldsymbol{\lambda})$. The observation O_y with the highest log-likelihood was considered as the most consistent with the given model, and hence, the respective trajectory X_y was selected as a reference sequence.

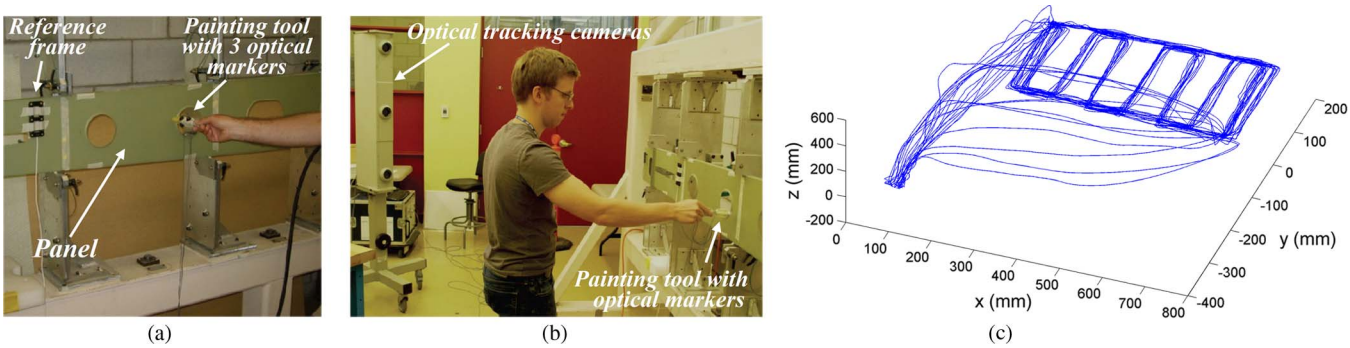
Next, the time vector of the trajectory X_y, i.e., $\mathbf{t}_y = \{t_1, t_2, \ldots, t_{N_y}\}$, was modified in order to represent the generalized time flow of all demonstrations. This was achieved by calculating the average durations of the hidden states from all demonstrations $\overline{\tau_i}$, for $i = 1,2,\ldots,N_s$. Since the key points were taken to be the transitions between hidden states in HMM, the new key point time stamps for the demonstration O_y were reassigned as

$$t_{\kappa_j} = 1 + \sum_{i=1}^{j-1} \overline{\tau_i}, \qquad \text{for } j = 2,\ldots,N_s+1 \tag{5}$$

where t_{κ_j} denotes the new time index of the jth key point and $t_{\kappa_1} = 1$ since the first key point is always taken at the starting location of each trajectory. Subsequently, linear interpolation was used between the new t_{κ_j}'s to find a warped time vector $\mathbf{t}_{y,\text{warped}}$. The reference sequence $X_{y,\text{warped}}$ was obtained from the most consistent demonstrated trajectory X_y with time vector $\mathbf{t}_{y,\text{warped}}$.

Afterward, multidimensional DTWs were run on all demonstrated trajectories against the time-warped data set $X_{y,\text{warped}}$. A shape preserving constraint was imposed, which forced the warped time waves for each demonstration to stay within a specified time frame bandwidth of their normalized time vector, as proposed in [22]. It resulted in a set of warped time waves $G_1, G_2, \ldots, G_M$, which aligned every trajectory with the $X_{y,\text{warped}}$ trajectory. Accordingly, all key points were shifted in time by reassigning the time index of every key point $\kappa_{j,m}$ with the time index contained in the jth member of G_m. The resulting warped sequences of key points were denoted by $K_m^{\text{DTW}} = \{\kappa_{j,m}^{\text{DTW}}\}$, for $j = 1,2,\ldots,N_s+1$.

With the key points from all demonstrations temporally aligned, parametric curve fitting across each dimension would suffice to create a generalized trajectory that represents the set of demonstrated trajectories. However, this approach would fail to separate the deliberate and precise motions from transitional portions of the trajectories (such as approach or depart). Therefore, weighting coefficients were assigned for the different portions of trajectories, as bias coefficients of their relative importance for reproduction.



(a) (b) (c)

Fig. 4. (a) Experimental setup at NRC-IAR's laboratories for the first experiment: panel, painting tool, and reference frame. (b) Perception of the demonstrations with the optical tracking system. (c) Demonstrated trajectories by subjects 1, 2, and 3.

B. Key Point Weighting

Introducing the weighting coefficients was inspired by [32] and [33], where the relative importance of each dimension from a set of measured variables was compared, and for reproduction purposes, greater weights were put on the dimensions that were less variant across different demonstrations.

Similarly, in this paper, weighting coefficients were assigned to all clusters of key points. The proposed weighting is based on the assumption that the key points that are not closely clustered compared to the rest of the key points are generally less significant for reproduction. The root-mean-square error (RMSE) was utilized to measure the closeness of each cluster of key points as follows:

$$\mathrm{RMSE}(k_j) = \sqrt{\sum_{m=1}^{M} \left(\kappa_{j,m}^{\mathrm{DTW}} - \overline{\kappa}_j^{\mathrm{DTW}}\right)^2} \tag{6}$$

where $\overline{\kappa}_j^{\mathrm{DTW}}$ denotes the centroid of the cluster k_j of key points with time index j. For each key points' cluster, a normalized weighting coefficient is calculated based on

$$w_j = \begin{cases} 0, & \text{for} \quad \mathrm{RMSE}(k_j) \geq \varepsilon_{\max} \\ \frac{\varepsilon_{\max} - \mathrm{RMSE}(k_j)}{\varepsilon_{\max} - \varepsilon_{\min}}, & \text{for} \quad \varepsilon_{\min} < \mathrm{RMSE}(k_j) < \varepsilon_{\max} \\ 1, & \text{for} \quad \mathrm{RMSE}(k_j) \leq \varepsilon_{\min} \end{cases} \tag{7}$$

where $\varepsilon_{\min}$ and $\varepsilon_{\max}$ are empirically chosen threshold RMSE values. The null key points were not used for weighting or interpolation; instead, they were introduced simply to ensure that the key points with the same index correspond to identical spatial positions of the trajectories.

C. Spline Fitting and Interpolation

After determining the weights of the key points' clusters, a generalized trajectory can be created by using a number of curve fitting techniques. A penalized cubic smoothing-spline regression [34] was used here for that purpose. This technique is well suited for fitting of a smooth curve to scattered data, such as generation of a smooth trajectory from the set of key points from multiple trajectories. Each dimension of the trajectories was fitted parametrically over time with the corresponding weighting coefficients. The spline curve was interpolated at intervals equal to the sampling period, resulting in a generalized trajectory X_{gen} suitable for following by a robot.

