# Online gesture recognition from pose kernel learning and decision forests

Leandro Miranda[a], Thales Vieira[a,*], Dimas Martínez[a], Thomas Lewiner[b], Antonio W. Vieira[c], Mario F. M. Campos[c]

[a]*Institute of Mathematics — UFAL — Maceió — Brazil*
[b]*Department of Mathematics — Pontifícia Universidade Católica — Rio de Janeiro — Brazil*
[c]*Department of Computer Science — UFMG — Belo Horizonte — Brazil*

**Abstract**

The recent popularization of real time depth sensors has diversified the potential applications of online gesture recognition to end-user natural user interface (NUI). This requires significant robustness of the gesture recognition to cope with the noisy data of popular depth sensor, while the quality of the final NUI heavily depends on the recognition execution speed. This work introduces a method for real-time gesture recognition from a noisy skeleton stream, such as those extracted from Kinect depth sensors. Each pose is described using a angular representation of the skeleton joints. Those descriptors serve to identify key poses through a support vector machine multi-class classifier, with a tailored pose kernel. The gesture is labeled on-the-fly from the key pose sequence through a decision forest, that naturally performs the gesture time control/warping and avoids the requirement for an initial or neutral pose. The proposed method runs in real time and have shown its robustness in several experiments.

*Keywords:* Online gesture recognition, Key pose identification, Skeleton representation, Depth sensors, SVM multi-class, Decision forest, 3d motion, Natural user interface

## 1. Introduction

Human gesture recognition is an active topic of research and covering a wide range of applications including originally monitoring, control and analysis. More recently, several applications use real-time (online) gesture recognition to control entertainment devices such as game consoles, virtual reality setups, motion capture to graphics model animation, or automatic control of domestic utilities.

The variety of potential applications have intensified the efforts to improve the automatic recognition of human gestures. The evolution and popularization of depth sensors, which currently generate depth maps in real time as well as their skeletons, is paving the way for the development of high quality natural user interface (NUI) beyond their use as game consoles. Improving the quality of a NUI essentially means to increase the execution speed of the gesture identification and its robustness, in particular with noisy data such as skeletons extracted from Kinect sensors, and this is the objective of the present work.

The gesture recognition problem can be stated as the process of automatically labeling gestures performed by a person based on sensory data, usually captured as sequences of positions in space. This is a particularly challenging task, specially considering that different users perform gestures with different speeds and distinct sequences of poses. In this work, we propose a gesture recognition method from captured skeletons in real time that tackles the aforementioned issues. More specifically, all our experiments are performed using the popular Kinect platform, a real-time depth sensing system that parses a depth-map stream at 30 frames per second, from which positions of the skeleton nodes for each frame can be estimated in real time [23].

*Contributions.* A human gesture can be formally described as the continuous evolution of body poses over time. Interestingly, we verbally describe such gestures by sequentially identifying a few extreme poses, referred to as *key poses* [13], as illustrated in Figure 1. In this case, we recognize a gesture by extracting those key poses, classifying them, and then identifying sequences of key poses as a gesture. Following this observation, we focus here on improving and tailoring the three main ingredients of key pose gesture recognition: pose descriptor, pose identification, and labeling of pose sequences.

Our pose descriptor relies on spherical angular representations of joints, similarly to the recent work of Raptis *et al.* [20]. However, our method is more robust for usual gestures, and it allows for real-time pose classification. In particular, it improves the representation of secondary joints (arms, hands, legs and feet) to better suit NUI applications.

*Corresponding author

*URL:* `leandrobotelhoalves@gmail.com` (Leandro Miranda), `http://www.im.ufal.br/professor/thales/` (Thales Vieira), `http://www.im.ufal.br/professor/dimas/` (Dimas Martínez), `http://thomas.lewiner.org/` (Thomas Lewiner), `http://homepages.dcc.ufmg.br/~awilson/` (Antonio W. Vieira), `http://www.verlab.dcc.ufmg.br/` (Mario F. M. Campos)

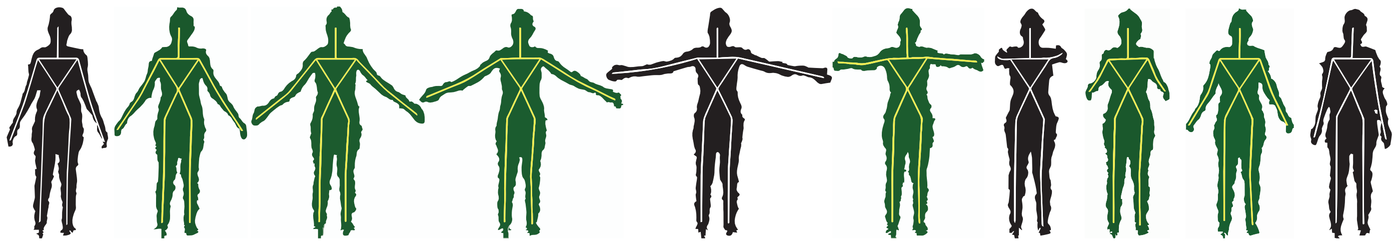

Figure 1: Gesture representation from key poses: Our method represents a body gesture as a sequence of a few extreme body poses, referred to as *key poses*. In the example above, a gesture composed by opening arms and then clapping hands is represented by the key poses in black.

