Taking Advantage of Data Dimensionality Reduction for Dynamic Gesture Recognition from Incomplete Data

Miguel Simão, Pedro Neto, and Olivier Gibaru

***Abstract*—Continuous gesture spotting is a major topic in human-robot interaction research. Human gestures are captured by sensors that provide large amounts of data that can be redundant or incomplete, correlated or uncorrelated. Data dimensionality reduction (DDR) techniques allow to represent such data in a low-dimensional space, making the classification process more efficient. This study demonstrates that DDR can improve the classification accuracy and allows the classification patterns gesture patterns with incomplete data, i.e., with the initial 25%, 50% or 75% of data representing a given dynamic gesture (DG) - time series of positional and hand shape data. Re-sampling with bicubic interpolation and principal component analysis (PCA) were used as DDR methods. Experimental tests indicate that after DDR PCA the classification accuracy is higher with 50% of gesture data (100% accuracy) than with 100% of gesture data (96% accuracy), tested in a library of 10 hand/arm dynamic gestures. Recognized gestures are used to control a robot in an collaborative process.**

***Index Terms*—Gesture recognition, data dimensionality reduction, PCA, human-robot interaction**

I. Introduction

THE ability to interact with a robot in a natural and intuitive way, for example using speech and gestures, has brought important advances to the way our societies look to robots. The paradigm for robot usage has changed in the last few years, from a concept in which robots work with complete autonomy to a scenario in which robots cognitively collaborate with human beings. This brings together the best of each partner, robot and human, by combining the coordination and cognitive capabilities of humans with the robots' accuracy and ability to perform monotonous tasks. For this end, robots and humans have to understand each other and interact in a natural way, creating a co-working partnership. This will allow a greater presence of robots in our companies, schools, hospitals, etc., with consequent positive impact on societys life standards. The current problem is that the existing interaction modalities are neither intuitive nor reliable. Instructing and programming an industrial robot by the traditional teaching method is a tedious and time-consuming task that requires technical expertise.

The robot market is growing and the human-robot interaction (HRI) interfaces will have a main role in the acceptance of robots as co-workers. Gestures and other natural interaction modalities may decrease the need for technical expertise in robot programming, therefore decreasing the cost of owning a robot.

Multimodal HRI interfaces combining gestures, speech and tactile based-actions are expected to be in a near future the standard for a reliable and intuitive interaction process. Nonverbal communication cues in the form of gestures are considered to be an effective way to approach natural HRI. For instance, a person can point to indicate a position to a robot, use a dynamic gesture to instruct a robot to move and a static gesture to stop the robot [1], [2]. In this scenario, the user has little or nothing to learn about the interface, focusing on the task and not on the interaction [3]. For all the reasons mentioned above, continuous and real-time gesture spotting (segmentation and recognition) are key factors to bridge the gap between laboratory research and real world application of novel HRI modalities.

Some gestures, although not all, can be defined by their spatial trajectory. This is particularly true for pantomimic gestures, which are often used to demonstrate a certain motion to be done, e.g., a circle. Burke and Lasenby focused with success on using PCA and Bayesian filtering to classify these time series [4].

The importance of DDR PCA to reduce the dimensionality of a dataset representing human gestures for HRI has been studied in [5] [6]. In a different study, PCA is applied to a continuous stream of time series data capturing body motions with an accuracy of 91% [7]. In [8] it is proposed a unified sparse learning framework by introducing the sparsity or L1-norm learning, which further extends the locally linear embedding (LLE)-based methods to sparse cases. A DDR approach using sparsified singular value decomposition (SSVD) technique to identify and remove trivial features before applying feature selection is proposed in [9]. A reference study presents a sequence kernel dimension reduction approach (S-KDR) in which spatial, temporal and periodic information is combined in a principled manner and an optimal manifold is learned [10].

Gesture classification has been studied over the years. However, there remains the problem with reliability and intuitiveness, which are key factors for a system's acceptance by

Miguel Simão is with the Department of Mechanical Engineering at University of Coimbra, Coimbra, Portugal, and Ecole Nationale Supérieure d'Arts et Métiers, ParisTech, Lille, France e-mail: miguel.simao@uc.pt.

Pedro Neto is with the Department of Mechanical Engineering at University of Coimbra, Coimbra, Portugal, e-mail: pedro.neto@dem.uc.pt.

Olivier Gibaru is with the Ecole Nationale Supérieure d'Arts et Métiers, ParisTech, Lille, France, e-mail: Olivier.GIBARU@ENSAM.EU.