V. Experiments

To evaluate the approach, two experiments were conducted for a manufacturing process. A hand tool was moved by an operator in a simulated painting process during which certain areas of two planar parts were traversed. For practical reasons, the paintings were done in a dry form, i.e., the tool used does not dispense paint during the experiments. The first experiment followed a simple trajectory, while the second experiment used a more complex geometry and trajectory, which included non-planar regions and waving motions of the painting tool with different amplitudes.

A. Experiment of Learning Trajectories With Simple Geometry

The experimental setup of the simulated painting task for the first experiment is shown in Fig. 4, where a demonstrator is performing virtual painting of the panel using a hand tool that serves as a spray gun. The attached optical markers on the tool were tracked by an optical tracking system Optotrak Preseon (Fig. 4(b)). The pose of the tool with respect to the reference frame of the panel, shown in Fig. 4(a), was recorded with a frequency of 100 Hz. The task trajectory consisted of approaching the upper left-side corner of the panel from an initial position of the tool, painting the rectangular contour of the panel in a clockwise direction, painting the inner part of the panel with waving left-to-right motions, and bringing the tool back to the initial position.

The task was demonstrated four times by four different operators, i.e., the measurement set consisted of trajectories $\{X_m\}$, for $m = 1, 2, \ldots, 16$, containing the position and orientation data. The robustness of the learning process can be enhanced by eliminating the demonstrations which are inconsistent with the acquired data set [6]. Otherwise, the learning system might fail to generalize correctly from a set of demonstrations which are too dissimilar. The sources of variations can be attributed to the uncertainties arising from the following: 1) involuntary motions of the demonstrator; 2) different conditions in performing the demonstrations; 3) difference in the performance among the demonstrators; 4) fatigue due to multiple demonstrations of

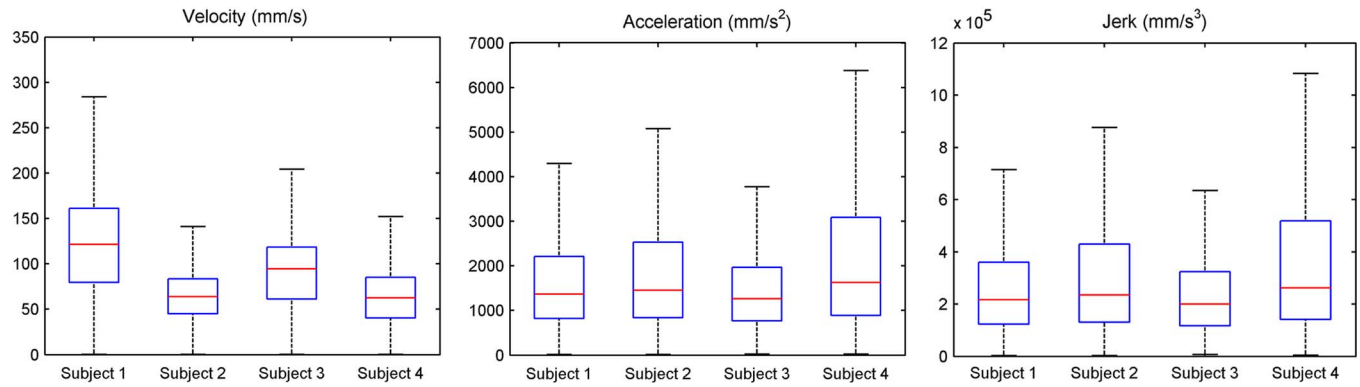


Fig. 5. Distributions of velocities, accelerations, and jerks of demonstrated trajectories by the subjects and the generalized trajectory.

a same task, etc. This paper emphasizes on the differences between the human demonstrators, through introduction of a preliminary step for evaluation of the demonstrators' performance. This step carries out an initial comparison of the distributions of velocities, accelerations, and jerks of the demonstrated trajectories. For each subject, the values of the parameters from all trajectories were combined, and the distributions of the parameters were represented with box-plots in Fig. 5. The plots, for example, indicate that the demonstrations by subject 4 have the greatest deviations of the accelerations and the jerks, while for the rest of the subjects, the distributions are more uniform. Therefore, the demonstrations by subject 4 were eliminated from the recorded data set. These parameters can also be exploited as criteria regarding the motor capabilities of the robot learner. For instance, if the levels of accelerations of the demonstrations are greater than the maximum accelerations available from the robot link motors, the executed task will differ from the demonstrated task, and it can even lead to a task failure, depending on the circumstances. Consequently, the input set of trajectories was reduced to $M = 12$ trajectories, demonstrated by subjects 1, 2, and 3. This initial refinement results in a more consistent initial data set and leads to generation of a more robust task model. The position trajectories of the demonstrations are shown superimposed in Fig. 4(c), including the approaching and departing parts. The number of measurements of demonstrations (N_m) varied from 3728 to 7906 measurements, with an average of 5519. One can infer that, despite the similarity of the trajectories in Fig. 4(c), velocities of the demonstrations varied significantly between subjects.

After an initial preprocessing of the data, which included smoothing of the recorded trajectories with a Gaussian filter of length 10, candidate key points were identified using the LBG algorithm, as explained in Section II. The parameter p was chosen equal to 20 frames (200 ms), with the number of clusters $k = 64$. These parameters were chosen heuristically to balance the complexity and time duration of the demonstrated trajectories. The number of key points per trajectory ranged from 77 to 111 (average value of 87.75 and standard deviation of 11.7). For the minimum distortion trajectory X_{12}, 82 key points were identified. The key points which were closer than 20 time frames were eliminated from the minimum distortion trajectory, resulting in 72 key points in total (shown in Fig. 6).

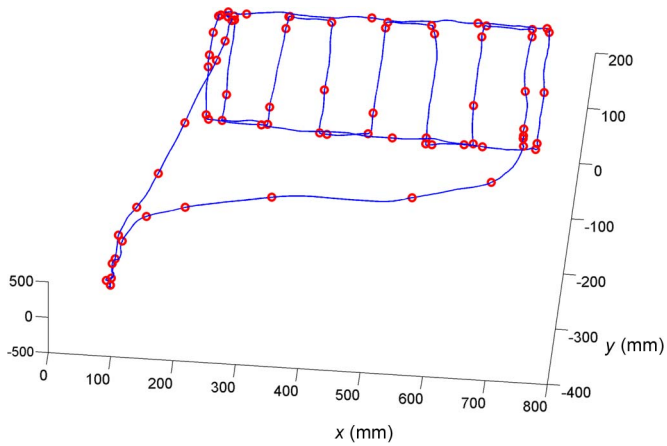


Fig. 6. Initial selection of the key points for the trajectory with minimum distortion.