The pose identification process combines several Support Vector Machine (SVM) classifiers [25], one per reference key pose. We propose a *pose kernel* that entails the angular nature of our representation, and use that pose kernel distance in feature space as the confidence measure. This pose kernel significantly improves the robustness of the method over a kernel based on Euclidean distances [15], as seen in the results section. Moreover, the small pose descriptor size allows for online training and recognition.

Finally, we propose a scheme for gesture recognition based on decision forests. Each forest node is a key pose, eventually including time constraints, and the leaves are the gesture labels. This decision forest is learned during the training phase. Each tree is rooted at a key pose, and a leaf-to-root path represents a possible sequence of key poses of that leaf gesture. At each identification of a new key pose, the tree rooted at that pose is used to check if it completes a gesture. This allows for real time gesture recognition without the need for a neutral/initial pose. Moreover, the decision state machine produces a natural and robust time warping. The whole process is robust even with noisy depth-based skeletons [23] as shown in the results section.

This work is an extension of a conference work presented at Sibgrapi 2012 [15].

## 2. Related work

Human gesture recognition has been extensively studied, and a large body of literature has been produced for application in areas such as surveillance, home monitoring and entertainment. We summarize here the most related work for gesture recognition according to the spatial representation used: local, global or parametric.

In the local category, the methods typically use point-wise descriptors evaluated at some points of interest, and then use a bag of features (BoF) strategy to represent actions. This approach has attracted much attention in the past few years [24, 2, 7, 19], and an example of largely used local feature is the Space-Time Interest Point (STIP) presented by Laptev and Lindeberg [9]. A drawback of local features approaches is that they lose spatial context information between interest points.

The methods in the global category use features such as silhouettes [13, 29, 10] or template based representations [3, 1], where spatial context information is preserved. However, global features usually miss some precise details of the pose such as body joints identification.

Finally, parametric methods try to reconstruct a model of the human body with identification of joints to obtain an skeleton. Skeletons can be obtained by strategies such as the Motion Capture (MoCap) models [8, 18], where a set of markers is attached to the human body at specific points that are tracked during motion. Using this representation, spatial features are constructed using some geometric measures and relations, e.g. angles [6, 5] and body joints positions [17, 16]. In particular, Vieira *et al.* [26] show that the matrix of distances between body joints completely describe a pose up to rigid movements, and such matrices serve to define low-dimensional invariant features for classifying actions.
MoCap skeletons strongly depend on rather sophisticate capture systems. A more accessible way to generate skeletons is proposed by Shotton *et al.* [23], who obtain skeletons without markers by computing joint coordinates in real time from depth maps. In particular, such skeletons can be obtained from the popular *Kinect* sensor. Compared to MoCap data, skeletons from Kinect are easier to obtain, which have driven the popularity of this sensor. However, they show a high level of noise and spatial discontinuity, turning gesture recognition from depth data a sizable challenge, and that is the focus of the present work.

Gesture recognition using skeletons from Kinect has received a lot of attention recently. In particular, Li *et al.* [11] published the MSR Action3D dataset, a database with depth maps sequences and their respective skeletons, composed of 20 different short and non-repetitive action classes, each performed by several subjects [12]. They distinguish three different subsets and present their classification rate obtained for each test. This dataset became a benchmark for several recent works [27, 30, 31]. We also use this dataset to validate our approach for action classification and present comparative results with other state-of-the-art works in the literature.

Reyes *et al.* [21] obtain 3D coordinates of skeletal models using a reference coordinate system to make the description view point invariant and tolerant to corporal differences among subjects. Results are reported for only five

