Spatial and rotation invariant 3D gesture recognition based on sparse representation

Ferran Argelaguet Sanz, Mélanie Ducoffe, Anatole Lécuyer, Rémi Gribonval

Spatial and Rotation Invariant 3D Gesture Recognition Based on Sparse Representation

Ferran Argelaguet* Inria, IRISA, France
Mélanie Ducoffe† ENS Rennes, France
Anatole Lécuyer‡ Inria, IRISA, France
Remi Gribonval§ Inria, IRISA, France

ABSTRACT

Advances in motion tracking technology, especially for commodity hardware, still require robust 3D gesture recognition in order to fully exploit the benefits of natural user interfaces. In this paper, we introduce a novel 3D gesture recognition algorithm based on the sparse representation of 3D human motion. The sparse representation of human motion provides a set of features that can be used to efficiently classify gestures in real-time. Compared to existing gesture recognition systems, sparse representation, the proposed approach enables full spatial and rotation invariance and provides high tolerance to noise. Moreover, the proposed classification scheme takes into account the inter-user variability which increases gesture classification accuracy in user-independent scenarios. We validated our approach with existing motion databases for gestural interaction and performed a user evaluation with naive subjects to show its robustness to arbitrarily defined gestures. The results showed that our classification scheme has high classification accuracy for user-independent scenarios even with users who have different handedness. We believe that sparse representation of human motion will pave the way for a new generation of 3D gesture recognition systems in order to fully open the potential of natural user interfaces.

Index Terms: I.5.2 [Pattern Recognition]: Design Methodology—Classifier design and evaluation; I.6.3 [Computing Methodologies]: Methodologies and Techniques—Interaction Techniques

1 INTRODUCTION

In recent years, the number of tracking technologies has been continuously rising, from commodity hardware devices such as the Microsoft's Kinect, the Leap Motion or the HTC Vive to high-end commercial solutions relying on optical or magnetic tracking [21]. Yet, the majority of virtual reality applications still rely on basic interaction capabilities such as pick rays or virtual hand metaphors, limiting the interaction capabilities of the user. In contrast, natural user interfaces (NUI) exploiting multi-modal input [20] could enable the creation of a new generation of user interfaces that increase the expressiveness of the user interaction. Gestural interfaces are a good candidate for NUIs as they enable the use of the whole body as an input device and have a wide range of potential application scenarios such as medical [36, 1], interaction with robots [16, 9] or entertainment [13, 39].

A gesture can be considered as a meaningful and intentional movement performed by the user. It can have a meaning by itself (e.g. push to move an object), or it can represent a predefined command by the system (e.g. performing a cross to close the application). Gestures can be used to enable natural interactions with virtual environments, by exploiting the affordances and the previous experience of users. However, there are a number of other factors that interface designers have to take into account in order to ensure that interfaces based on gestural interaction are usable. For example, how to make the user aware of the available interactions, the gestures different users prefer [17], and the number of gestures a user can remember [15].

Nevertheless, the enabling component of gestural interfaces is efficient and robust gesture recognition algorithms. These algorithms have to deal with two major challenges: gesture variability and gesture dimensionality. The same gesture performed by different users, or even the same user performing it twice, will not generate the exact same gestural information (e.g. trajectory, acceleration profile). Biomechanical limitations of the body (neuromuscular noise) will introduce variability on the user motion, generating slightly different versions of the same gesture. Moreover, the motion information contained in a gesture is temporal and lies in a high-dimensional space. The analysis and classification of gestures can be computationally expensive and can discourage its usage in interactive applications with low end-to-end latency requirements. In order to address both challenges, existing methods rely on subsampling [33] and on machine learning algorithms such as Hidden Markov Models [14, 31], linear classifiers [13] or template matching algorithms [18]. Although existing methods are able to achieve high classification accuracies (up to 90%), they also have a tradeoff between complexity, performance and recognition accuracy. For example, they typically consider a small amount of gestures and one-handed interactions [24].

In this work, we propose a novel 3D gesture recognition algorithm based on the sparse representation of 3D human motion. The proposed algorithm is able to extract the most representative motion patterns from a given set of gestures and uses this information to enable the real-time classification of 3D gestures. Our approach, inspired by state of the art 3D sparse representation algorithms [4], provides a robust gesture classifier, that is tolerant to the speed, the scale and the rotation of the gesture. Furthermore, the sparse representation of human motion is robust to noise (only salient features are kept) and generates a compact representation. To summarize, the main contributions of our work are:

- A novel approach for 3D gestures recognition based on sparse representation. Enclosing: (1) a gesture pre-processing step to ensure that the input data is well suited for the sparse representation and (2) a 3D sparse coding optimization enabling real-time gesture classification of 3D human motion data.
- The evaluation of the proposed gesture recognition algorithm in two stages: (1) an off-line comparison with existing techniques and motion databases, and (2) a real-time evaluation with naive users in a virtual reality system.

The paper is structured as follows: Section 2 discusses related works on gesture recognition and sparse representation. Section 3 details our novel approach for gesture recognition. Then, Section 4 presents the evaluation results. Finally, Section 5 provides the concluding remarks and future perspectives.

*e-mail:ferran.argelaguet@inria.fr
†e-mail:melanie.ducoffe@ens-rennes.fr
‡e-mail:anatole.lecuyer@inria.fr
§e-mail:remi.gribonval@inria.fr

2 Related Work

2.1 Gesture Recognition

Gestural interaction is gaining more and more relevance in common user interfaces from 2D (e.g. tactile based) to 3D (e.g. kinect-based) interfaces [13], yet gesture recognition is a missing feature in existing 3DUI frameworks [32]. Mitra et al. [27] characterized the information contained in a gesture in four different dimensions: spatial, pathic, symbolic and affective. Traditionally, gesture recognition algorithms for 3DUIs have been mainly focused in the spatial (where it occurs) and in the pathic information (the trajectory). Symbolic information can be extracted once the gesture has been recognized using the context and the pathic information. Regarding the affective information, existing works have mainly focused on vision-based face and body gestures recognition [27]. In this work we only focus on gesture recognition from 3D motion tracking data, which is the most common for 3DUIs.

Current gesture recognition algorithms take advantage of existing machine learning algorithms to build knowledge about the different gestures that can be used in the system. In this paper, we will just review the most salient gesture recognition algorithms, focusing on feature-based and template matching classifiers. For a complete survey on 3D gesture recognition we refer the reader to the survey from LaViola [21].

Feature-based classifiers transform the input data into a feature vector which is used to classify the input gesture. The first problem is how to express the characterization of different gestures in a set of distinctive features. Chen et al. [7] defined a set of 45 features which can be used for classification (e.g. mean speed, mean curvature or the bounding volume of the gesture). Depending on the target set of gestures, just a subset of features can be used. These features provide mainly spatial and pathic information. However, such features "simplify" the input data potentially loosing the gestural information and the notion of distance between two gestures. Once the set of features is decided, a classification algorithm should be chosen. The most common approaches are based on Hidden Markov Models (HMM), used for the recognition of sign gestures [14], interaction with mobile devices [30] or video games [31, 13, 7]. Additional works have used other machine learning approaches such as linear classifiers [13], nearest-neighbors classifiers [19], support-vector machines [17] or bayesian and AdaBoost classifiers [5]. In average, although the previous cited solutions provide average recognition rates around 90%, they are dependent on the quality of the gesture database and are sensible to changes in rotation, position and noise.

Template matching classifiers consider the information from the pathic information of the gesture and search for a gesture in the training data set which minimizes a given distance function. The most common approach is to compute the mean square error (euclidean distance) between the gesture to recognize against the different gestures in the vocabulary [37, 18]. Additionally, other distance metrics can be employed such as the angular inverse cosine distance [22] or dynamic time warping [28, 24]. Most of them use a nearest-neighbors classifier in order to determine the class of the performed gesture, although there are no strong differences in classification performance between the different distance metrics [35]. Such approaches enable a fast gesture recognition at the expense of lower classification rates than feature-based classifiers. Still, template matching is mostly used for 2D gesture recognition, although it has been explored for 3D gesture recognition [18, 34].