This work was supported in part by the European Comission under Contract number HORIZON2020-688807-ColRobot and the Foundation for Science and Technology (FCT), SFRH/BD/105252/2014.

end-users. Other challenges are related with the ability to perform the recognition in real-time and continuously, recognize gesture patterns with incomplete data, and performing DDR keeping or increasing the recognition accuracy. DDR allows to reduce training data in supervised classification methods and consequently reduce the training time.

To overcome the problem of classification of time sequences, i.e., DGs, we present two approaches to their feature extraction and DDR: one based on up- or down-sampling of gesture frames using interpolation, and another based on PCA. The PCA approach has the advantage of being capable of yielding a good classification even before the gesture is finished – with incomplete/partial gesture data. Complex DGs are defined by a large set of features, with a variable number of frames, and with both trajectory and finger movement. The classification was done with artificial neural networks (ANN).

Independently of the type of gesture, a feature vector for a sample is defined by the vector $\mathbf{z} \in \mathbb{R}^d$:

$$\mathbf{z}^{(i)} = [f_1 \ f_2 \ ... \ f_d] \tag{1}$$

where f_i is the ith element of $\mathbf{z}$.

II. Data Dimensionality Reduction

A. Overview

The dynamic blocks that compose DGs are segmented in an unsupervised way. A segmentation function s based on a motion-threshold algorithm [11] is applied to a stream or set of data S, of dimensionality d and length n. This function is here generically defined by s:

$$\begin{array}{rlll} s: & \mathbb{R}^{d\times n} & \rightarrow \{0,1\}^n \\ & S & \mapsto m \end{array} \tag{2}$$

where the static frames are defined by $m = 0$ and the dynamic are $m = 1$. The dynamic segments are then extracted by a search function that finds transitions in m (from 0 to 1 and 1 to 0). Given two consecutive transitions in the frames i and $i+k$ so that $m_{i-1} = 0$, $m_i = 1$, $m_{i+k-1} = 1$ and $m_{i+k} = 0$, the DG sample is defined by:

$$\mathbf{X}^D = [S_{\bullet i} \ S_{\bullet i+1} \ ... \ S_{\bullet i+k-1}], \quad \mathbf{X}^D \in \mathbb{R}^{d\times k} \tag{3}$$

where the $S_{\bullet i}$ vector is the ith column (frame) of the data stream. In terms of notation, for a generic $\mathbf{A} \in \mathbb{M}_{m\times n}$, $\mathbf{A}_{ij}$ represents the element of the array $\mathbf{A}$ with row i and column j, $\mathbf{A}_{i\bullet} \equiv [\mathbf{A}_{i1} \cdots \mathbf{A}_{in}]$ and $\mathbf{A}_{\bullet j} \equiv [\mathbf{A}_{1j} \cdots \mathbf{A}_{nj}]^T$.

A specific sample i of a data set is represented by $\mathbf{X}^{(i)}$. A function f is then used to extract the feature-vector $\mathbf{z}'$ from each sample (4). This transformation can have multiple steps but the last one always outputs $\mathbf{z}'$. In this work, the number of primes decreases with each transformation step, i.e., $\mathbf{z}''' \rightarrow \mathbf{z}'' \rightarrow \mathbf{z}' \rightarrow \mathbf{z}$. For example, the last step before classification is always scaling, so this step is represented by $\mathbf{z}' \rightarrow \mathbf{z}$.

$$\begin{array}{rlll} f: & \mathbb{R}^{d\times n} & \rightarrow \mathbb{R}^b \\ & \mathbf{X} & \rightarrow \mathbf{z}'' \end{array} \tag{4}$$

in which b is the dimensionality of the feature vector. The vector $\mathbf{z}''$ is the input for the classifiers and $\mathbf{t} \in \{0,1\}^{n_{classes}}$ is the target value of the classifier for that sample (supervised learning). If the target class has the number o, the target vector $\mathbf{t}^{(o)}$ is defined by:

$$\mathbf{t}_j^{(o)} = \delta_{oj}, \ j = 1, ..., n_{classes} \tag{5}$$

where δ is the Kronecker delta and $\mathbf{t}_j$ is the jth element of $\mathbf{t}$.