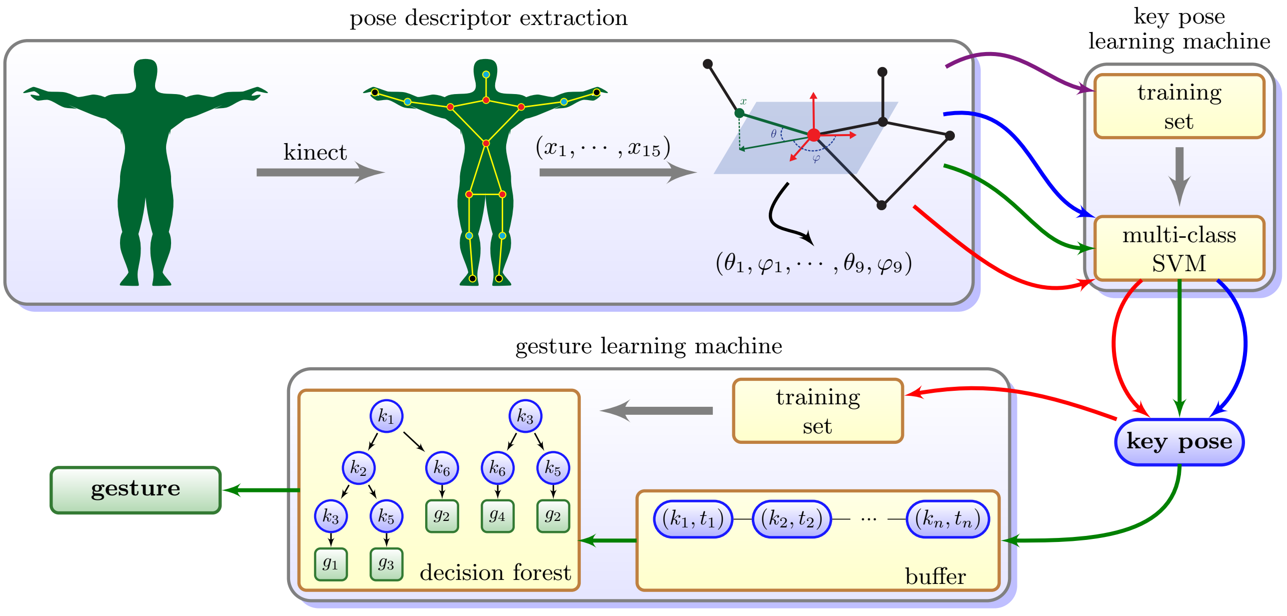

Figure 2: Our method learns and recognizes, in real time, key poses and gestures from a compact joint-angle skeleton representation. The same machine is used for the four steps: training the key pose learning machine (purple arrow); recognizing user key poses in real time (blue arrows); training the gesture learning machine (red arrows); recognizing user gestures in real time (green arrows).

different categories. Raptis *et al.* [20] introduce a method for classifying dance gestures using Kinect skeletons where a large number of gesture classes are used. Their pose descriptor uses spherical coordinates in a frame obtained by Principal Component Analysis on a subset of the body points that define the torso. In spite of the high recognition rate reported, their classification method is limited to dance gestures and is conceived based on the strong assumption that the input motion adheres to a known music beat pattern. We extend their pose descriptor to a completely invariant angular representation. Vieira *et al.* [27] propose a global feature, called Space-Time Occupancy Patterns for action recognition from depth map sequences where space and time axes are used to define a 4D grid. A saturated histogram of point count in the grid cells is used as features for action classification. Yang & Tian [30] propose a new type of feature based on position differences of joints which combines action information including static posture, motion, and offset. They employ the Naïve-Bayes-Nearest-Neighbor classifier for multi-class action classification and also explore the number of frames that are needed to classify the actions. Yang *et al.* [31] project depth maps onto three orthogonal planes and accumulate global activities across entire video sequences to generate the Depth Motion Maps (DMM). Histograms of Oriented Gradients (HOG) are then computed from DMM as the representation of an action video. The pose descriptor proposed here is simpler but still invariant under rigid motion. We derive a pose kernel to adapt its angular nature to the SVM classifier, leading to a robust yet efficient pose identification.

Beyond spatial representation, gesture recognition has to address the issue of temporal alignment among different sequences that may present significant variability. To address time alignment, Hidden Markov Models is largely used [4, 32]. Li *et al.* [10] propose a graphical modeling of human action where key poses define nodes of the graph and each action describes a path in the Action Graph. Müller *et al.* [17, 16] use Dynamic Time Warping (DTW) on MoCap data to address time alignment in their Motion Templates, and Reyes *et al.* [21] extend DTW, assigning weights to features based on inter-intra class action variability. In this work, we propose a simpler, yet robust and efficient method to perform alignment through a decision forest scheme.

## 3. Technical Overview

Our method use three main ingredients both for training and recognition, as illustrated in Figure 2: A pose descriptor that concisely represents a human body pose, described in Section 4.1; a multi-class SVM learning machine that robustly identifies key poses in real-time, detailed in Section 4.2; and a decision forest to recognize human gestures, optionally considering time/speed constraints (Section 5).

*Pose Descriptor.* We convert the set of points representing the nodes of the skeleton to a reduced, invariant representation to robustly solve the gesture recognition problem. We extend the representation of Raptis *et al.* [20] to a pure joint-angle representation. It provides invariance to sensor orientation and reduces redundancy and space dimensionality, while preserving relevant information for the classification of key poses.