2.2 Normalization and Resampling

Gesture variability and gesture dimensionality are two of the main challenges when designing real-time gesture classifiers. Gesture variability arises due to differences in the equipment (accuracy, calibration) and in the nature of the gesture on itself. The same user will not be able to perform exactly the same gesture twice and the differences are even more noticeable among gestures performed by different users. Although the gesture classification algorithms tolerate a certain variability, most of them include a normalization [14, 7] and/or a filtering step [31] during the preprocessing step. Existing approaches are simple in nature, they are based on the normalization of the input signal considering the range of the input data and the physical limits. Other solutions rely on removing the noise of the input signal by applying linear filters [31] or a uniform resampling of the trajectory [37].

Orthogonally, subsampling methods can also be employed to speed up the recognition process (decrease the dimensionality). Vatavu [34] presented an exhaustive evaluation on how different gesture recognition systems behave when considering subsampled version of the input gestural data. In his analysis, Vatavu considered the subsampling both in terms of dimensionality (samples per gesture) and cardinality (bits per sample). The results showed that the sampling rate and bit cardinality could be decreased (up to 16 sample points and 3 bits per gesture).

2.3 Sparse Representation

Sparse representation allows to express an input signal (y) through a linear combination of an overcomplete basis (the dictionary, Φ). The reconstruction (x) (see Equation 1) should minimize the error ε with the original signal (y). A sparsity constraint (C) ensures that the vector of sparse coefficients x has few elements different from zero. We will refer to the elements of the dictionary as atoms (ϕ).

$$y = \Phi x + \varepsilon = \sum_{m=1}^{M} x_m \phi_m + \varepsilon \;\; s.t \;\; |\, x \,|_0 \leq C \tag{1}$$

Assuming that the dictionary is known, the sparse representation (Φx) of the input signal (y) can be computed using greedy algorithms such as matching pursuit (MP) [26] or orthogonal matching pursuit (OMP) [8, 10]. Both algorithms, in an iterative process, will select the atoms which minimize the residual error (ε). In addition to the reconstruction algorithms, a dictionary learning algorithm (DLA) is required to compute the dictionary (Φ) which minimizes the representation error of a set of representative signals (training dataset). One example is the K-SVD algorithm [2], which initializes the dictionary randomly and updates it with a sequence of a two step iteration: for each signal in the training dataset, it builds its sparse representation and given that representation updates the atoms based on a gradient descent to minimize ε. Sparse representation has been successfully applied to a wide variety of research fields such as face recognition [38] of action recognition of video streams [12].

2.3.1 Sparse Representation of Human Motion

Sparse representation is convenient to handle the sparsity inherent to human motion which can be spatially invariant (different actions can share a similar movement) and temporally invariant (the same action can be performed at different times) [4]. Human motion can be recorded as time series of position, orientation, speed or acceleration data [13] from one or multiple joints of the human body. The majority of previous works focus on "lossy" motion compression, which is one of the major applications of sparse representation. In such context, the motion database is encoded in a dictionary while each motion sample is associated with a small list of atoms and their corresponding parameters. The compression is achieved with matching pursuit algorithms by minimizing ε.

Several representations have been proposed towards this aim. Zhu et al. [41] proposed a quaternion dictionary in which actions are divided into different segments during a preprocessing step and each segment is then represented by a quaternion time series. In contrast, instead of dividing an action during preprocessing, Li et

al. [23] proposed a convolutional dictionary. Atoms in the dictionary are times series whose duration is much shorter than the duration of any action so they can be shifted along time using a convolution operator. Other solutions can also consider wavelets [40]. Finally, in order to account for rotational variations, Barthelemy et al. [4] introduced a discrete time and 3D Rotation Invariant (3DRI) sparse representation, which allows the encoding of multivariate signals, mainly for 3D human motion data. The 3DRO model (see Equation 2) includes a rotation component (R) which accounts for variations in rotation for each atom. The rotation invariance is achieved by an optimization using the single value decomposition (SVD) algorithm. We refer to [4] for further details.

$$y = \sum_{m=1}^{M} x_m R_m \phi_{m,\tau_m} + \varepsilon \;\; s.t \;\; |x|_0 \leq C \tag{2}$$

2.3.2 Sparse Representation for 3D gesture recognition

As sparse representation is tolerant to the noise from the initial time series, a number of the works regarding human motion compression have been extended to use sparse representation for classification purposes. Nevertheless, sparse representation for classification has been mainly studied for 2D gesture recognition such as hand gestures in a video stream [29, 11, 23]. However, according to the representation used (huge number of atoms in the dictionary) such approaches require an enormous amount of training data to ensure a good representation accuracy. To sum up, sparse representation [4] seems a good candidate for the recognition of human gestures, however there is lack of works focusing on the recognition of 3D gestures. In this work we want to fill this gap proposing a gesture classification algorithm based on 3D sparse representation of human motion.

3 Gesture Recognition Framework

In this work we propose the characterization and classification of 3D gestures using sparse representation. The proposed method enables the use of the sparse components as features and provides an error measure through the computation of the representation error (ε). Our approach takes advantage of the invariances from the 3D rotation invariant sparse representation [4] in order to provide a gesture recognition which tolerates changes in the scale and the rotation of the input gesture.

The recognition algorithm has two phases: (1) an off-line training phase and (2) an on-line classification phase (see Figure 1). In the training phase (see Section 3.3), the dictionary (Φ) which minimizes ε of a training dataset is computed and an additional kNN classifier is trained based on the sparse representation of the training dataset. The on-line classification phase (see Section 3.4) computes the sparse representation of an input gesture using (Φ) and uses its sparse representation to determine its gesture class.

3.1 Requirements

The main purpose of gesture recognizers is to discriminate between classes of gestures. Typically, gesture recognition applications consider a finite (reduced) number of classes. However, gesture recognizers must define a set of invariances to determine when two gestures are considered equivalent. In our representation, we consider that two gestures are equivalent if they share a common trajectory (a straight line or a circle) with no regards of the following properties of the gesture:

1. **Position**. The location in space. Two users executing the same gesture (e.g. straight line) at different locations perform equivalent gestures.

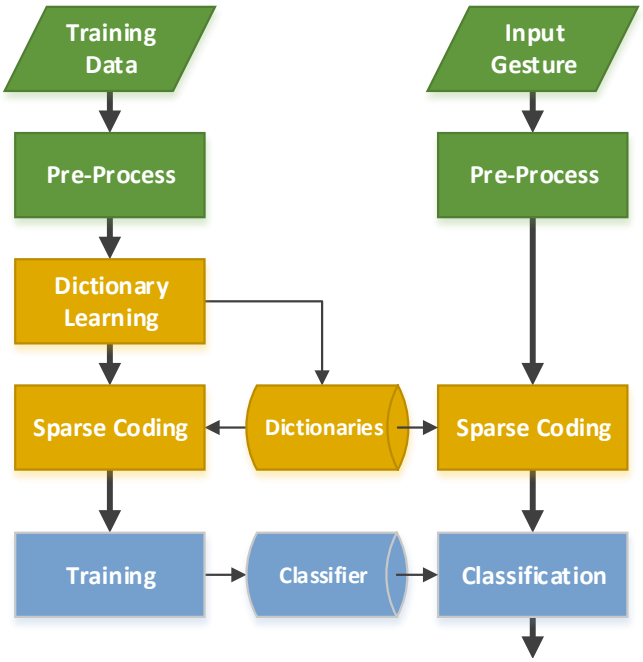

Figure 1: Gesture recognition pipeline. First, gestures are normalized in a pre-processing step. During the off-line training phase, sparse dictionaries are built and the classification algorithms are trained. The real-time classification uses the sparse dictionaries and the trained classifiers in order to classify the input gesture.