Afterward, the recorded trajectories were mapped into discrete sequences of observed symbols, and a discrete HMM was trained. For that purpose, Kevin Murphy's HMM Toolbox [35] was used in MATLAB environment. The number of discrete observation symbols Q was set to 256. The number of hidden states N_s was set equal to the number of key points of the minimum distortion trajectory plus 1, i.e., 83. The Viterbi algorithm was employed for finding the sequences of hidden states for each observed sequence, and the key points were assigned to the position and orientation values of the trajectories that corresponded to transitions between the hidden states in the observation sequences. Thus, including the starting and ending points of the trajectories, each demonstration was described by an indexed sequence of 84 key points. Note that, for observation sequences with missing hidden states, the coordinates of the corresponding key points were set to zero (for that reason, they were referred to as null key points). This resulted in spatiotemporal ordering of the key points for all trajectories with a one-to-one mapping existing between the key points from separate demonstrations (e.g., the key points with index $j = 16$ corresponded to the upper right corner of the contour). Log-likelihoods of the observation sequences for the trained HMM were obtained with the forward algorithm. The observation sequence O_3 had the highest log-likelihood, and therefore, the demonstration X_3 was used for temporal alignment of the observed sequences with the DTW algorithm.

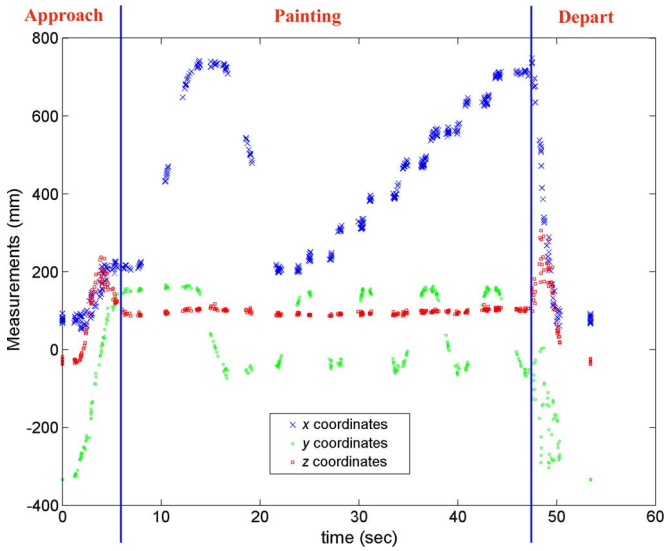


Fig. 7. Spatiotemporally aligned key points from all demonstrations. For the parts of demonstrations which correspond to the approaching and departing of the tool with respect to the panel, the clusters of key points are more scattered when compared to the painting part of the demonstrations.

To obtain a generalized trajectory that would represent the whole set, its length should be approximately equal to the averaged length of the trajectories in the set. Hence, the average durations of the hidden states for the trained HMM were calculated, whereby the newly created sequence of hidden states had a length of 5522 time frames, which is indeed very close to the average length of the demonstrations (5519). This time vector was used to modify the time flow of the demonstration X_3 as explained in Section IV-A. Subsequently, it was used as a reference sequence for the DTW algorithm to align each of the demonstrated trajectories against it. DTW alignment resulted in time-warped trajectories with a length of 5522 time frames, from which the time instances of the key points were extracted. The reassigned time instances of the key points (e.g., with index 16) were now corresponding to time-warped trajectories with equal lengths. The position coordinates x, y, and z of the key points from all demonstrations are shown in Fig. 7. Obviously, for the approaching and departing parts of trajectories, the variance among the key points is high, which implies that these parts are not very important for reproduction. Therefore, weighting coefficients with values close to zero were assigned for those parts of the trajectories. The variance of the key points associated with painting of the contour and inner side of the panel is low, and the computed weighting coefficients had values close to 1. Values of 0.5 and 2 standard deviations of the RMSE for the corresponding cluster were adopted for $\varepsilon_{\min}$ and $\varepsilon_{\max}$ in (7). The last step dealt with smoothing spline interpolation through the key points by using an empirically determined smoothing factor $\lambda = 0.975$, which gave both satisfactory smoothness and precision in position for our application. The $x{-}y{-}z$ coordinates of the resulting generalized trajectory X_{gen} are shown in Fig. 8. For the roll–pitch–yaw angles, the 1-D generalization results are shown in Fig. 9. The plots show the angles of the key points and the interpolated curves for each of the angle coordinates.

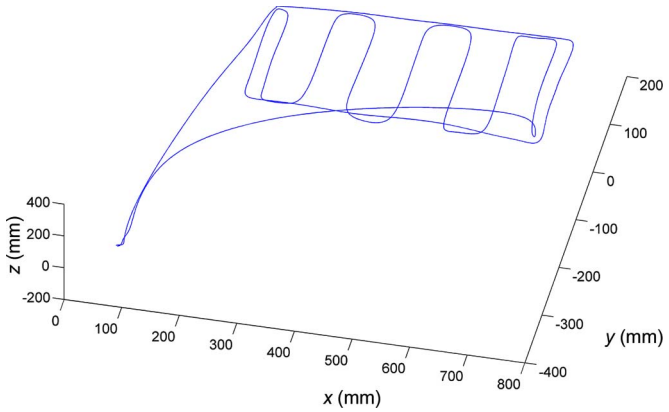


Fig. 8. Generalized trajectory for the $x{-}y{-}z$ position coordinates of the first experiment.