The transformation f may still yield a vector with very high dimensionality b, which may be detrimental for the classification. Therefore, further processing techniques, such as PCA and interpolation, are proposed for DDR:

$$\begin{array}{rlll} r: & \mathbb{R}^{b''} & \rightarrow \mathbb{R}^{b'} \\ & \mathbf{z}'' & \rightarrow \mathbf{z}' \end{array}, \qquad b'' < b' \tag{6}$$

Nevertheless, DDR can be done either before or after feature extraction, so in this work we consider that the composition of the functions is independent of their order, i.e., $\mathbf{z}' = (f \circ r)(\mathbf{X}) \equiv (r \circ f)(\mathbf{X})$.

B. Re-Sampling with Bicubic Interpolation

Re-sampling is done with bicubic interpolation to transform a DG sample $\mathbf{X}^{(i)}$, $i \in i^D$, $\mathbf{X} \in \mathbb{M}^{d\times n}$, which has a variable number of frames n, into a fixed-dimension sample $\mathbf{X}'$, $\mathbf{X}' \in \mathbb{M}^{d\times k}$. Usually $k \geq n$, being k arbitrarily defined as the maximum n in all the samples i so that $i \in i^D$. So, although in almost every case the proposed transformation is up-sampling the sample, it is also valid for new cases where $k < n$, effectively down-sampling the sample.

$$\begin{array}{rlll} interp: & \Re^{d\times n} & \rightarrow \Re^{d\times k} \\ & \mathbf{X} & \rightarrow \mathbf{X}' \end{array} \tag{7}$$

The bicubic interpolation method [12] yields a surface p described by 3rd order polynomials in both dimensions of space. Given a patch of dimension 2×2, there are 4 data points in which we know the values f and derivatives f_x, f_y and f_{xy}. The derivatives are not known at the boundaries, but they can be estimated using finite differences. The interpolated values inside the uniformized 2×2 sector are given by:

$$p(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij} x^i y^j \tag{8}$$

A representation of the sector is presented in Figure 1.

The problem is determining the 16 coefficients a_{ij}. The function values and 3 derivatives at the 4 points provide $4 \times 4 = 16$ linear equations, which can be written as an equation system $\mathbf{A}\alpha = \mathbf{x}$ with:

$$\alpha = [a_{00} \ a_{10} \ a_{20} \ a_{30} \ a_{01} \ ... \ a_{33}]^T \tag{9}$$

$$\mathbf{x} = [f(0,0) \ f(1,0) \ ... \ f_x(0,0) \ ... \ f_y(0,0) \ ... \ f_{xy}(0,0) \ ... \ f_{xy}(1,1)]^T \tag{10}$$

The matrix $\mathbf{A}$ is nonsingular, so the equation system can be rewritten as $\alpha = \mathbf{A}^{-1}\mathbf{x}$. This process is used for all patches in the bi-dimensional grid. The derivatives at the boundaries of a patch are maintained across neighboring patches. In order to apply the method to the whole data grid efficiently, techniques such as Lagrange polynomials, cubic splines or cubic convolution algorithms are used. The resulting interpolated data points are smother and have less artifacts than those using other interpolation methods, such as bilinear interpolation.

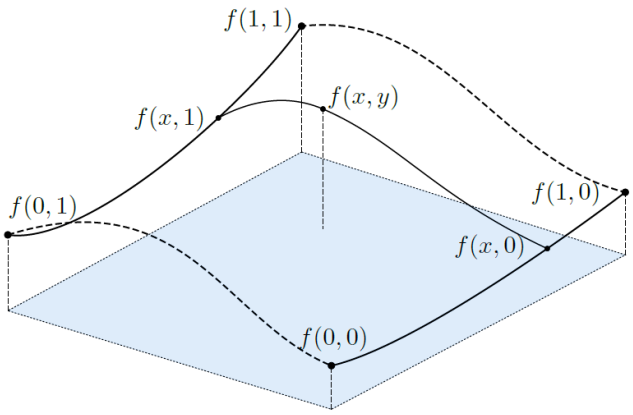

Fig. 1. Representation of the result of bicubic interpolation on a 2×2 grid of points $f(0,0),\ f(1,0),\ f(0,1),\ f(1,1)$.