2. **Scale**. The dimension of the gesture. The same gesture (e.g. drawing a circle in the air) performed by two users with different arm lengths (e.g. child, adult) are equivalent gestures.

3. **Rotation**. The orientation of the gesture. The same gesture with different orientations in space (e.g. circle in the XY plane and a circle in the YZ plane) are equivalent gestures.

4. **Speed**. The speed over time. Two gestures performed at different speeds are equivalent gestures.

5. **Time**. The interval of time in which the gesture is performed. Two gestures performed at different moments are equivalent gestures.

Depending on the recognition algorithm, differences in these parameters might alter the classification result. In our algorithm, scale and rotation are directly handled by the 3DRI representation (see Section 3.3) while position and speed invariances are handled with the pre-processing step (see Section 3.2).

Finally, gesture recognizers must be tolerant to noise. The noise can be due to the bad quality of the motion data (tracking system) or the intra and inter-user differences. On the one hand, the same user will not be able to execute two times the exact same movement, a non-negligible variability will be introduced for each repetition. On the other hand, different users might perform gestures with slightly different features (see Figure 2). Such variability introduces uncertainty during the classification process which might lead to wrong classifications. In our algorithm, the sparse representation will not be affected by the noise, as the noise will just contribute to the representation error ε. Regarding the user variability, it can be handled by increasing the redundancy of the dictionary, the same gesture can be described by multiple atoms (ϕ).

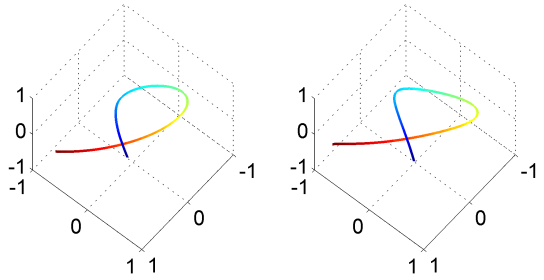

Figure 2: Two examples of gestures for an X-shape gesture. Different users might perform the same gesture with slight variations. The color of the path determines the time for each sample (from colder to warmer).

3.2 Gesture Representation and Pre-processing

For each tracked joint implied in the movement (e.g. the hand trajectory), we consider its trajectory in a fixed orthonormal basis. The trajectory for each joint is discretized and stored as a matrix $\mathbb{R}^{3\times N}$, being N the number of points. In order to decrease gesture variability and just focus on the pathic information, before computing the sparse representation of a given gesture, it is pre-processed in order to ensure speed and position invariances: **(Speed)** The trajectory is converted into a speed invariant domain by expressing the temporal abscissa as a curvi-linear abscissa. First the instantaneous speed for each sampling point is computed. Then, the accumulated speed is divided by the number of points (n) used for resampling, so to obtain the speed increment (I) between two successive points. The gesture trajectory is then stepped through linear interpolation. **(Position)**. The gesture is centered by substrating its centroid in order to decrease the gesture misalignment (position invariance).

We apply the preprocessing operators sequentially: first the speed, then the position. The pre-processing did not include any noise removal step as the sparse coding algorithm used is not affected by high-frequency noise. As the noise is not representative, it will be smoothed during the optimization process (see Section 3.3.1).

3.3 Sparse Model of 3D Gestures

The main criteria for choosing the sparse model is driven by the need to ensure the invariances that are not satisfied in the preprocessing step (rotation, delay and scale) and to avoid noise in the sparse representation. The chosen sparse model considered is the 3DRI model [4] in which rotation and scale variations are accounted by the scale ($x \in \mathbb{R}^*$) and the rotation ($R \in SO_3(\mathbb{R})$) parameters (see Equation 2). Moreover, as the atom can have a different number of samples than the input gesture, a shift operator (τ) is required to determine the temporal position of the atom (ϕ_{i,τ_i}). In other words, whereas the atom is associated to the action, the shifted atom (pattern) takes in account its position along the gesture. So, if we take a look at the physical meaning of a sparse component $\{x_i, R_i, \phi_{i,\tau_i}\}$, it represents a sub-gesture which fits well with a part of the original gesture and the addition of all sub-gestures will result in the original one. The number of atoms in the decomposition (M) is constrained so not to represent the noise inherent to the motion capture system and to ensure real-time recognition. Thus, the problem to solve is the computation of a set of representative atoms (Φ) which minimizes the reconstruction error given a training dataset. In our case we consider the L2-norm to compute the error. We will first detail the algorithm (see Algorithm 1) used to compute the sparse representation of a given gesture when the dictionary is known. Then, the dictionary learning algorithm, which is described in section 3.3.2, will use the same algorithm during the learning phase.

Algorithm 1 3DRI Matching Pursuit. Computes M optimal sparse coefficients which minimize the representation error of the sparse decomposition of the gesture y using the dictionary (Φ).

1: **procedure** $(\phi_\tau^M, x^M, R^M, \varepsilon) \leftarrow$ `3DRI_MP`(y, Φ, M)
2: $\quad\varepsilon = y$
3: $\quad$**for** m$\leftarrow 1, M$ **do**
4: $\quad\quad$**for** k $\leftarrow 1, K$ **do**
5: $\quad\quad\quad$**for** $\tau_k \leftarrow 1, \tau_{max}$ **do**
6: $\quad\quad\quad\quad(x_k, R_k) \leftarrow$`SVD_Adjust`$(\varepsilon, \phi_{k,\tau_k})$
7: $\quad\quad\quad$**end for**
8: $\quad\quad$**end for**
9: $\quad\quad(x_m, R_m, \phi_{m,\tau_m}) \leftarrow argmax_{x\in\mathbb{R}}(x_k)$
10: $\quad\quad(x^m, R^m) \leftarrow$ `3DRI_OPT`$(\varepsilon, y, x^n, R^n, \phi^n_{\tau_n})$
11: $\quad\quad\varepsilon \leftarrow$y$-\sum_{n=1}^{m} x_n R_n \phi_{n,\tau_n}$
12: $\quad$**end for**
13: **end procedure**

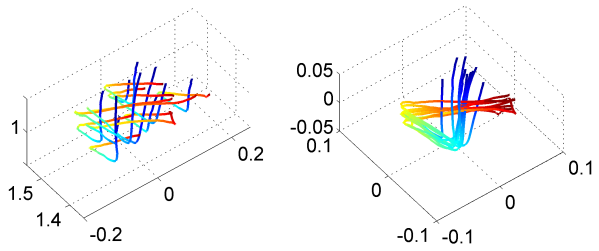

Figure 3: Illustration of the preprocessing procedure. Left, raw gestures. Right, preprocessed samples after applying the curvi-linear abscissa transformation and the centering.

3.3.1 Sparse representation of 3D Gestures

The 3DRI matching pursuit algorithm (see Algorithm 1) computes the M patterns (ϕ_τ^M) which minimize the representation error (ε) of the input gesture (y) given a dictionary (Φ). For each pattern, the algorithm also computes the associated scale factors (x^M) and the rotation matrices (R^M) which minimize the reconstruction error (see Equation 2). The algorithm at each iteration chooses, in a greedy way, the pattern which minimizes the representation error (line 9) after computing the better fit for each atom (line 6). Finally, it optimizes the sparse components that have been previously computed (line 10) and computes the current representation error (line 11). The SVD optimization algorithm is used to fit each atom (see Algorithm 3). The reconstruction algorithm is inspired by the work of Barthelemy et al.[4]. Note that the optimization criterion when selecting the atoms to improve the convergence speed has been modified: we apply iteratively the Procruste methods (see Algorithm 2) to update the parameters of one shifted atom given the other. When it comes to the optimization criterion as we are rotating the atoms, the orthogonal projection is no longer possible, so our method is denoted with the matching pursuit (MP) appellation rather than orthogonal matching pursuit (OMP). Furthermore, the number of sparse components are constraint to ensure that all sparse representations share the same amount of components.