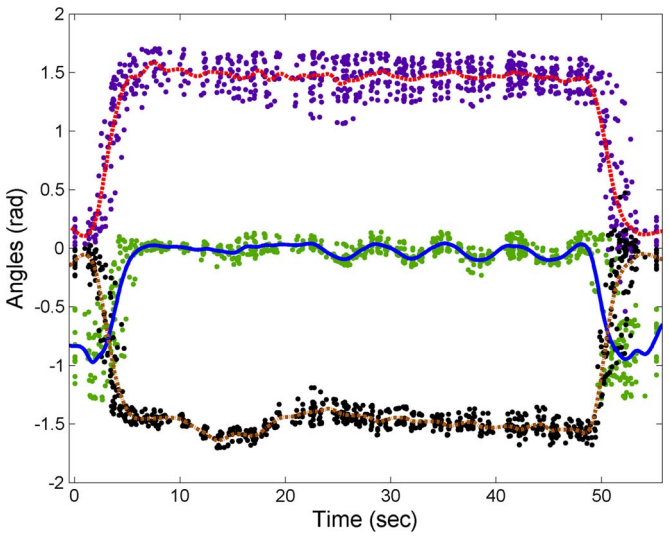


Fig. 9. Generalization of the tool orientation from the spatiotemporally aligned key points. Roll angles are represented by a dashed (red) line, pitch angles are represented by a solid (blue) line, and yaw angles are represented by a dash-doted (brown) line. The dots in the plot represent the orientation angles of the key points.

For comparison, the distributions of velocities, accelerations, and jerks for the demonstrated trajectories by the three subjects and the generalized trajectory are shown in Fig. 10. From the plot of velocity distributions, it can be concluded that the mean value of the generalized trajectory is approximately equal to the average value of the velocities from the demonstrations by all subjects. This is a result of the generalization procedure, where demonstrations by all subjects were used to find a generalized trajectory that will represent the entire set of demonstrations. Furthermore, the distributions of accelerations indicate that the values of the generalized trajectory are lower than the values of the demonstrated trajectories. It is due to the fact that spline smoothing interpolation was employed for reconstruction of the generalized trajectory, which reduced the changes of velocities present in human motions. Third, the jerk distribution for the generalized trajectory is significantly condensed when compared to the demonstrated trajectories. Namely, the inconsistency of the human motions represented by high levels of jerk was filtered out by the generalization

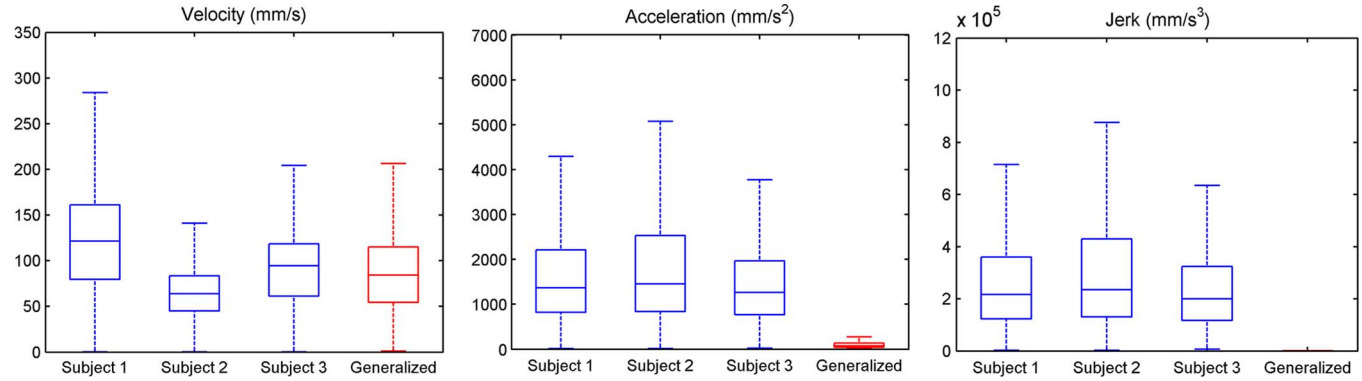


Fig. 10. Distributions of velocities, accelerations, and jerks of demonstrated trajectories by the subjects and the generalized trajectory.

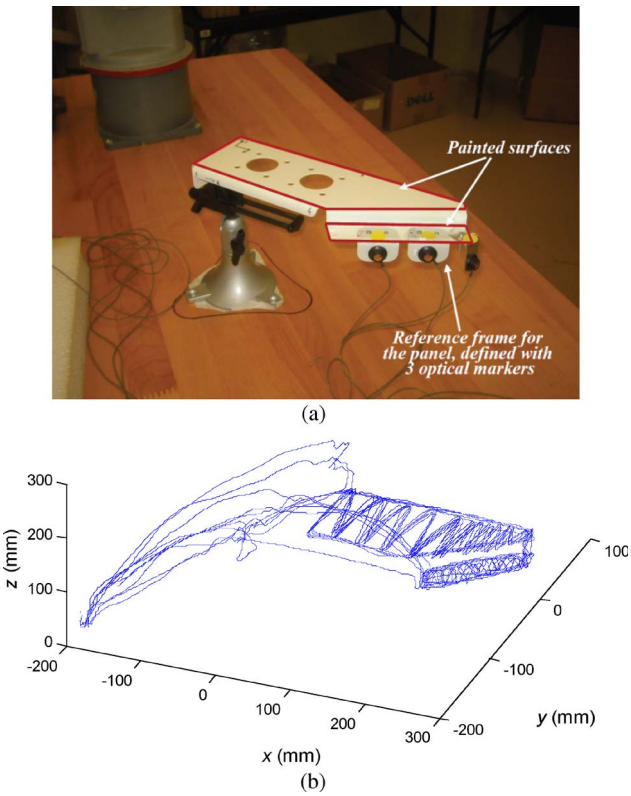


Fig. 11. (a) Panel used for the second experiment. The surfaces to be painted are bordered with solid (red) lines. (b) Demonstrated trajectories.

procedure, resulting in a smooth trajectory with low levels of jerk.

B. Experiment of Learning Trajectories With More Complex Geometry

The second experiment consisted of painting a panel with more complex geometry (Fig. 11). The goal was to paint the top side of the panel and the horizontal plane on the right-hand side of the panel, both marked with solid lines in Fig. 11(a). The three optical markers shown in the picture were used to define the reference coordinate system of the panel. The task was demonstrated five times by a single demonstrator, and all trajectories $X_1, \ldots, X_5$ are shown superimposed in Fig. 11(b). If in the first experiment the shape of the generalized trajectory

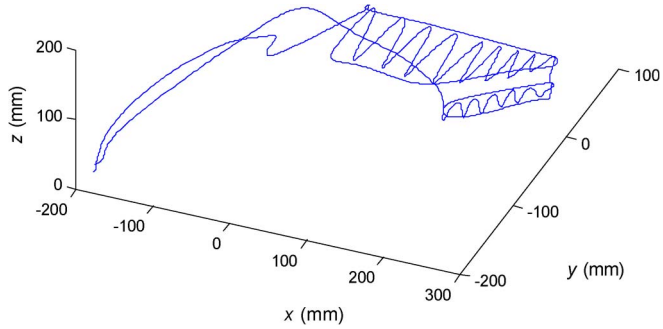


Fig. 12. Generalized trajectory for the second experiment.

was obvious from Fig. 4(c), in this case, it is not easy to infer the generalized trajectory, especially for the small right-hand plane where the waving motions are not controlled.