C. Principal Component analysis

PCA is a mathematical tool that performs an orthogonal linear transformation of a set of n p-dimensional observations, $\mathbf{X} \in \mathbb{R}^{n \times p}$, into a space defined by the PC. The PC have necessarily a size less than or equal to the number of original dimensions, p. The first component has the largest possible variance observed in the observations. Each of the following PC is orthogonal to the preceding component and has the highest variance possible under this orthogonality constraint. The PC are the eigenvectors of the covariance matrix and its eigenvalues are a measure of the variance in each of the PC. Therefore, PCA can be used for reducing the dimensionality of data by projecting that data into the PC space and truncating the lowest-ranked dimensions. These dimensions have the lowest eigenvalues, so truncating them retains most of the variance present in the data.

The first step in PCA is centering the data, because PCA is sensitive relative to the scaling of the original dimensional space. This is done by subtracting each of a dimension's values by its overall average.

The PC transformation is very often determined by another matrix factorization method, the SVD of $\mathbf{X}$:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{11}$$

where $\mathbf{X} \in \mathbb{M}^{n \times p}$ is the original data matrix. $\mathbf{\Sigma} \in \mathbb{M}^{n \times p}$ is a diagonal matrix with the singular values of $\mathbf{X}$, $\mathbf{U}$ is a $n \times n$ matrix whose columns are orthogonal unit vectors that are the left singular vectors of $\mathbf{X}$, and $\mathbf{V} \in \mathbb{M}^{p \times p}$ is a matrix whose columns are unit vectors, the right singular vectors. Both $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, so that $\mathbf{U}^T\mathbf{U} = \mathbf{I}_p$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}_p$. The singular values $\sigma_1, \sigma_2, ...$ in the diagonal of $\mathbf{\Sigma}$ are the positive square roots, $\sigma_i = \sqrt{\lambda_i} > 0$, of the nonzero eigenvalues of the Gram matrix $\mathbf{K} = \mathbf{X}^T\mathbf{X}$, thus being always positive.

The implementation used for this purpose was Matlab's *pca* function. Since the input data matrix $\mathbf{X}$ is most often rectangular, the function uses the aforementioned SVD method (11) for the matrix decomposition. The singular values, i.e., the variance in each of the PC, are the eigenvalues of the covariance matrix of $\mathbf{X}$. The covariance matrix of p sets variates $\{\mathbf{x}_1\}, ..., \{\mathbf{x}_p\}, \mathbf{x}_i = \mathbf{X}_{\bullet i}$ is defined by $\mathbf{W} \in \mathbb{M}^{p \times p}$:

$$\mathbf{W}_{ij} = \mathrm{cov}(\mathbf{x}_i, \mathbf{x}_j) \equiv \langle (\mathbf{x}_i - \mu_i)(\mathbf{x}_j - \mu_j) \rangle, \quad i, j = 1, ..., p \tag{12}$$

where μ and $\langle\ \rangle$ denote mean value, being $\mu_i = \langle \mathbf{x}_i \rangle$. $\mathbf{W}$ can also be written as $\mathbf{W}_{ij} \equiv {}^1\!/_{n-1}\mathbf{X}\mathbf{X}^T$. The product $\mathbf{X}\mathbf{X}^T$ has as eigenvectors the columns of $\mathbf{U}$.

Although PCA is most often performed to reduce the dimensionality of the observations, in this work we preferred to use the PCs as features. The first PC or singular vector $\mathbf{U}_{\bullet 1}$ determines the direction in the PC-space in which there is the most variance during a DG. The variance is measured by the respective singular value, $\mathbf{\Sigma}_{11}$. Therefore, we expect these values to produce good features for the DG classification, even if the gesture is incomplete. We used also PCA to represent gesture features in lower two- and three-dimensional spaces, for easier visualization.

III. Experimental Results

A. Interaction technologies and data acquisition

The proposed system works with any source of positional and hand shape data. For these tests, we used two sensors to capture the hand shape, position and orientation: (1) a data glove; (2) a magnetic tracker.

The data glove was a CyberGlove II, developed by CyberGlove Systems. The version used has 22 resistive bend sensors: three flexion sensors per finger, four abduction sensors, a palm-arch sensor, and two sensors to measure wrist flexion and abduction. The sensors output real-time digital joint-angle filtered data at an average rate of 100 Hz.

The magnetic tracker used was P olhemusLiberty. It has a very low latency and outputs at a rate of 120 Hz.