*Key Pose Learning.* We use a multi-class SVM [25] approach to recognize key poses in real time. During a training phase, the users performs and label several examples of key poses, such as in the MSR Action 3D data set [12]. This training set is used to build several SVM binary classifiers that robustly recognize key poses, in a *one-versus-all* approach. We propose a pose kernel that properly handles the angular nature of our joint-angle representation. The efficiency of this method permits both real-time training and classification, in particular allowing the user to eventually improve the training set by adding key poses or correcting misclassified poses.

*Gesture Training and Recognition.* After key pose training, the user execute and label examples of gestures. The key poses appearing in each gesture are automatically identified from the SVM machines. After training, a decision forest is optimized to allow efficient search for any sequence of key pose that compose a gesture, as described in Section 5. For each gesture performance, the key poses are accumulated into a circular buffer, checking in the decision forest if the sequence completes a known gesture. This avoids the need for an initial/neutral pose. The forest nodes can optionally consider time/speed constraints. Even when different users perform the same gesture with different duration of key poses, the decision forest provides an effective and robust solution to that temporal alignment problem.

## 4. Key Pose Statistical Learning

Key pose gesture recognition methods are highly dependent on the robustness of pose classification, demanding efficiency in order to provide real time performance. To solve this classification problem, we propose a supervised learning approach, where the training key poses are obtained from the user. We further aim to deliver robust pose identifications even with small training sets, as those provided by a single short training session. Finally, we would like the user to be able to provide, at any moment, additional labeled training data to correct and improve the classifier robustness, while keeping its efficiency.

We build such a classifier using a multi-class composition of *Support Vector Machines* (SVM) binary classifier, whose formulation is well suited to meet our requirements. SVM received a lot of attention in the machine learning community since it is optimal in the sense of the *VC* statistical learning theory [25]. We refer the reader to the book of Smola and Schölkopf [22] for a complete introduction to SVM techniques. This section briefly describes some basic aspects of the multi-class SVM approach we adopted, as well as of our joint-angle skeleton representation and the tailored pose kernel we developed.

### *4.1. Joint-angle Representation*

The skeleton representation must be invariant to sensor orientation and to global translation of the body. It also must be able to minimize issues with skeleton size variations from different individuals, while still concisely capture all the relevant body pose information.

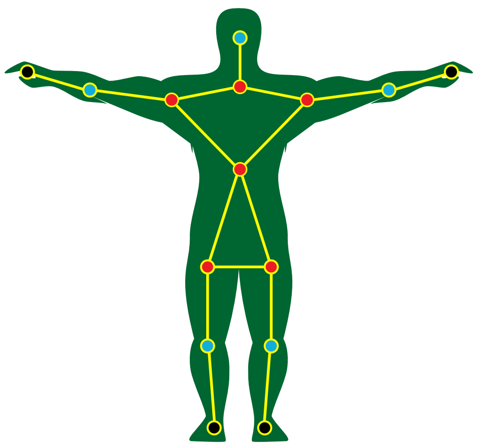

Figure 3: Skeleton's graph: torso joints are represented in red; first-degree joints in blue; and second-degree joints in black.

The raw skeleton input from Kinect sensor is, for each frame, is a sequence of graphs with 15 nodes [23], where each node has its geometric position represented as a 3D point in a global Cartesian coordinate system (Figure 3). The joints adjacent to the torso are called first-degree joints, while joints adjacent to first-degree joints are classified as second-degree joints. First-degree joints include the elbows, the knees and the head, while the extremities – hands and feet – are regarded as second-degree joint. Different body poses are essentially obtained by rotating the first and the second-degree joints. Note that each joint movement has two degrees of freedom, later referred to as a zenith angle $\theta$ and an azimuth angle $\varphi$ (Figure 4). The distance between adjacent joints is always constant for a given individual.

In the work of Raptis *et al.* [20], a straightforward joint-angle representation is proposed by writing the position $x_l \subset \mathbb{R}^3$ of a joint $l$ in local spherical coordinates, omitting the radial distance. To do so, a torso basis is estimated by applying a PCA to a $7-3$ torso matrix filled with the torso node positions. Then, the spherical coordinates of each first-degree joint are computed as a translation of this torso basis to the joint. However, the same torso basis is used as a reference to convert the second-degree joints, leading to a non-local description of the angles. Also, as mentioned by the authors, some combinations of joint positions can result in collapsed projections, and consequently, in inconsistent angles, as in the open arms position [20]. We use the same torso basis for first-degree joints, but improve the representation of second-degree joints by considering rotations of the orthonormal torso basis $\{\mathbf{u}, \mathbf{r}, \mathbf{t}\}$.