The lengths of the demonstrations ranged between 4028 and 4266 measurements. The initial selection of candidate key points with the number of clusters k set to 64 resulted in numbers of key points between 122 and 134. Based on the sequence with minimum distortion, the number of hidden states N_s for training the HMM was set to 123. Afterward, the demonstrations were temporally aligned with DTW, with the length of the sequence of average state duration of 4141 time frames. The resulting set of key points from all five demonstrations was interpolated to generate a generalized trajectory X_{gen} (Fig. 12). The plot also shows the approach and retreat of the painting tool with respect to the panel.

The generalized trajectories for the pose of the painting tool were transferred to an industrial robot (Motoman UPJ), followed by execution by the robot (see Fig. 1). The height of the robot's end-effector was set up to 30 mm above the part. Our objective was not to perform the actual painting with the robot but to illustrate the approach for learning complex trajectories from observations of demonstrated tasks. The results showed that, with encoding of demonstrations at the trajectory level, complex demonstrated movements can be modeled and generalized for reproduction by a robot in a PbD environment.

VI. Comparison with Related Works

To compare the described method with the similar works in the literature, the trajectory for task reproduction based on the approach by Calinon and Billard reported in [11] is shown in

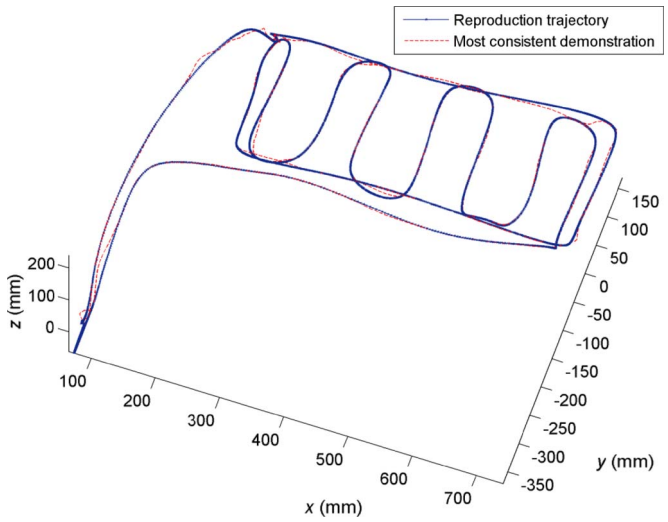


Fig. 13. Generated trajectory for the reproduction of the task of panel painting from Section V-A based on Calinon and Billard [11]. The most consistent trajectory (dashed (red) line) corresponds to the observation sequence with the highest likelihood of being generated by the learned HMM.

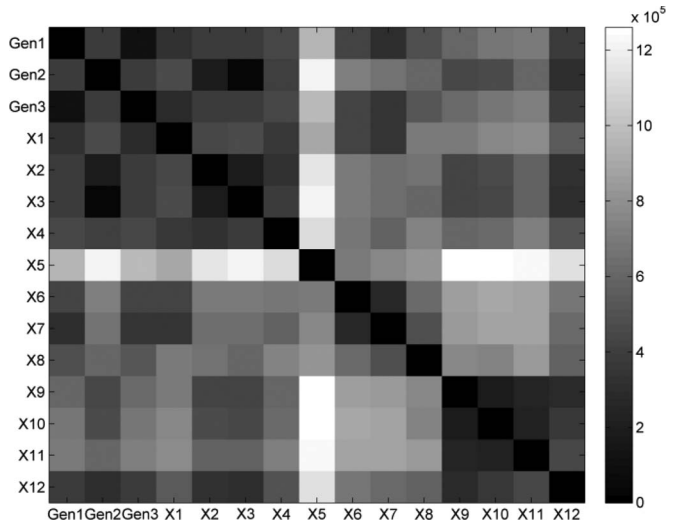


Fig. 14. RMS differences for the reproduction trajectories generated by the proposed approach (Gen_1), the approaches proposed in [11] (Gen_2), [21] (Gen_3), and the demonstrated trajectories (X_1–X_{12}). As the color bar on the right side indicates, lighter nuances of the cells depict greater RMS differences.

Fig. 13. The demonstration set from Section V-A was used, and the result was obtained by using the same parameters presented in this paper. The initial key point selection based on the LBG algorithm was employed in order to compare the approaches on the most similar basis. The trajectory for task reproduction was generated by smoothing spline interpolation of the key points generated from the Viterbi path of the most consistent trajectory, as explained in [11]. The most consistent trajectory is shown with the dashed line in Fig. 13. The main drawback of this approach is that it does not generalize from the demonstrated set. Instead, the learned trajectory is obtained by using a criterion for selecting the best demonstration from the recorded set, followed by a smoothing procedure. In comparison, the trajectory for the task reproduction by our proposed approach shown in Fig. 8 was obtained by taking into account the features from the demonstrated set of trajectories, and it does not rely so heavily on only one of the demonstrated trajectories.

Another advantage of our approach with respect to the work in [11] is in providing a mechanism for interpretation of the intent of the demonstrator. Based on the variance of the key points which correspond to the same spatial features across the demonstrated set, the method presented here assigns different weights for reproduction purposes. For instance, for the approaching and departing (transitional) sections of the trajectories, the variability across the demonstrated set was high, and subsequently, the generalized trajectory does not follow strictly the demonstrated key points. On the other hand, the section of the trajectories corresponding to painting the panel was more constrained, which was reflected in the generation of the generalized trajectory with tight interpolation of the identified key points. It can be observed in Fig. 8 that the approaching and departing sections are smoother due to the higher variance across the demonstrations, while in Fig. 13, the approaching and departing sections simply follow the most consistent trajectory. The adopted procedure has similarities to the work in [4] based on GMM/Gaussian mixture regression (GMR), where the sections of the demonstration with high values of the Gaussian covariances are considered less important for reproduction purposes and vice versa.

The work by Asfour *et al.* [21], which employs generalization from the set of demonstrations by using the concept of common key points, is similar to our proposed approach, although it may provide suboptimal results in some cases. For instance, if a key point is missing in one of the demonstrated trajectories and it is present in all of the remained trajectories, this key point will be eliminated as not being common, and it will not be considered for generalization purposes. To avoid this case, the approach might require applying a heuristic for analysis of the demonstrations and elimination of the suboptimal demonstrations.