The glove transmits the data to through a Bluetooth connection and the tracker is connected by a physical serial connection. Serial objects read the available data in windows of 10 samples and store the available data with timestamps in buffers. A script then reads the k newest samples from both the buffers and processes them. A full frame of data with the 22 DOF from the glove and 6 DOF from the tracker is represented by $\mathbf{f}$ (13), where g_k represents the kth DOF of the glove and l_k represents the kth DOF of the tracker. Before further processing, the stream of data is segmented by a motion-threshold method [11]. The method appends to

the frame a binary segmentation variable m which represents whether the frame belongs to a dynamic segment or not.

$$\mathbf{f} = [g_1\ g_2\ ...\ g_{22}\ l_1\ l_2\ ...\ l_6\ m] \tag{13}$$

B. Dataset

A sample in the dataset is represented by ($\mathcal{S}^D$):

$$\mathcal{S}^D = \left\{\mathbf{X^{(i)}}, t^{(i)}\right\},\ \mathbf{X}^{(i)} \in \mathbb{R}^{d\times n},\ t^{(i)} \in \left\{1, ..., L^D\right\} \tag{14}$$

where d is the number of DOF of the system, n is the number of data frames in the sample, t is the target class and L^D is the number of classes for DG.

A total of 10 DGs combining hand/arm and fingers motion were selected, Figure 2. For testing purposes, a total of 20 samples were acquired from only one subject. This sums up to 200 gesture samples.

C. Features

To ensure reliable classification independently of the subject position and orientation, every gesture sample has its feature data reported to their local reference frame. A coordinate transformation t is applied to the features vector:

$$\mathbf{X}'^{(i)}{=}t(\mathbf{X}^{(i)}), \quad i \in i^D, \quad \mathbf{X}^{(i)} \in \mathbb{M}^{28\times n}(\mathbb{R}) \tag{15}$$

Following that, two distinct feature extraction methods from $\mathbf{X}'$ are proposed:

1) Re-sampling the samples with bicubic interpolation (DG1);
2) Extracting principal vectors and values using PCA (DG2).

In the first case, DG1, given a sample $\mathbf{X}^{(i)} : i \in i^D$ with n frames ($\mathbf{X}^{(i)} \in \mathbb{M}^{28\times n}$), the objective is to resample it to a fixed size n'. The number n' can be chosen arbitrary but in order to retain as many of the gesture features as possible, n' should be an upper bound such that $n' \geq n,\ \forall n|\mathbf{X}^{(\mathbf{i}^D)} \in \mathbb{M}^{28\times n}$. Therefore, we opted by choosing the highest n of the the DG samples, specifically $n' = 161$. Applying the bicubic interpolation algorithm, the result is a matrix $\mathbf{Z} \in \mathbb{R}^{28\times 161}$. The following step is to transform $\mathbf{Z}$ into a vector $\mathbf{z} \in \mathbb{R}^{4508}$, which is done by concatenating every frame vertically:

$$\mathbf{z}^{(i)} = \text{concat}(\mathbf{Z}^{(i)}) = \begin{pmatrix} \mathbf{Z}^{(i)}_{\bullet 1} \\ \vdots \\ \mathbf{Z}^{(i)}_{\bullet 161} \end{pmatrix} \tag{16}$$

In a second case, DG2, we use PCA to extract features. The advantage is that it allows us to obtain features from incomplete gestures and still obtain coherent features. From each sample $\mathbf{X}^{(i)} : i \in i^D,\ \mathbf{X} \in \mathbb{M}^{28\times n}$ are extracted 4 feature vectors $\mathbf{z}^{(i)}_k : k \in \{0.25, 0.50, 0.75, 1\}$, where k defines the fraction of the number of frames that were used:

$$\mathbf{z}^{(i)}_k = \begin{pmatrix} \mathbf{U}_{\bullet 1} \\ \mathbf{\Sigma}_{11} \end{pmatrix} \tag{17}$$

where $\mathbf{U}_{\bullet 1}$ and $\mathbf{\Sigma}_{11}$ are the first singular vector and value, respectively. The singular vector has 28 dimensions and the value just 1 dimension. They are calculated using the the partial sample $\mathbf{X}^{(i)}_{\bullet \mathbf{m}}$, so that:

$$\text{pca}(\mathbf{X}^{(i)}_{\bullet \mathbf{m}}),\ \mathbf{m} = \{1, ..., \lceil nk \rceil\} \tag{18}$$

where $\lceil nk \rceil$ represents the ceiling function, since $\lceil nk \rceil \in \mathbb{N}$.