3.3.2 Dictionary Learning of 3D Gestures

The Dictionary Learning Algorithm (DLA) consists in a two-step iteration (see Algorithm 4). First, a gesture from the training database is selected stochastically and then the gesture sparse representation is computed using the 3DRI matching pursuit algorithm (line 6). Second, the atoms involved in the sparse representation of the gesture are updated by applying a gradient descent on the L2 norm of the reconstruction error (line 7). We refer the reader to [4] for further details on the gradient descent step.

Algorithm 2 3DRI Optimization. The current M' components $\{x^{M'}, R^{M'}\}$ are been already optimized ($1 < M' \leq M$). ε is the current reconstruction error and C determines the number of optimization steps.

1: **procedure** $(x'^{M}, R^{M'}) \leftarrow$ `3DRI_OPT`$(\varepsilon, y, x^{M'}, R^{M'}, \phi_\tau^M)$
2: $\varepsilon_0 = \varepsilon$
3: **for** c$\leftarrow 1, C$ **do**
4: **for** m $\leftarrow 1, M'$ **do**
5: $\tilde{\varepsilon}_m = \varepsilon_{c-1} + x_m R_m \phi_{m,\tau_m}$
6: $(x_m, R_m) \leftarrow$ `SVD_Adjust`$(\tilde{\varepsilon}_m, \phi_{m,\tau_m})$
7: **end for**
8: $\varepsilon_c \leftarrow$ y$- \sum_{m=1}^{M} x_m R_m \phi_{m,\tau_m}$
9: **end for**
10: **end procedure**

Algorithm 3 SVD adjustment. It computes the rotation matrix (R) and the correlation coefficient (x) through a SVD decomposition which minimizes the distance between the signal (ε) and the given pattern (ϕ_τ).

1: **procedure** $(x, R) \leftarrow$ `SVD_Adjust`(ε, ϕ_τ)
2: Correlation Matrix : $M_{corr} \leftarrow \varepsilon \phi_\tau^T$
3: SVD : $(U, \sum_1, V) \leftarrow SVD(M_{corr})$
4: Matrix $\sum_2$: $\sum_2 \leftarrow diag(1, 1, det(UV^T))$
5: Optimal rotation matrix : $R \leftarrow U \sum_2 V^T$
6: Correlation : $x \leftarrow trace(\sum_2 \sum_1)$
7: **end procedure**

Algorithm 4 Dictionary Learning Algorithm. Computes the dictionary Φ which minimizes the error of the sparse reconstruction of the training dataset (Y). The final dictionary will have K atoms. M sparse components are used to represent each gesture.

1: **procedure** $(\Phi) \leftarrow$ `DLA_MP`(Y, K, M)
2: $\varepsilon = \infty$
3: $\Phi =$ `CreateDictionary`$(Y, 2 \times K)$
4: **while** $\varepsilon > \varepsilon_{min}$ **do**
5: **for** $\forall y | y \in \mathrm{Y}$ **do**
6: $(x^M, R^M, \phi_\tau^M, \varepsilon_y) \leftarrow$ `3DRI_MP`(y, Φ, M)
7: $\Phi \leftarrow$ `3DRI_LMS`$(y, x^M, R^M, \phi_\tau^M)$
8: **end for**
9: $\varepsilon \leftarrow \frac{1}{|\mathrm{Y}|} \sum_{y=1}^{|\mathrm{Y}|} |\varepsilon_y|_2$
10: $\Phi \leftarrow$ `Prune`(Φ)
11: **end while**
12: **end procedure**

In order to improve convergence and to ensure that the resulting atoms in the dictionary are all representative of the input data, the two main changes are introduced in the DLA algorithm. The initial dictionary contains twice the number of atoms that the final dictionary and each atom is initialized by averaging a random number of gestures for each atom (ten gestures per atom). In addition, to ensure that the final dictionary has the desired number of atoms and only the most relevant ones are selected, after each iteration, the less relevant atom in the dictionary is pruned (line 10). The pruning criteria considers the number of times that the atom is used and the correlation between atoms. The atom removed will be the least used, or one of the two more similar ones. Finally, the stop criteria (line 4) considers the mean reconstruction error ($\varepsilon_{min} = 0.001$), which is updated at each outer iteration (line 9).

3.4 Gesture Recognition Algorithm

The main goal is to take advantage of the 3DRI sparse model to provide an efficient and reliable gesture classification. For practical reasons we consider that the training data is labeled which allows to group the training gestures into different gesture classes. In summary three main decisions have to be taken:

- The number of sparse components used in the decomposition. This choice has two main implications, the computational cost and the quality of the sparse representation.
- The dictionary learning strategy. Whether to train a single dictionary using all the training dataset or train multiple dictionaries (one for each gesture class). How many dictionaries are needed?
- The classification strategy. How to classify the gesture using its sparse representation?

3.4.1 Sparse Components

The number of components in the sparse representation provides a trade-off between the computational cost and the representation quality. However, for gesture recognition, the main constraint is the computational cost. First, the recognition system should be real-time, thus a reduced number of sparse components has to be used if this choice does not negatively impact on recognition accuracy. The cost of building the dictionary is not considered as it is computed in a pre-processing step. Second, we assume that gestures are differentiable through the raw shape of their trajectory. Thus the model can be restricted to one sparse component, assuming that it will unambiguously represent a gesture. If we analyze the cost of Algorithm 1, using only one component it will minimize the computational cost. The SVD decomposition (line 6) is a quadratic operation in terms of the number of points in the trajectory. Furthermore, when considering multiple sparse components, the matching pursuit optimization is required. This optimization step increases the latency while not being relevant in terms of accuracy as shown in our experiments (see Section 4).

In addition, informal testing showed that using one component was enough to achieve high classification rates in the tested application scenarios. Nevertheless, for other situations additional sparse components could be considered in order to capture more details of the gestures. For example, it would be relevant if multiple tracked elements are considered (e.g. bimanual interaction), in particular with uncorrelated movements. However this is not explored in the current work and it is left as a future perspective.

3.4.2 Dictionary Learning Strategy

When building the dictionary, two different strategies were considered named hereinafter supervised and semi-supervised. The main difference among both strategies is the training gesture set (Y) used for the DLA (see Algorithm 4).

The **supervised** approach splits the training data set considering the different gesture classes ($\mathrm{Y} \supset \{\mathrm{Y}_1, \mathrm{Y}_2, ..., \mathrm{Y}_c\}$) and for each Y_i a dictionary is built ($\{\Phi_1, \Phi_2, ..., \Phi_c\}$). Gestures in the same group will have similar pathic information. Each dictionary can have a variable number of atoms according to the gesture variability, which is determined by the pruning strategy. For relatively simple gestures the number of atoms range between one and four. This approach allows the creation of dictionaries that better represent each gesture class as there is no interference between gesture classes while building the dictionary.

The **semi-supervised** approach trains one single dictionary using the entire training dataset, without any a priori information about the different gesture classes. At this step, there is no need of labeled data, as all gestures are processed in the same way. Regarding the

optimal number of atoms for each gesture class, the DLA algorithm will determine the number of needed atoms for each gesture class as more variable gestures will be more represented in the dictionary (additional atoms). Once the dictionary is built, the labeled data is used to split the dictionary into c dictionaries (one for each class) by just selecting the atoms characterizing each class. This step is done in order to enable a common classification scheme for the supervised and the semi-supervised approaches.

3.4.3 Gesture Classification

The proposed classification method is based on the computation of the representation error ε for each dictionary ($\Phi_{1..c}$) using the 3DRI matching pursuit algorithm (see Algorithm 1). The representation error provides the measure of the fit between the input gesture and the atoms for each dictionary. The dictionary which minimizes the representation error (provides a better fit) will determine the class of the input gesture.