More precisely, let $\mathbf{v}, \mathbf{w}$ be the vectors supporting the bones adjacent to a joint. For example, $\mathbf{v}$ would be the vector defined by the right shoulder and the right elbow and $\mathbf{w}$ the vector between the right elbow and the right wrist. To define a local basis for the right hand, we rotate the torso basis $\{\mathbf{u}, \mathbf{r}, \mathbf{t}\}$ by the angle $\beta = \widehat{\mathbf{v}, \mathbf{r}}$ around the axis $\mathbf{b} = \mathbf{v} \times \mathbf{r}$. The rotated basis is translated to the right elbow and the spherical coordinates of the right hand are computed as

- $\theta$ - the angle between $\mathbf{v}$ and $\mathbf{w}$
- $\varphi$ - the angle between the rotation of $\mathbf{t}$ and the projection of $\mathbf{w}$ on the plane orthogonal to $\mathbf{v}$

If $\mathbf{v}$ and $\mathbf{w}$ are collinear, we just set $\phi = 0$, since the azimuth is not defined, and this will not be an issue to our SVM pose classifier. The second-degree joints are thus constructed using variants of the torso basis, such that collapsing issues are avoided by other body constraints.

Finally, each joint position $x_l$ is represented using a pair of spherical angles $(\theta_l, \varphi_l)$ that specifies it in a locally defined spherical coordinate system. Considering a skeleton with 9 joints, a body pose joint-angle representation is a pose descriptor vector $\mathbf{v} = (\theta_1, \varphi_1, \ldots, \theta_9, \varphi_9) \in \mathbb{R}^{18}$. Actually each pair $(\theta_l, \varphi_l)$ represents a pair of positions on the sphere, and this particularity will be correctly handled by the pose kernel.

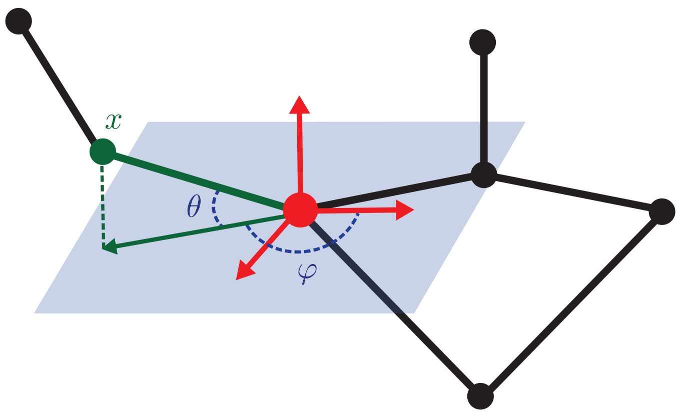

Figure 4: Joint-angle representation: each body pose is a product of joints movements, each of which has two degrees of freedom in local spherical coordinate systems: the zenith angles $\theta$ and the azimuth angles $\varphi$.

### *4.2. Multi-class SVM formulation*

The classifier looks for similarities with reference key poses in a pre-defined set $\mathcal{K} = \{\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^{|\mathcal{K}|}\}$. Those key pose classes will be used to build gesture representations later on. During key pose training, the user creates a training set by providing several examples of each key pose. In our experiments, 10 examples per key pose were usually enough, as in the MSR Action 3D data set[12]. The multi-class SVM learning machine is fed with the training set $\mathcal{T} = \{(\mathbf{v}^1, \mathbf{c}^1), (\mathbf{v}^2, \mathbf{c}^2), \ldots\}$, where each key pose $\mathbf{v}$ is trained by the user for a specific label $\mathbf{c}$. More precisely, each vector $\mathbf{v}^i \in \mathbb{R}^{18}$ is the pose descriptor of the trained key pose, while $\mathbf{c}^i \in \{1, \ldots, |\mathcal{K}|\}$ is an integer identifying the key pose class.

For each reference key pose $\mathbf{p} \in \{1, \ldots, |\mathcal{K}|\}$, we build an SVM classifying function $\hat{f}_p$ as the kernel distance to some of the trained pose (the support vectors SV):

$$\hat{f}_{\mathbf{p}}(\mathbf{v}) = \textstyle\sum_{j \in SV} \alpha^j \ \psi_{\mathbf{p}}(\mathbf{c}^j) \ \phi(\mathbf{v}^j, \mathbf{v}) + b,$$

where

$$\psi_{\mathbf{p}}(\mathbf{c}) = \begin{cases} 1 & \text{if } \mathbf{c} = \mathbf{p}, \\ -1 & \text{otherwise,} \end{cases}$$

and the set of support vectors $SV$ and weights $\alpha^j$ are determined by the SVM optimization.