The last feature processing step is feature scaling. Feature scaling is essential for achieving smaller training times and better network performance with less samples. It harmonizes the values of different features so that all of them fall within the same range. This is especially important when some features have distinct orders of magnitude. The scaling function chosen was linear rescaling, l:

$$l(\mathbf{x}) = \frac{2\mathbf{x} - \widehat{\mathbf{X}}^T}{\widehat{\mathbf{X}}^T} \tag{19}$$

where $\widehat{\ }$ is the max+min operator defined in (20). $\mathbf{X}^T = \left(\cup \mathbf{z}^{(i)} :\ i \in i^T\right)$ is the set of unscaled features of the training set. This operator is valid both for static and dynamic gestures but the sample subsets used should be exclusive.

$$\widehat{\mathbf{X}}_i = \max \mathbf{X}_{i\bullet} + \min \mathbf{X}_{i\bullet}, \quad i = 1, ..., d \tag{20}$$

D. Results and discussion

The available samples $\mathcal{S}(i) :\ i \in \mathbf{i}^D$ were randomly divided in two sets – a training set ($i \in \mathbf{i}^{DT}$), and a validation set ($i \in \mathbf{i}^{DV}$). In this case, we have 20 samples per each of the 10 classes but only from one subject. Therefore, each set has 10 samples/class.

The ANN architecture is composed by one hidden layer with 25 nodes and 10 output neurons (classes) in both approaches, as shown in Figure 3. The difference between DG1 and DG2 is the size of the input feature vector, 4508 and 29 for DG1 (16) and DG2 (17), respectively. In both cases the transfer function is the hyperbolic tangent in the first layer and the *softmax* function in the second layer.

The classification results are displayed in Table I. For DG1, the accuracy in the validation data set was 99.0% (99/100), where the only error was gesture 7 being classified as 8. This can be seen in the confusion table in Figure 5 (a). In Figure 4 it is possible to see the distribution of the features in a reduced principal component space (first and third PCs) for DG1. In this $\mathbb{R}^2$ space, the classes show good separability, being gesture 7 the exception, where higher than normal dispersion can be seen. One of the samples representatives of class 7 fell into the middle of the class 8 cluster, which led to it being miss-classified. There are other very close clusters, such as the gestures D1, D2, D4 and D5, but they have very low dispersion and show better separability when seen in higher spaces.

While for the DG1 case the gesture frames are interpolated after the gesture is finished, for DG2 the features can be extracted at any time during a gesture. This allows for recognition even before the gesture is even finished. Given that, the network DG2 was trained and validated with four sets of features originated from $\mathbf{i}^{DT}$, $\mathbf{z}^{(i)}_k : k \in \{0.25, 0.50, 0.75, 1\}$,

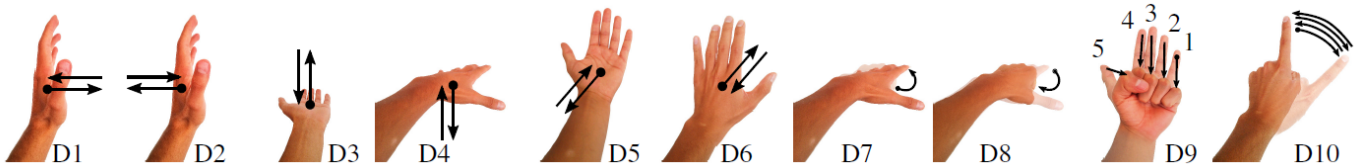

Fig. 2. Representations of the 10 DGs.

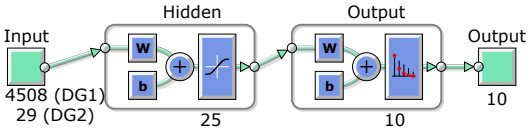

Fig. 3. ANN architecture used for DG classification using interpolated frames (DG1) and the first principal component (DG2) as features.

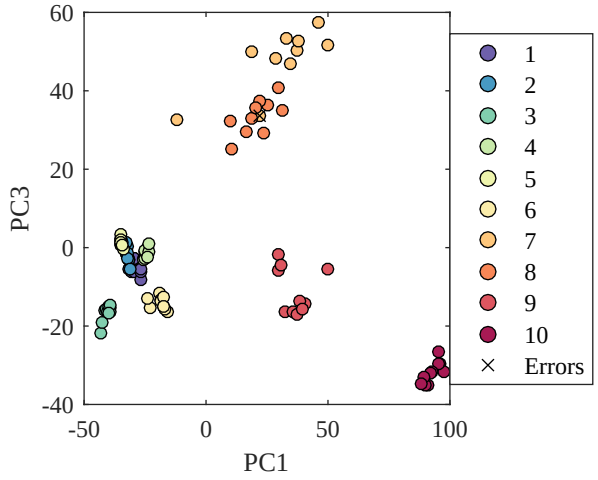

Fig. 4. Distribution of the validation samples $\mathbf{i}^{DV}$ for the test case DG1 in the PC space. Each color discriminates a class and the sample that originated the error is marked with an $\times$.

see (17). Therefore, the training and validation sets had each 40 feature vectors per gesture. The network has the same configuration as in DG1 except for the number of input nodes which is 29, Figure 3.