However, as the 3DRI model is invariant to rotations, gestures with different orientations (but same pathic information) will share the same atoms. Existing 3D classification methods consider the same gesture with different orientations as different gesture classes [6] (i.e. swiping your arm to the left or to the right). When there is a need of rotation discrimination, the information encoded in the rotation matrix (R) can be considered in a second classification step. Due to simplicity, in our implementation, a k-nearest-neighbors (kNN) was chosen, considering the quaternion-based representation of R. During the training step, a kNN classifier is trained for each gesture class (dictionary) having more than one potential orientation, the distance function considers the angle between quaternions. Note that the choice of the second classification step can be highly dependent on the application. However, as the gesture class is already known at this step, the classification scheme can be highly optimized. Finally, another interesting feature of our approach is that gestures with orientations which have not been trained can be still be recognized during the first classification step. In these situations the rotation information R could be used to infer the gesture orientation.

4 Evaluation

In this section we explore the suitability of our gesture recognizer based on the sparse representation of human motion. First, we analyze the performance when testing our algorithms (supervised, semi-supervised) against an existing motion gesture database. There are rather few 3D gesture databases focusing on 3DUIs, such as: the 6DMG database [6], the IsueLab database [5] or the uWave database [24]. We focused on the 6DMG database as it is the only one which provides trajectory data, whereas other databases provide mostly acceleration data. Furthermore, in order to evaluate our gesture classification algorithm in a real use-case scenario, an additional user evaluation was conducted in which participants had to train and interact with an interactive system using self-defined gestures.

4.1 6DMG Gesture Database

The 6DMG dataset[1] consists of 5600 samples obtained from 28 users (7 left-handed). When observing the gestures included in the 6DMG database we can observe that although it has 20 different gestures, it contains only 6 different gesture classes. The 20 gestures can be defined by the 6 gestures classes and a rotation (e.g. a left swipe and a right swipe). In our test, we only considered 5 gesture classes (see Figure 4) being 18 gestures in total. The 6th gesture was not considered as it only has a rotational component (wrist twist) and thus does not present any distinctive path.

[1] http://www.ece.gatech.edu/6DMG/6DMG.html

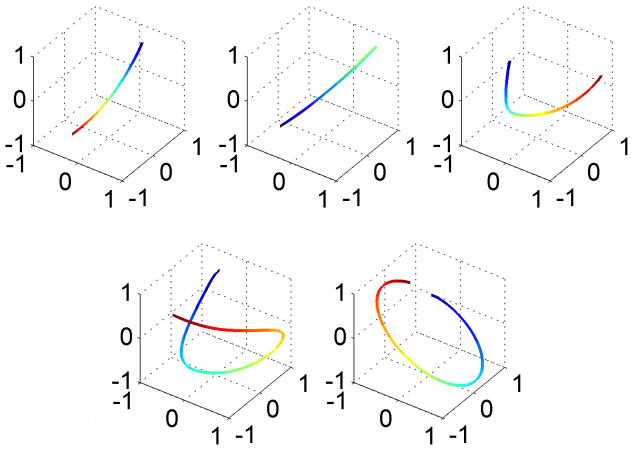

Figure 4: Reconstructed patterns for the 6DMG database (supervised). From left to right and top to bottom: Swipe, Poke, V-shape, X-shape and Circle gesture.

4.1.1 Dictionary Parametrization

In order to build the dictionaries which will be used during the classification step, we have to determine first the number of sparse components (M), the size of the training dataset (Y), the number of atoms in the dictionary (K) and the resampling size for each gesture (see Algorithm 4). Considering the lack of high-frequency motions of gestures in the 6DMG database, one sparse component is enough to capture the gesture variability ($M = 1$). Altough additional sparse components can be used, our preliminary tests did not show an increased recognition rate and had a rather negative impact on performance. In general, we believe that for fast hand trajectories $M = 1$ will be enough in most situations. Regarding the training dataset (Y), two situations are considered, the user dependent (UD) and the user independent condition (UI). For the UD condition, training datasets and test datasets were built without mixing the data from different users. In contrast, for the UI condition a subset of users is used to train the system and the remaining ones are used for testing. The choice of the number of atoms (K) is dependent on (Y) and the learning method considered (supervised vs semi-supervised). While large training datasets can benefit from additional atoms (higher variability in the data), small training datasets might generate noisy or non-representative patterns. The supervised approach only requires a small number of atoms. For the current experiment we considered $K = 1$ for the user dependent scenario and $K = 2$ for the user independent scenario. Smaller values of K also result in lower computation times. In contrast, the semi-supervised approach requires a variable number of atoms which is determined by the algorithm on itself. While the starting value is $K = 10$, during the learning process it is pruned into 5 to 7 atoms depending on the variability of the data. Finally, all gestures were resampled into 60 samples which provided a good trade-off between performance and discretization.

4.1.2 Experimental Conditions

In order to explore the behavior of our approach we considered two independent variables, (1) the algorithm (supervised vs semi-supervised (S vs SS)) and (2) the training dataset (UD vs UI). When building the training and test datasets, we used the same approach as [7] to enable a fair comparison. For the UI condition, five users were used to train the classifier, and the remaining ones used to test it. To avoid potential bias, we considered 100 permutations while building the training and test data sets without differentiating among left and right-handed users. For the UD condition, for each

Table 1: Mean classification rates for different tested configurations. The results from Chen et al. [7] are provided for comparison.

	3D Sparse Representation				
	Supervised		Semi-Supervised		HMM [7]
	Class	Total	Class	Total	Total
UD	99.74%	**99.55%**	**99.75%**	98.22%	99.51%
UI	99.18%	97.73%	**99.33%**	**97.82%**	97.52%

user, half of the data is used for training and half of the data is used for testing. Fifty different permutations for each user are considered, resulting in 560 tests. As dependent variables we considered the classification accuracy when only determining the gesture class (Class Accuracy, 5 classes) and when also determining the rotation (Rotation Accuracy, 18 gestures).

4.1.3 Results

The summary of the class and rotation classification results are shown in Figure 5. Also, the summary of the overall classifications rates are provided in Table 1. We observed that the mean classification rate is higher for the user dependent scenario. This observation is supported by the two-way ANOVA *algorithm* and *training dataset* vs *class classification* which presents a significant main effect for training dataset ($F_{1,658} = 124.85$, $p < 0.001$, $\eta_p^2 < 0.159$). Post-hoc tests (Bonferroni $\alpha = 0.05$), shows that mean classification rate is significantly higher for the user-dependent scenario. In contrast there was no significant difference between the supervised and semi-supervised techniques ($F_{1,657} = 3.31$, $p = 0.069$, $\eta_p^2 < 0.004$).

When considering the recognition for each gesture class (see Figure 5, top), we observed that the class classification rate was higher than 99% for all classes except for *VShape* in the user independent condition. This observation is supported by the three-way ANOVA *algorithm*, *training dataset* and *gesture class* vs *class classification* which showed an interaction effect between training dataset and gesture class ($F_{4,5855} = 246.58$, $p < 0.001$, $\eta_p^2 < 0.144$). Post-hoc tests showed that the classification rate for *VShape* in the UI condition had the lowest classification accuracy.

Regarding rotation accuracy (see Figure 5, bottom), the three-way ANOVA of algorithm, training dataset and gesture class vs rotation classification accuracy, showed an interaction effect training dataset and gesture class ($F_{4,5855} = 98.93$, $p < 0.001$, $\eta_p^2 < 0.063$). Post-hoc tests showed that the classification rate for the *Poke* was significantly higher for the UI condition, and that the classification rate for the *Swipe* and *VShape* were significatly higer for the UD condition.