*Pose kernel.* The function $\phi\colon \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ maps a pair of pose descriptors to their scalar product in a feature, possibly infinite-dimensional space, turning $\hat{f}_{\mathbf{p}}$ into a non-linear classifier. Our pose descriptors represent a body pose through several joints spherical angles. Consequently, using a metric on spheres is more suitable to evaluate distances between pose descriptors than the usual euclidian distance ([15]). However, as most conventional kernels (e.g., the Gaussian kernel) are usually parameterized by euclidian distances between vectors. We propose an alternative *Pose kernel* to respect the angular nature of our pose representation, inspired by spectral kernels [14].

Given two points $v_l^i = (\theta_l^i, \varphi_l^i)$ and $v_l^j = (\theta_l^j, \varphi_l^j)$ on the sphere $\mathbb{S}^2$, their geodesic distance on $\mathbb{S}^2$ is given by:

$$\Delta(v_l^i, v_l^j) = \arccos(\sin\theta_l^i \sin\theta_l^j + \cos\theta_l^i \cos\theta_l^j \cos|\varphi_l^i - \varphi_l^j|) \quad .$$

Let $\mathbf{v^i}, \mathbf{v^j}$ be two pose descriptors. For each joint $l$, $\Delta(v_l^i, v_l^j)$ represents the geodesic distance of the pose descriptors projected on the unit sphere restricted to that joint. By summing the squares over all nine joints representing a body pose, the metric is obtained as the geodesic distance on $(\mathbb{S}^2)^9$:

$$\Delta(v^i, v^j) = \textstyle\sum_{l=1}^{9} \left[\Delta(v_l^i, v_l^j)\right]^2 \quad .$$

Finally, the Pose kernel is obtained by adapting the Gaussian kernel to this metric:

$$\phi(v^i, v^j) = \exp\left(-\tfrac{1}{2\sigma^2} \Delta(v^i, v^j)\right) \quad .$$

We calibrate the pose kernel parameter $\sigma$ and the flexible margin parameter $C$ used in the SVM optimization by using an out-of-sample approach to evaluate the quality of the parameters, and a gradient descent algorithm to find the optimal parameters. In our experiments, the best results were obtained when $\sigma \approx 0.3$ and $C \approx 2.5$.

*Classification.* Given a query pose represented by its descriptor $\mathbf{v}$, each classifying function $\hat{f}_p$ returns positive values if $\mathbf{v}$ is likely to be of a key pose class $\mathbf{p}$, and negative values otherwise. Intuitively, the larger the value, the higher the confidence. Such use of the SVM-distance for classification confidence has been successful in other contexts, as for intelligent galleries design [28]. The key pose classifier then chooses the key pose associated with

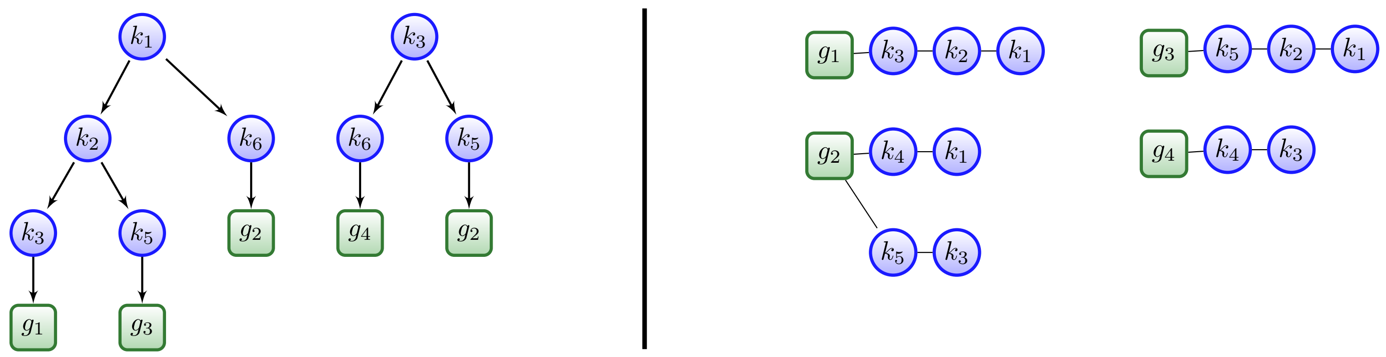

Figure 5: Gesture learning machine example: a forest containing five key poses (left); and the four gestures represented by the forest (right).

the classifier returning the highest confidence among all key pose classifiers. Formally, the key pose class from descriptor $\mathbf{v}$ is then

$$\hat{f}(\mathbf{v}) = \begin{cases} \mathbf{q} = \underset{\mathbf{p}}{\operatorname{argmax}}\, \hat{f}_{\mathbf{p}}(\mathbf{v}) & \text{if } \hat{f}_{\mathbf{q}}(\mathbf{v}) > 0, \\ \text{-1} & \text{otherwise.} \end{cases} \tag{1}$$

Note that if all classifiers return negative values, then the queried pose does not belong to any trained key pose class, causing the key pose classifier to return $-1$. This could simply be an intermediate pose, or a missing pose in the set $\mathcal{K}$.