The accuracy results in the validation data set given this configuration are shown in Table I on page 5 for each of the 4 sets of features. With only the initial 25% of the gesture, the accuracy was 95% and increased to 100% when half of the gesture data was used. Nevertheless, when 75 and 100% of the available data is available, the accuracy decreased to 98% and 96%, respectively. The confusion table in Figure 5 shows that the gesture with the most errors was gesture 7, being classified three times as 8 and two times as 9. All these gestures are short in duration (less data) and are similar, e.g., the starting and ending poses of 7 and 9 are the same (closed hand), the difference being in the order the fingers are closed.

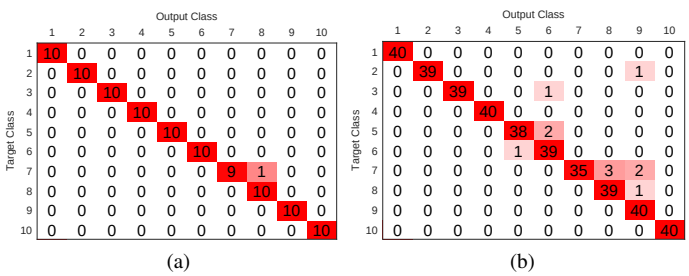

Fig. 5. Confusion table for the classification of the DG samples from the validation set $\mathbf{i}^{DV}$ using both approaches: (a) DG1, (b) DG2.

TABLE I
FINAL RESULTS FOR THE CLASSIFICATION ACCURACY ON THE VALIDATION DATA SETS.

DG1	DG2 (0.25)	DG2 (0.5)	DG2 (0.75)	DG2 (1.0)
99.00	95.00	100.00	98.00	96.00

It is possible to see the evolution of the features obtained from 25% to 100% of the data in Figure 6. Even at 25% there is already good separation of the classes, although the dispersion is still high, compared to the later stages. As more data is available, the dispersion decreases and the feature samples form defined intra-class clusters. In the case of gestures D6, D7 and D8 this also happens, but the clusters partially intercept, causing the decrease of the accuracy rate. This also means that the selected features are not ideal for the classification of these three gestures.

In both cases DG1 and DG2 the features for the classes D1 through D6 have well defined albeit close clusters, sometimes generating errors. This happens because there is very little inter-gesture variation of most of the variables during these gestures. They are distinguished by path and hand orientation, which are defined by 6 out of the 28 DOFs of the system. The classification accuracy remained high for these samples, but in the future these gestures should be represented by better features, e.g, trajectory direction, in order to improve the robustness of the classification.

E. Interacting with a real robot

We developed a HRI interface with an industrial robot using the classified gestures as input. The attempted task was preparing a breakfast meal, composed of grabbing a cereal box, pouring the contents into a bowl, grabbing a yogurt bottle and also pouring its contents into the same bowl, Figure 7. The output gestures were mapped to robot actions for motion and other tasks, i.e., open/close gripper.

IV. CONCLUSIONS

This paper demonstrated that dynamic gesture data can be subject to DDR making the classification process more