The results from the method proposed here and the methods reported in [11] and [21] are compared by using the RMSE as a metric for evaluation of learned trajectories [36]

$$e_{m_1,m_2} = \sum_{n} \left\| \mathbf{x}_{n,m_1} - \mathbf{x}_{n,m_2} \right\|, \qquad \text{for } n = 1, 2, \ldots \quad (8)$$

where m_1 and m_2 were used for indexing the trajectories. The trajectories were scaled to sequences with the same length by using two techniques: linear scaling and DTW alignment. With the linear scaling, the length was set equal to the average length of the demonstrated trajectories. For the DTW scaling, the entire set of trajectories was aligned against one of the demonstrated trajectories. Rather than selecting the generalized trajectory for scaling the data set with DTW, we used some of the demonstrated trajectories as reference sequences for alignment to avoid biased results for the RMS differences.

A color map table with the RMS differences for the linearly scaled trajectories from the Section V-A is shown in Fig. 14. The first three trajectories in the table represent the generalized trajectories obtained by our proposed approach (Gen_1), by the approach proposed in [11] (Gen_2), and the one proposed in [21] (Gen_3). The rest of the cells correspond to the demonstrated trajectories X_1–X_{12}. The darker colors in the table pertain to cells with smaller RMS differences. For instance, the trajectory

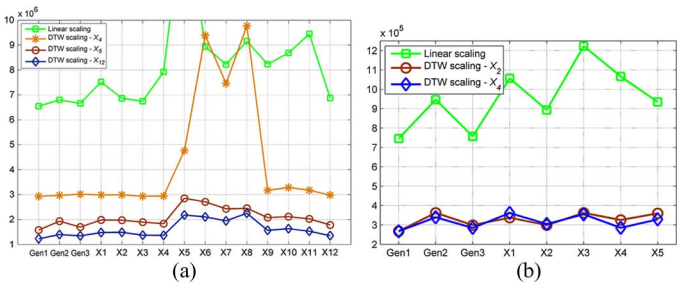


Fig. 15. Cumulative sums of the RMS differences for the reproduction trajectories generated by the proposed approach (Gen_1), the approaches proposed in [11] (Gen_2), and [21] (Gen_3). (a) Demonstrated trajectories (X_1–X_{12}) from the experiment in Section V-A. (b) Demonstrated trajectories (X_1–X_5) from the experiment in Section V-B.

X_5 is the most dissimilar to the whole set, and it can be noticed that the trajectories demonstrated by the same subject X_1–X_4, X_5–X_8, and X_9–X_{12} are more similar when compared to the trajectories demonstrated by the other subjects.

The sums of the RMS differences for the trajectories from Fig. 14 are shown in Fig. 15(a). The RMS differences for scaling of the trajectories with DTW are also given. The demonstrated trajectories X_4, X_5, and X_{12} were chosen as reference sequences for aligning the trajectories. The DTW algorithm was repeated for three of the demonstrated trajectories to avoid any subjective results which can arise from using a particular trajectory as a reference sequence. As expected, the DTW algorithm aligns the trajectories better than the simple linear scaling. For the presented four cases, the cumulative RMS differences for the trajectory obtained by the proposed approach (Gen_1) are the smallest, which infers that it represents the best match for the demonstrated set.

Similarly, for the second experiment from Section V-B, the sums of the RMS differences are shown in Fig. 15(b). The DTW alignment was repeated for two of the demonstrated trajectories—X_2 and X_4. Based on the presented results, it can be concluded that the generalized trajectory generated by the proposed approach fits better the demonstrated set, when compared to the similar approaches proposed in [11] and [21].

Another work in the literature that has some similarities to the one proposed here was reported by Gribovskaya and Billard [37]. In their study, DTW was employed initially to align the recorded sequences. Afterward, a continuous HMM was employed for encoding the trajectories. One drawback of such procedure is that the DTW algorithm distorts the trajectories, and as a result, important spatial information from the demonstrations may be lost. In addition, the velocity profile of the DTW aligned demonstrations is deformed. In our presented method, HMM was used for segmentation of the trajectories as recorded, and DTW was used before the interpolation procedure, only to shift the key points to a trajectory with a given time duration (which was found as the average of time durations of the hidden states form HMM). Hence, the velocity information was preserved for modeling of the demonstrations with HMM, and the DTW affected the trajectories only at the level of key points. On the other hand, a disadvantage of our proposed method is in using discrete HMMs since mapping of the trajectory measurements to a codebook of discrete symbols can also cause local deformations of the trajectories.

TABLE II
MEAN VALUES AND STANDARD DEVIATIONS OF THE COMPUTATION TIMES FOR LEARNING OF TRAJECTORIES FROM SECTIONS V-A AND V-B BY APPLYING THE APPROACH PROPOSED IN THIS PAPER

Steps of the program code:	CPU times (seconds)	
	First experiment	Second experiment
1. Initial preprocessing (smoothing, removing NANs)	0.618 (± 0.267)	0.266 (± 0.006)
2. LBG clustering (using 64 symbols)	38.38 (± 23.195)	5.688 (± 0.504)
3. HMM initialization (discretization with 256 symbols)	185.172 (± 27.053)	51.328 (± 16.76)
4. HMM training	108.407 (± 22.359)	149.43 (± 36.003)
5. HMM inference	24.404 (± 0.814)	12.742 (± 0.719)
6. DTW alignment	44.501 (± 0.347)	23.922 (± 0.368)
7. Key point weighting and interpolation	2.563 (± 1.037)	2.28 (± 0.01)
Total:	404.045 (± 26.808)	245.726 (± 39.21)

TABLE III
MEAN VALUES AND STANDARD DEVIATIONS OF THE COMPUTATION TIMES FOR LEARNING OF TRAJECTORIES FROM SECTIONS V-A AND V-B BY APPLYING THE GMM/GMR APPROACH

Steps of the program code:	CPU times (seconds)	
	First experiment	Second experiment
1. Initial preprocessing (smoothing, removing NANs)	0.495 (± 0.004)	0.208 (± 0.003)
1. DTW alignment	40.164 (± 0.109)	20.032 (± 0.377)
2. GMM encoding	847.771 (± 140.27)	948.6 (± 197.945)
3. GMR	0.697 (± 0.103)	1.223 (± 0.023)
Total:	889.129 (± 140.166)	953.862 (± 197.882)