## 5. Gesture Recognition through Decision Forest

A human body gesture can be represented as a continuous curve over time in the space of all realizable body poses. However, most gestures can be identified by a sequence of just a few key poses [13], which can be seen as a discretization of the continuous gesture curve. Our gesture recognition also uses a supervised learning approach. Instead of SVM, the gesture training set is structured in a decision forest that efficiently recognizes gestures, parsing the key poses sequence of the user's performance in real time.

### *5.1. Defining a gesture*

We presented gestures as sequences $g = \{k_1, \cdots, k_{n_g}\}$ of key poses $k_i \in \mathcal{K}$. Indeed, a gesture is typically identified through a small sequence from two to five key poses. For example, a gesture composed by opening arms and clapping hands may need as few as 4 key poses, as shown in Figure 1. Often, slightly different executions of the same gesture class can lead to different sequences of key poses.

A straightforward representation of a gesture would be a unique sequence $\{k_1, k_2, \cdots, k_{n_g}\}$ of key poses composing the gesture $g$. Although effective, this approach would ignore an alternative way to perform the same gesture.

In this work, we restrict the set of gestures so that the sequence defining a gesture cannot be extended to a longer sequence defining another gesture, i.e. gestures are irreducible. A gesture is then defined uniquely by a set of key poses sequences, different sequences characterizing different user's performances for the same gesture.

This differs slightly from the action graph [10], omitting transition probabilities between key poses, but allowing a decision forest representation, and this will avoid the need for a neutral/initial pose. We optimize the decision forest scheme to efficiently identifies gestures in real time.

### *5.2. Recognizing gestures*

Given a gesture training set composed of key pose sequences, we build a forest whose nodes represent key poses and whose leaves are associated to gestures. Each path in a tree of the forest, from the parent of a leaf $g$ to the root, represents a possible sequence for gesture $g$. Thus, each tree rooted at a key pose $k$ encodes all the gestures whose final key pose is $k$, while each leaf stores a gesture identifier. Note that there are at most as many trees as there are key poses.

During the recognition phase, the key pose classifiers try to recognize key poses performed by the user. Since we do not require a neutral/initial pose, we accumulate the key poses into a circular buffer $\mathcal{B}$ of the last identified key poses. We avoid repeating a key pose in $\mathcal{B}$ by checking if the last added key pose is identical to a newly detected key pose, eventually updating its duration.

Each time a key pose $k$ is recognized, it is inserted in $\mathcal{B}$, and a new search is started at the root of the tree representing gestures ending in $k$. We search down the tree by reversely iterating the buffer starting at $k$. If the preceding element in $\mathcal{B}$ (i.e., the previously detected key pose) is a child of the current node, the search continues. Otherwise, the search fails, as there is no trained key pose sequence which is a suffix of the last performed key pose sequence. If the search reaches a leaf, the gesture stored in that leaf is recognized and reported. In practice, the circular buffer $\mathcal{B}$ does not need to be emptied, but we require that it has enough space to contain the largest gesture.

The choice of storing the gestures back-to-front in the decision forest simplifies the recognition work. In this way, the search is only performed in one tree, avoiding dynamic-programming backlogs. In practice, the trees are relatively thin, i.e. non-matching sequences rapidly fail. Also, this does not require to know which key pose of $\mathcal{B}$ is the first key pose of the next gesture, since we only use the last key pose to start the search.

We emphasize that two different key pose sequences can be safely tagged as the same gesture. Figure 5 shows an example of a simple forest with six key poses and five recognizable gestures. Note how two different sequences of key poses are assigned to the same gesture $g_2$. This is convenient in applications, for example when a gesture performed with the right hand is considered to be the same as the one realized with the left hand. Also, it is possible to perform the same gesture with one pass through slightly different sequences of key poses.

Finally, our formulation does not allow sub-gestures of gestures, to avoid ambiguity. However, if sub-gestures must be identified, one can easily adapt our method by representing a sub-gesture as an interior node of the tree, along the path of its complete gesture.

### *5.3. Time constraints*

Until now, the execution speed of the gesture is not represented, while it can matter in several applications such as dance movements Raptis *et al.* [20]. In our gesture representation, we can include the interval between consecutive key poses as a time vector $[t_1, t_2, \cdots, t_{n-1}]$ associated to the circular buffer $\mathcal{B}$. Thus, the same key pose sequence performed at different speeds can be considered as executions of gestures belonging to different classes, or even not considered a gesture at all.