4.1.4 Discussion

The proposed classification scheme obtained a high classification accuracy for the 6DMG database in all explored conditions. The obtained results are comparable to the ones obtained by [7] (99.51% for the used dependent and 97.52% for the user independent scenarios) and [34] (98% in the user-dependent scenario). We want to notice that the existing classification accuracy was already really high which left few room for improvement. Yet, these results shows that our algorithm is able to produce results at the same level of the current state of the art techniques.

While analyzing the classification errors, we observed two main challenging situations, which also arise in existing gesture classification algorithms. First, we observed abnormal misclassifications due to the higher variability of the user independent scenario for the *VShape*. The confusion matrix shows that most of the misclassified *VShape* gestures are classified as *Swipe* gestures. This contrasts with the higher classification accuracy for the user dependent scenario (almost 100%). The gesture data set showed that for

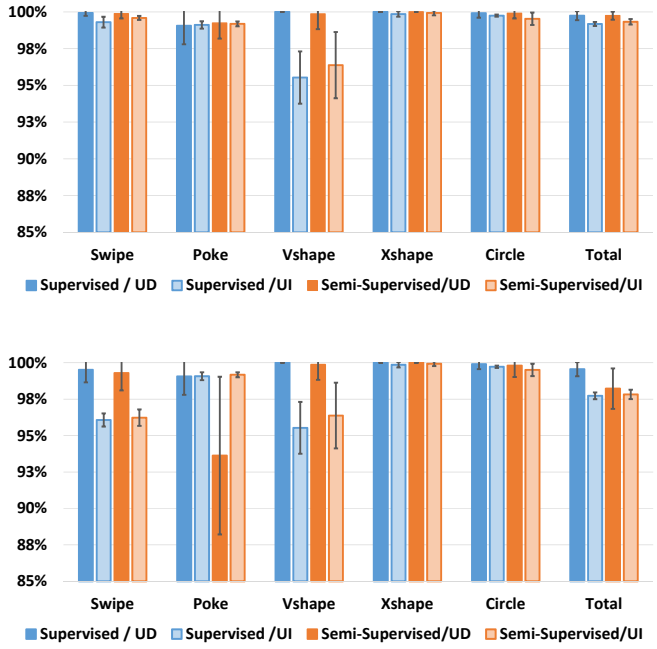

Figure 5: Mean recognition scores for the 6DMG database. Top, mean class classification accuracy (%). Bottom, mean rotation classification accuracy (%).

several users, the *VShape* was closer to a U-shaped gesture. In several cases such U-shaped gestures where erroneously classified as *Swipes*. This shows that different users might have different behaviors to perform *Swipe* and *VShape* gestures which for the user independent scenario can introduce a higher degree of variability.

The second limitation was the orientation discrimination of the eight orientations for the *Swipe* gesture, specially for the user independent scenario. Again, situations prone to increased variability result in more challenging scenarios. Nevertheless, mean classification error is lower than 5% for almost all conditions which is comparable to the average error rate in point and click interfaces [25].

4.1.5 Computational Performance

The computational cost for training a dictionary, when considering only one sparse component is $O(G \times A \times (L_G - L_P + 1) \times L_G^2)$ being G the number of gestures in the training database, L_G the samples for each gesture, A the number of atoms and L_A the samples for each

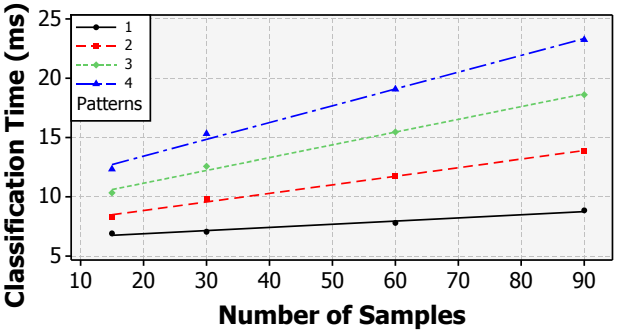

Figure 6: Total mean classification time (including pre-processing) per gesture for the 6DMG database when varying the number of patterns and the number of samples per gesture.

atom ($L_G > L_A; L_G \approx L_A$). Analogously, the cost of computing the sparse representation of a gesture is $O(A \times (L_G - L_A + 1) \times L_G^2)$. Nevertheless, the computational cost is mainly driven by the number of patterns for each dictionary (see Figure 6). The addition of more sparse components will increase the cost of both algorithms exponentially. Regarding the memory footprint, it is negligible as the required memory to store the dictionaries is linearly dependent on the number of patterns.

4.2 Real-Time User Evaluation

The previous analysis showed the suitability of the sparse representation-based classification in a set of a priori designed gestures. In this additional study we aimed to study how a user would be able to interact with a gesture-based interface, by allowing the user to define the desired gestures or using an already defined database. The real-time user evaluation was designed in order to study the robustness of the classification algorithm when using arbitrary gestures and to explore the robustness of the system in a real use-case scenario. Additionally, at the end of the experiment, participants were able to interact with a virtual environment with a set of predefined gestures, this environment also presented contextual graphical elements (3D widgets) in order to provide visual feedback regarding the available actions and the gestures which enabled them (see Figure 7). Finally, in order to gather the subjective impressions of the users, a subjective questionnaire (Likert scale from -5 to 5) was provided at the end of the experiment. The questionnaire had the goal of gathering information about the quality of the gesture recognition, the usability of the system and the user impressions.

4.2.1 Apparatus and Participants

Motion tracking was enabled through a Razer Hydra (magnetic tracking) which was controlled by the participant's dominant hand. A button in the Razer Hydra controller allowed participants to input a gesture to the system. During the first part of the experiment the user interacted in front of a 45" HDTV. In the second part of the experiment, participants were immersed in a virtual environment using an Oculus Rift (DK2). The system was driven by a standard workstation PC and the virtual environment was built using Unity 3D providing a constant refresh rate of 75 frames per second. The gesture classification framework was built in Matlab and interfaced to Unity using a DLL interface. Nine male volunteers with a mean age of ($M = 25.6, SD = 3.05$) took part in the experiment.

4.2.2 Experimental Protocol

After reading and signing the consent form, participants were asked to design a gesture dataset composed of six gestures. The only guidelines provided were to design six gestures being easy to remember, to execute and not having similar trajectories. Once the experimenter validated that the participant followed the indications when creating the gesture dataset, the participant was asked to train the gesture recognition system by performing twenty times each of the designed gestures (train dataset). Then, participants performed an additional twenty repetitions for each gesture (test dataset), during this phase participants were provided with real-time classification feedback. Given the size of the training data set, computing Φ and training the kNN classifier required just a few seconds.

In the second part of the experiment, users were presented a predefined gesture dataset of six gestures (the 5 gestures from the 6DMG database and an additional Z-shaped gesture). For this dataset, the classification system was trained using the 6DMG dataset and data recorded in the lab (just the Z-shaped gesture). After a short training session (one or two repetitions for each gesture), as in the first part of the experiment, users were asked to perform twenty repetitions for each gesture. Participants had to perform 360 (240 + 120) gestures in total. The overall classification accuracy was measured for each condition.

Furthermore and lastly, in order to evaluate the system in a realistic use-case, users were immersed using an HMD in a virtual apartment (see Figure 7 bottom) and instructed to interact with the different elements in the apartment. In the virtual environment users had to navigate using a simple steering technique (joystick) in order to fully explore it. In total, including the questionnaires, the entire experiment took approximately 40 minutes.

4.2.3 Results and Discussion

Although users were provided with minimal instructions when designing the gesture database, and all had never used the system before, all succeeded on creating unambiguous gesture datasets (see Figure 9). Furthermore, the overall classification rate was higher than 96% ($M = 96.98, SD = 3.36$). For the second dataset, the mean classification accuracy was higher than 95% ($M = 95.63, SD = 4.91$). We have to note that the 5 out of 6 gestures where trained with a different tracking system (from the 6DMG database) and potentially participants had different instructions while performing them.