The computational expense of the proposed approach was compared with the state-of-the-art approach reported in [4] and [11], which use GMM and GMR for modeling and generalization of continuous movements in robot PbD. The simulations were run on a 2.1-GHz dual core CPU with 4 GB of RAM running under Windows XP in the MATLAB environment. Since there are slight variations for the processing times during different simulations, the codes were run five times, and the mean values and standard variations were reported. The computation times for obtaining of a generalized trajectory from the sets of measured trajectories for both experiments from Section V are presented in Tables II and III. The DTW alignment step of the program was performed with a MATLAB Executable (MEX) file, which increased the computational speed of the DTW algorithm for about six to eight times compared to the standard m-files. The results of the computational cost for the proposed approach indicate that most of the processing time was spent on learning the HMM parameters and discretization of the continuous trajectories. For the reported computation times for the GMM/GMR method in Table III, the same MATLAB MEX subroutine for performing DTW alignment in our proposed approach was used. For the first experiment, the trajectory X_{12} (with the length of 4939 measurements) was selected as a reference sequence for aligning the set of 12 trajectories, whereas for the second experiment, the trajectory X_2 (4048 measurements) was selected as a reference sequence for aligning the set of five trajectories. For the steps of GMM encoding and

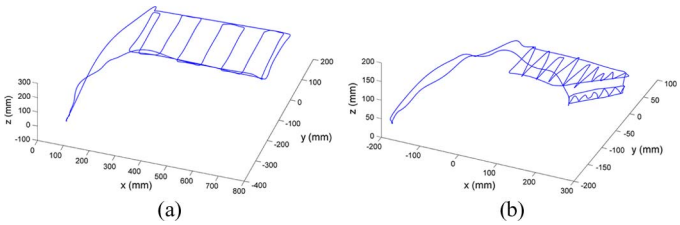


Fig. 16. Generalized trajectory obtained by the GMM/GMR method [4] for the (a) first experiment and (b) second experiment in Section V.

GMR generalization, the MATLAB codes provided in [4] were used. The number of Gaussian components for modeling the trajectories was approximated by trial and error, and the values used for the reported times were 40 for the first experiment and 75 for the second experiment. The generalized trajectories generated from the GMM/GMR method are shown in Fig. 16. Based on the results from Table III, it can be concluded that the computational cost of modeling the 6-D input data into a large number of Gaussians is computationally expensive. The total computation times with the proposed approach were less than the times obtained with the GMM/GMR approach for both sets of trajectories.

VII. Conclusion

In this paper, a method for trajectory learning and generation using a robot PbD approach has been proposed. Key points related to the transitions between different types of movements were used for encoding the relevant features of the trajectories. Hidden state transitions of an HMM were assumed to correspond to the key points in the trajectories. DTW technique was used for temporal clustering of all key points into a format that represents the generalized time-flow of all demonstrations. Weighting coefficients were introduced based on the variability of the key points across the demonstrations to account for the significance of the key points in the reproduction of the different sections. The proposed approach was implemented for tasks of painting a panel. The generalized trajectories of the painting tool were used to generate input commands for a robot, leading to task execution by the robot.

The RMS differences were used for comparison of the proposed approach to the approaches reported in [11] and [21] for the set of trajectories considered for learning. Different scaling of the trajectories was applied in order to avoid biased results. The obtained values for the RMS differences indicated an improved fit by our proposed approach when compared to the other two approaches and to the individual demonstrated trajectories. The computational efficiency of our method was compared to the GMM/GMR method. The results indicated that the GMM/GMR method had two to four times greater computation time for the considered trajectories when compared to the proposed approach.

The entire process of initial key point selection, HMM initialization, trajectory segmentation, DTW alignment of the key points, weighting of the different parts of the trajectories, and interpolation (Fig. 1) was automated. In other words, the proposed algorithm was capable of processing the recorded demonstration data and extracting a generalized trajectory without user's intervention, and it is therefore suited for implementation in robot PbD. The robustness of the learning process is enhanced by elimination of outlier trajectories that are inconsistent with the demonstrated set and by assigning lower weights for reproduction to the key points with high variance across the demonstrations.

For initial selection of the trajectory key points, we have proposed the use of a clustering technique which employs the LBG algorithm. This procedure segments the trajectories by clustering the data points into bins of different positions and/or velocities. Thus, the need for manual adjustment of a large set of parameters to identify the key points (as proposed in [11], [21]) has been eliminated.

Compared to other works which use key points for reproduction of human trajectories, in our proposed approach, all of the key points from all demonstrations were exploited to create a generalized trajectory. This represents an improvement toward generating a generalized trajectory that combines the essential features from the whole set of demonstrations. This generalization property is especially important for learning manual trajectories with complex forms such as the case that we have considered here, i.e., the trajectories of a tool in a manufacturing process.

References

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and K. Oussama, Eds. New York: Springer-Verlag, 2008, ch. 59.

[2] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[3] M. Lopes and J. Santos-Victor, "A developmental roadmap for learning by imitation in robots," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 2, pp. 308–321, Apr. 2007.

[4] S. Calinon, *Robot Programming by Demonstration: A Probabilistic Approach*. Boca Raton, FL: EPFL/CRC Press, 2009.

[5] M. Pardowitz, R. Zoellner, S. Knoop, and R. Dillmann, "Incremental learning of tasks from user demonstrations, past experiences and vocal comments," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 2, pp. 322–332, Apr. 2007.

[6] J. Aleotti and S. Caselli, "Robust trajectory learning and approximation for robot programming by demonstration," *Robot. Auton. Syst.*, vol. 54, no. 5, pp. 409–413, Mar. 2006.

[7] F. Janabi-Sharifi and W. J. Wilson, "Automatic grasp planning for visual-servo controlled robotic manipulators," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 28, no. 5, pp. 693–711, Oct. 1998.

[8] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[10] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. Int. Conf. Mach. Learn.*, Williamstown, NJ, 2001, pp. 282–289.

[11] S. Calinon and A. Billard, "Stochastic gesture production and recognition model for a humanoid robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sendai, Japan, 2004, pp. 2769–2774.

[12] A. Vakanski, F. Janabi-Sharifi, I. Mantegh, and A. Irish, "Trajectory learning based on conditional random field for robot programming by demonstration," in *Proc. IASTED Int. Conf. Robot. Appl.*, Cambridge, MA, 2010, pp. 401–408.

[13] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 2, pp. 286–298, Apr. 2007.

[14] J. Yang, Y. Xu, and C. S. Chenc, "Human action learning via hidden Markov model," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 27, no. 1, pp. 34–44, Jan. 1997.