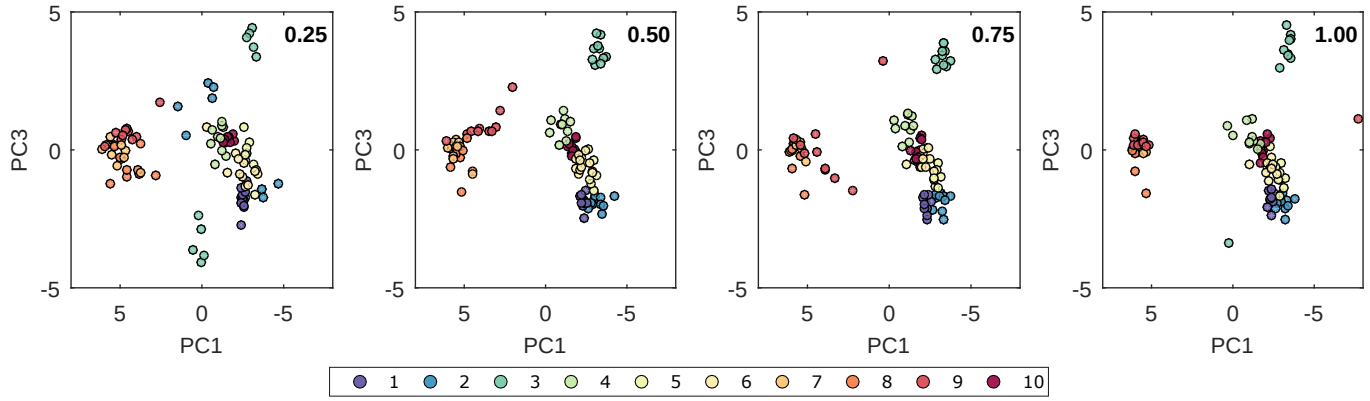

Fig. 6. Plots of the features obtained from the validation data set (including the sets with incomplete data – 25%, 50%, 75% and 100% of the data) in a reduced principal component space. The features were centered and scaled. Each color represents a different class.

Fig. 7. Visualization of different stages of a teleoperation HRI process: (a) starting point, (b) virtual joystick guidance to a goal, (c) forceful stop command, (d) rotation of the end-effector, (e) gesture-command to open the gripper, (f) grabbing a bottle and puling it up, (g) rotation of the end-effector, (h) safe collaboration with the robot. NOTE: The virtual joystick mode moves the end-effector in a direction defined by the vector that joins a center position in which the hand is closed and the position of the hand when it is moved.

efficient. DDR PCA can improve the classification accuracy and allows to classify gesture patterns with incomplete data, i.e., with the initial 25%, 50% or 75% of data representing a given dynamic gesture (DG) - time series data. Experimental tests indicate that after DDR PCA the classification accuracy is higher with 50% of gesture data (100% accuracy) than with 100% of gesture data (96% accuracy), tested in a library of 10 hand/arm dynamic gestures. Recognized hand/arm gestures proved to be a natural and intuitive HRI interface.

References

[1] P. Neto, D. Pereira, J. N. Pires, and a. P. Moreira, "Real-time and continuous hand gesture spotting: An approach based on artificial neural networks," *2013 IEEE International Conference on Robotics and Automation*, pp. 178–183, 2013.

[2] M. T. Wolf, C. Assad, M. T. Vernacchia, J. Fromm, and H. L. Jethani, "Gesture-based robot control with variable autonomy from the JPL BioSleeve," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, may 2013, pp. 1160–1165.

[3] Y. Song, D. Demirdjian, and R. Davis, "Continuous body and hand gesture recognition for natural human-computer interaction," *ACM Transactions on Interactive Intelligent Systems*, vol. 2, no. 1, pp. 1–28, mar 2012.

[4] M. Burke and J. Lasenby, "Pantomimic gestures for human–robot interaction," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1225–1237, Oct 2015.

[5] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, March 2007, pp. 255–262.

[6] D. Kulic, W. Takano, and Y. Nakamura, "Online segmentation and clustering from continuous observation of whole body motions," *IEEE Transactions on Robotics*, vol. 25, no. 5, pp. 1158–1166, Oct 2009.

[7] J. F. S. Lin, V. Joukov, and D. Kulic, "Full-body multi-primitive segmentation using classifiers," in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 874–880.

[8] Z. Lai, W. K. Wong, Y. Xu, J. Yang, and D. Zhang, "Approximate orthogonal sparse embedding for dimensionality reduction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 723–735, April 2016.

[9] P. Lin, J. Zhang, and R. An, "Data dimensionality reduction approach to improve feature selection performance using sparsified svd," in *2014 International Joint Conference on Neural Networks (IJCNN)*, July 2014, pp. 1393–1400.

[10] A. Shyr, R. Urtasun, and M. I. Jordan, "Sufficient dimension reduction for visual sequence classification," in *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, 2010, pp. 3610–3617.

[11] M. Simão, P. Neto, and O. Gibaru, "Unsupervised gesture segmentation by motion detection on a real-time data stream," *Submitted to IEEE Transactions on Industrial Informatics*, 2016.

[12] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, Dec 1981.