Regarding the overall subjective impression of the system (see Figure 8), participants agreed that the system was simple to use ($M = 4.5, SD = 0.19$), responsive ($M = 4.38, SD = 0.74$) and accurate ($M = 3.75, SD = 0.45$). Users had no particular difficulty while training and testing the predefined gestures nor the custom gestures and the recognition latency was always lower than 10ms. Users also considered that gestural interfaces are useful ($M = 3.62, SD = 2.1$) and natural ($M = 3.13, SD = 1.81$). In contrast, we observe lower scores in terms of comfort ($M = 2.00, SD = 0.91$) and fatigue ($M = -0.75, SD = 0.94$), as it can be expected in such systems. Yet, subjective results show

Figure 7: Unity 3D application used to test the real-time gesture classification algorithm. Users were immersed in a virtual intelligent house driven by our 3D gesture recognition system. The user could walk in the virtual apartment and interact with different virtual objects.

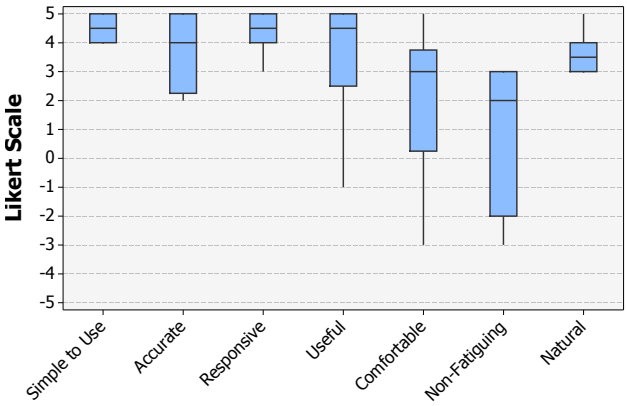

Figure 8: Subjective questionnaire results. Participants were asked to rate the usability of our gesture recognition system.

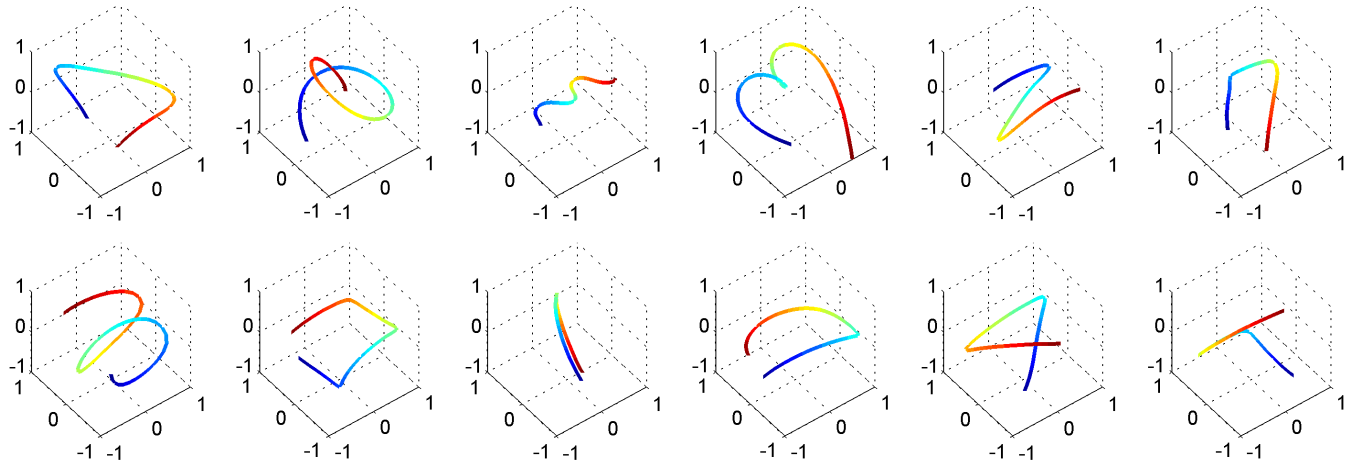

Figure 9: Patterns trained by two different participants (split by rows) during the user evaluation. Users had the tendency of design simple 2D gestures, although 3D gestures were sometimes used (first row, second column and second row, first column).

that participants had a positive appreciation of using gestures as the main input modality and were able to accurately control the system. Furthermore, users appreciated the contextualized feedback (3D widgets) as it allowed to use the system with little a priori information. Taken together, these results demonstrates the robustness of the proposed system with arbitrary gestures and positive subjective impressions.

4.3 Limitations

The gesture recognition algorithm presented in this work is the first 3D gesture recognizer based on the sparse representation of 3D human motion. However, it still presents two major limitations that should be addressed to enable its usage in real applications.

First, temporal segmentation is controlled by the user (e.g. by a button press). This solution, although it could be argued to be the most efficient mechanism for segmentation, both in terms of accuracy and computation cost, it is not suited when the user is not equipped with a hand-held device. In order to provide efficient segmentation, there is the need to infer user intentions: determine when the user is actually performing a gesture or when the user is just moving their hands for other purposes (e.g. idle). While, a sliding window can be used to spot gestures, the optimal size, and the number of sliding windows remains a challenging problem, mainly due to the strong gesture variability. Contextual information, such as the user state or the application state could be used to restrict the amount of active gestures classes and improve the gesture spotting accuracy, but additional works are required to ensure a robust segmentation algorithm.

Second, a limitation, but also a strength of the proposed approach, is the rotation and speed invariances. On one hand, invariances ease the training process (reduce the training samples) and ease the classification process (lower number of gesture classes). On the other hand, in some scenarios, the rotation and the speed (acceleration) of the gesture provide additional gestural information that should be extracted. Although the proposed solution includes an additional classification (kNN) step to discriminate the gesture rotation based on the rotation component of the sparse representation (R), no additional information is extracted from the speed data. We believe that in order to extract additional information from the gesture data, fusion approaches [3] could be considered. A combination of other gesture classifiers could be used to further analyze and classify gestures. However, this classification step might be dependent on the application requirements.

5 Conclusion

Efficient and robust gesture recognition algorithms are required for the democratization of natural user interfaces and allow the development of new human-computer interfaces. For this purpose, in this paper we have introduced the first gesture recognition algorithm based on the sparse representation of human motion. The proposed gesture recognition algorithm, in contrast to existing solutions, ensures position, scale and rotation invariance and provides a strong tolerance to noise. This allows to decrease the training data required, for example, there is no need to train the same gesture with different orientations or train the system with users having different morphologies (e.g. different heights). In addition, for each gesture, the redundancy of the sparse representation algorithm (additional atoms in the dictionary) supports the recognition of slight variations of the same gesture.

Moreover, we have validated the proposed algorithm through two different evaluations. The first evaluation, considering an existing 3D gesture database, has showed that the proposed algorithm achieves high classification accuracy in user independent situations and can even tolerate changes in the users' handedness. Such characteristics are key in order to ensure that the system does not require tedious training procedures each time a new user uses the system or the hardware configuration changes. On the other hand, we observed that the classification based on the sparse coding can be improved by using more adapted classification schemes, specially for the orientation discrimination (currently we were using a nearest-neighbors classifier). Additional information or more sophisticated machine learning approaches could be used in order to increase rotation discrimination. Furthermore, in a user evaluation, we have shown the robustness of the proposed approach when users have total freedom when choosing the gesture set and when interacting with the system in real-time. Although participants did not have previous experience on gestural interfaces, the system was able to recognize the gestures done with a classification error lower than 5%. Still, we observe that users are not always exploiting the full potential of the gesture recognition system as they constraint the length and orientation of the gestures, or they define too complex gestures that can be tedious to execute. This is not a limitation of the classification algorithm itself, but it has to be considered when designing interfaces based on gestural input.

Future works will envision the recognition of complex gestures, such as bi-manual or full-body motions. The proposed approach can be used to decompose gestures in multiple patterns, for exam-

ple, motion can be expressed by the movement performed by different body joints. The combination of multiple patterns provides additional information about the gesture being performed, which could improve gesture recognition for complex gestures.