[15] J. Gu, X. Ding, S. Wang, and Y. Wu, "Action and gait recognition from recovered 3-D human joints," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 4, pp. 1021–1033, Aug. 2010.

[16] A. Irish, I. Mantegh, and F. Janabi-Sharifi, "A PbD approach for learning pseudo-periodic robot trajectories over curved surfaces," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron.*, Montreal, QC, Canada, 2010, pp. 1425–1432.
[17] S. K. Tso and K. P. Liu, "Demonstrated trajectory selection by hidden Markov model," in *Proc. IEEE Int. Conf. Robot. Autom.*, Albuquerque, NM, 1997, pp. 2713–2718.
[18] T. Inamura, N. Kojo, and M. Inaba, "Situation recognition and behavior induction based on geometric symbol representation of multimodal sensorimotor patterns," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Beijing, China, 2006, pp. 5147–5152.
[19] D. Lee and Y. Nakamura, "Stochastic model of imitating a new observed motion based on the acquired motion primitives," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Beijing, China, 2006, pp. 4994–5000.
[20] T. Inamura, H. Tanie, and Y. Nakamura, "Keyframe compression and decompression for time series data based on the continuous hidden Markov model," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Las Vegas, NV, 2003, pp. 1487–1492.
[21] T. Asfour, F. Gyarfas, P. Azad, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in a humanoid robot," in *Proc. 6th IEEE-RAS Int. Conf. Human. Robots*, Genoa, Italy, 2006, pp. 40–47.
[22] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-26, no. 1, pp. 43–49, Feb. 1978.
[23] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84–95, Jan. 1980.
[24] F. Zhou, F. De la Torre, and J. K. Hodgins, "Aligned cluster analysis for temporal segmentation of human motion," in *Proc. IEEE Conf. Autom. Face Gestures Recogn.*, Amsterdam, Netherlands, 2008, pp. 1–7.
[25] G. Liu, J. Zhang, W. Wang, and L. McMilan, "A system for analyzing and indexing human-motion databases," in *Proc. Int. Conf. Manage. Data*, Baltimore, MD, 2005, pp. 924–926.
[26] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*. Upper Saddle River, NJ: Prentice-Hall, 1993.
[27] F. Jelinek, *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press, 1997.
[28] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[29] R. Nag, K. Wong, and F. Fallside, "Script recognition using hidden Markov models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Tokyo, Japan, 1986, pp. 2071–2074.
[30] H.-K. Lee and J. H. Kim, "An HMM-based threshold model approach for gesture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 10, pp. 961–973, Oct. 1999.
[31] G. A. ten Holt, M. J. T. Reinders, and E. A. Hendriks, "Multi-dimensional dynamic time warping for gesture recognition," in *Proc. 13th Annu. Conf. Adv. School Comput. Imag.*, Heijen, Netherlands, 2007.
[32] S. Calinon and A. Billard, "Learning of gestures by imitation in a humanoid robot," in *Imitation and Social Learning in Robots, Humans and Animals: Social and Communicative Dimension*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2007, ch. 8.
[33] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robot. Auton. Syst.*, vol. 47, no. 2/3, pp. 69–77, Jun. 2004.
[34] J. Rice and M. Rosenblatt, "Smoothing splines: Regression, derivatives and deconvolution," *Ann. Stat.*, vol. 11, no. 1, pp. 141–156, Mar. 1983.
[35] K. Murphy, *Hidden Markov Model (HMM) Toolbox for MATLAB*, 1998. [Online]. Available: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
[36] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard, "Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression," *IEEE Robot. Autom. Mag.*, vol. 17, no. 2, pp. 44–54, Jun. 2010.
[37] E. Gribovskaya and A. Billard, "Combining dynamical systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot," in *Proc. 3rd ACME/IEEE Int. Conf. Human-Robot Interact.*, Amsterdam, The Netherlands, Mar. 2008.

Aleksandar Vakanski received the B.Eng. and M.A.Sc. degrees in mechanical engineering from UKIM University, Skopje, Macedonia, in 1998 and 2003, respectively. He is currently working toward the Ph.D. degree in the Department of Mechanical and Industrial Engineering, Ryerson University, Toronto, ON, Canada.

In 2009, he was a Visiting Student at the National Research Council Canada—Institute for Aerospace Research (NRC-IAR), Montreal, QC. His research interests include robot learning, visual servoing, artificial intelligence, and control systems.

Iraj Mantegh received the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 1998 and the M.B.A. degree from Dessautel's School of Management, Montreal, QC, Canada.

He is a Senior Research Officer and a Project Leader at the National Research Council Canada, Montreal. He joined NRC's Institute of Aerospace Research in 2003 and pursued his industrial research interests in the areas of manufacturing process control, robot task planning and learning, and mobile manipulators. He is an Adjunct Professor at Ryerson University, Toronto, and an affiliate of several other Canadian universities.

Andrew Irish received the B.Eng. degree in electrical engineering from McGill University, Montreal, QC, Canada, in 2011. He is currently working toward the M.S. and Ph.D. degrees in electrical engineering (with a concentration on digital communications and signal processing) at the University of California, Santa Barbara.

In 2009–2010, he interned as a Research Assistant with the Department of Robotics and Automation, Aerospace Manufacturing Technology Centre, National Research Council, Montreal.

Farrokh Janabi-Sharifi (S'91–M'95–SM'02) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 1995.

He is a Professor of mechanical–industrial engineering and the Director of the Robotics, Mechatronics and Manufacturing Automation Laboratory (RMAL), Ryerson University, Toronto, ON, Canada. He was an NSERC Postdoctoral Fellow and an Instructor with the Center for Intelligent Machines and Department of Electrical–Computer Engineering, McGill University, Montreal, QC, Canada, from 1995 to 1997. He is a Visiting Professor at the Korea Advanced Institute of Science and Technology, Taejon, Korea; Institut de Recherche en Informatique et Systèmes Aléatoires-Institut National de Recherche en Informatique et en Automatique (IRISA-INRIA), Rennes, France; and Technical University Munich, Munich, Germany. His research interests span over optomechatronic systems with the focus on image-guided control and planning of robots.

Dr. Janabi-Sharifi has also been the organizer and/or coorganizer of several international conferences on optomechatronic systems control. He was the General Chair of the 2010 International Symposium on Optomechatronic Technologies, Toronto. He currently serves as the Associate Editor of several journals including *International Journal of Optomechatronics*.