REFERENCES

[1] Context-based hand gesture recognition for the operating room. *Pattern Recognition Letters*, 36:196 – 203, 2014.

[2] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[3] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16(6):345–379, 2010.

[4] Q. Barthélemy, A. Larue, and J. Mars. Decomposition and Dictionary Learning for 3D Trajectories. *Signal Processing*, 98:423–437, 2014.

[5] S. Cheema, M. Hoffman, and J. J. LaViola. 3D Gesture Classification with Linear Acceleration and Angular Velocity Sensing Devices for Video Games. *Entertainment Computing*, 4(1):11–24, 2013.

[6] M. Chen, G. AlRegib, and B.-H. Juang. 6DMG: A New 6D Motion Gesture Database. In *Proceedings of the 3rd Multimedia Systems Conference (MMSys'12)*, page 83, 2012.

[7] M. Chen, G. AlRegib, and B.-H. Juang. Feature Processing and Modeling for 6D Motion Gesture Recognition. *IEEE Transactions on Multimedia*, 15(3):561–571, 2013.

[8] S. Chen, S. A. Billings, and W. Luo. Orthogonal Least Squares Methods and their Application to non-Linear System Identification. *International Journal of Control*, 50(5):1873–1896, 1989.

[9] E. Coupet, F. Moutarde, and S. Manitsaris. Gesture recognition using a depth camera for human robot collaboration on assembly line. *Procedia Manufacturing*, 3:518 – 525, 2015. 6th International Conference on Applied Human Factors and Ergonomics.

[10] G. M. Davis, S. G. Mallat, and Z. Zhang. Adaptive time-frequency decompositions. *Optical Engineering*, 33(7):2183–2191, 1994.

[11] S. R. Fanello, I. Gori, G. Metta, and F. Odone. Keep It Simple and Sparse: Real-time Action Recognition. *J. Mach. Learn. Res.*, 14(1):2617–2640, 2013.

[12] T. Guha and R. K. Ward. Learning Sparse Representations for Human Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, 2012.

[13] M. Hoffman, P. Varcholik, and J. J. LaViola. Breaking the Status Quo: Improving 3D Gesture Recognition with Spatially Convenient Input Devices. In *IEEE Virtual Reality (VR'10)*, pages 59–66, 2010.

[14] F. G. Hofmann, P. Heyer, and G. Hommel. Velocity Profile Based Recognition of Dynamic Gestures with Discrete Hidden Markov Models. In *Gesture and Sign Language in Human-Computer Interaction*, volume 1371 of *Lecture Notes in Computer Science*, pages 81–95, 1998.

[15] J. F. Jégo, A. Paljic, and P. Fuchs. User-Defined Gestural Interaction: a Study on Gesture Memorization. In *IEEE 3D User Interfaces (3DUI'13)*, pages 7–10, 2013.

[16] N. Kawarazaki, I. Hoya, K. Nishihara, and T. Yoshidome. 7 Cooperative Welfare Robot System Using Hand Gesture Instructions. In *Advances in Rehabilitation Robotics*, volume 306 of *Lecture Notes in Control and Information Science*, pages 143–153, 2004.

[17] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo, and S. D. Marca. Accelerometer-Based Gesture Control for a Design Environment. *Personal and Ubiquitous Comp.*, 10(5):285–299, 2006.

[18] S. Kratz and M. Rohs. Protractor3D: A Closed-form Solution to Rotation-invariant 3D Gestures. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI '11)*, pages 371–374, 2011.

[19] K. Lai, J. Konrad, and P. Ishwar. A Gesture-Driven Computer Interface Using Kinect. In *IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 185–188, 2012.

[20] M. E. Latoschik and M. Erich. A user interface framework for multimodal VR interactions. In *Proceedings of the ACM 7th international conference on Multimodal interfaces*, pages 76–83, 2005.

[21] J. J. Laviola Jr. 3D Gestural Interaction: The State of the Field. *ISRN Artificial Intelligence*, pages 1–18, 2013.

[22] Y. Li. Protractor: A fast and accurate gesture recognizer. In *Proceedings of the 28th international ACM conference on Human factors in computing systems (CHI'10)*, pages 2169–2172, 2010.

[23] Y. Li, C. Fermuller, Y. Aloimonos, and H. Ji. Learning shift-invariant sparse representation of actions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, pages 2630–2637, 2010.

[24] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uWave: Accelerometer-based Personalized Gesture Recognition and its Applications. In *IEEE International Conference on Pervasive Computing and Communications*, pages 1–9, 2009.

[25] I. S. MacKenzie, T. Kauppinen, and M. Silfverberg. Accuracy Measures for Evaluating Computer Pointing Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'01)*, pages 9–16, 2001.

[26] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. on Signal Processing*, 41(12):3397–3415, 1993.

[27] S. Mitra and T. Acharya. Gesture Recognition: A Survey. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.

[28] C. S. Myers and L. R. Rabiner. A Comparative Study of Several Dynamic Time-Warping Algorithms for Connected-Word Recognition. *Bell System Technical Journal*, 60(7):1389–1409, 1981.

[29] S. Poularakis, G. Tsagkatakis, P. Tsakalides, and I. Katsavounidis. Sparse representations for hand gesture recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'13)*, pages 3746–3750, 2013.

[30] T. Pylvänäinen. Accelerometer Based Gesture Recognition Using Continuous HMMs. In *Pattern Recognition and Image Analysis*, pages 639–646, 2005.

[31] T. Schlömer, B. Poppinga, N. Henze, and S. Boll. Gesture Recognition with a Wii Controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 11–14, 2008.

[32] T. M. Takala, P. Rauhamaa, and T. Takala. Survey of 3DUI applications and development challenges. In *IEEE Symposium on 3D User Interfaces*, pages 89–96, 2012.

[33] R. D. Vatavu. The effect of sampling rate on the performance of template-based gesture recognizers. In *Proceedings of the 13th international conference on multimodal interfaces - ICMI '11*, pages 271–278, New York, New York, USA, 2011. ACM Press.

[34] R. D. Vatavu. The Impact of Motion Dimensionality and Bit Cardinality on the Design of 3D Gesture Recognizers. *International Journal of Human-Computer Studies*, 71(4):387–409, 2013.

[35] R. D. Vatavu, L. Anthony, and J. O. Wobbrock. Gestures as Point Clouds. In *Proceedings of the 14th ACM international conference on Multimodal interaction (ICMI'12)*, pages 273–280, 2012.

[36] J. P. Wachs, H. I. Stern, Y. Edan, M. Gillam, J. Handler, C. Feied, and M. Smith. A Hand Gesture Sterile Tool for Browsing MRI Images in the OR. *Journal of the American Medical Informatics Association*, 15(3):321–323, 2008.

[37] J. O. Wobbrock, A. D. Wilson, and Y. Li. Gestures Without Libraries, Toolkits or Training: a $1 Recognizer for User Interface Prototypes. In *Proceedings of the 20th annual ACM symposium on User interface software and technology (UIST'07)*, pages 159–168, 2007.

[38] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Yi Ma. Robust Face Recognition via Sparse Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[39] X. Zhang, X. Chen, W. Wang, J. Yang, V. Lantz, and K. Wang. Hand gesture recognition and virtual game control based on 3d accelerometer and emg sensors. In *Proceedings of the 14th International Conference on Intelligent User Interfaces (IUI '09)*, pages 401–406, 2009.

[40] G. Zhu, C. Xu, Q. Huang, W. Gao, and L. Xing. Player Action Recognition in Broadcast Tennis Video with Applications to Semantic Analysis of Sports Game. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 431–440, 2006.

[41] M. Zhu, H. Sun, and Z. Deng. Quaternion space sparse decomposition for motion compression and retrieval. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 183–192, 2012.