We store in each leaf of the forest one or more time vectors of the gestures sharing the same corresponding key pose sequence. Two gestures with the same key pose sequence but different timings would lead to sibling leaves. When searching for gestures in the decision forest, we choose or discard a gesture based on the time vectors with two criteria. Let $\mathbf{t}_i$ be a time vector stored in a leaf representing the gesture $g_i$ and $\mathbf{t}$ the time vector of the current user's performance. If $\|t_i - t\|_\infty > T$ for all time vectors stored with $g_i$, where T is a small threshold, then $g_i$ is discarded. Among all non-discarded gestures on the sibling leaves, the gesture $g_i$ that minimizes $\|t_i - t\|_1$ is chosen as the recognized gesture.

## 6. Results

We present in this section the experiments we have performed to validate the robustness and to evaluate the performance of the proposed method. We also compare our method to two state of the art methods, and discuss its improvements and limitations.

### *6.1. Experiment setup*

To evaluate the robustness of our key pose learning machine, we designed a key pose set $\mathcal{K}$ composed of 18 key poses to be used in all tests. One example of each key pose class is shown in Figure 6. Note that we focused mainly on superior limbs poses, which are more relevant for natural user interfaces. To create the key pose training set $\mathcal{T}$, a single trainer performed around 30 examples of each key pose, resulting in approximately 600 examples of key poses.

Table 1: Trained gestures and average recognition rate from experiments with 10 individuals. Key poses are described in Figure 6. Note that some gestures have multiple definitions; the good bye gesture is time constrained; and raising the right arm laterally can be classified as one of two different gestures (quick or slow), depending on the execution speed, as illustrated in Figure 7.

| **gesture** | **id** | **key pose seq.** | **rec. rate** |
|---|---|---|---|
| Open-Clap | $g_1$ | $k_1,\ k_4,\ k_7$ | **99%** |
| Open Arms | $g_2$ | $k_1,\ k_7,\ k_4$ | **97%** |
| Turn Next Page | $g_3$ | $k_1,\ k_2,\ k_5,\ k_1$<br>$k_1,\ k_6,\ k_3,\ k_1$ | **94%** |
| Turn Previous Page | $g_4$ | $k_1,\ k_5,\ k_2,\ k_1$<br>$k_1,\ k_3,\ k_6,\ k_1$ | **95%** |
| Good Bye<br>($k_{11}$ time constraint: 1 sec) | $g_5$ | $k_1,\ k_{11}$ | **92%** |
| Quick Raise Right Arm<br>(time constraint: 1 sec) | $g_6$ | $k_1,\ k_2,\ k_8$ | **82%** |
| Slow Raise Right Arm<br>(time constraint: 2 secs) | $g_7$ | $k_1,\ k_2,\ k_8$ | **88%** |
| Lower Right Arm | $g_8$ | $k_8,\ k_2,\ k_1$ | **82%** |
| Japanese Greeting | $g_9$ | $k_1,\ k_{14},\ k_1$ | **100%** |
| Put Hands Up Front | $g_{10}$ | $k_1,\ k_5,\ k_{18}$<br>$k_1,\ k_5,\ k_8$<br>$k_1,\ k_5,\ k_{11},\ k_8$<br>$k_1,\ k_8$ | **96%** |
| Put Hands Up | $g_{11}$ | $k_1,\ k_4,\ k_{10}$ | **100%** |

Then, we designed a set $\mathcal{G}$ of 11 gestures, as shown in Table 1. We asked the single trainer to perform 10 times each gesture from this set, and captured the sequences of key poses. We also considered time constraints in gesture $g_5, g_6, g_7$ to validate our formulation. In the GoodBye gesture $g_5$, the last pose $k_{11}$ must be kept for 1 second to characterize that gesture.

Note that $\mathcal{K}$ restricts the set of recognizable gestures $\mathcal{G}$ to all finite combinations of key poses from $\mathcal{K}$. Thus, the design of $\mathcal{K}$ must take into account the desired recognizable gesture set $\mathcal{G}$, exactly as complying with a step-by-step tutorial.

### *6.2. Key pose recognition*

*Robustness.* We asked the trainer and 10 inexperienced individuals to perform all trained key poses to evaluate the recognition rate of our classifiers. Each individual performed each key pose 10 times, with results reported in Table 2. The key pose learning machine was able to recognize the users' key poses in most cases, achieving an average recognition rate of 96.41%. Even in similar poses, like $k_{13}$ and $k_{17}$, the machine succeeded in classifying the right pose in most examples. We observed that most failures were in challenging poses to the skeleton tracker, where the depth image suffers from occlusion, such as the pose $k_{18}$. Also, one female individual $u_{10}$ with large frizzy hair

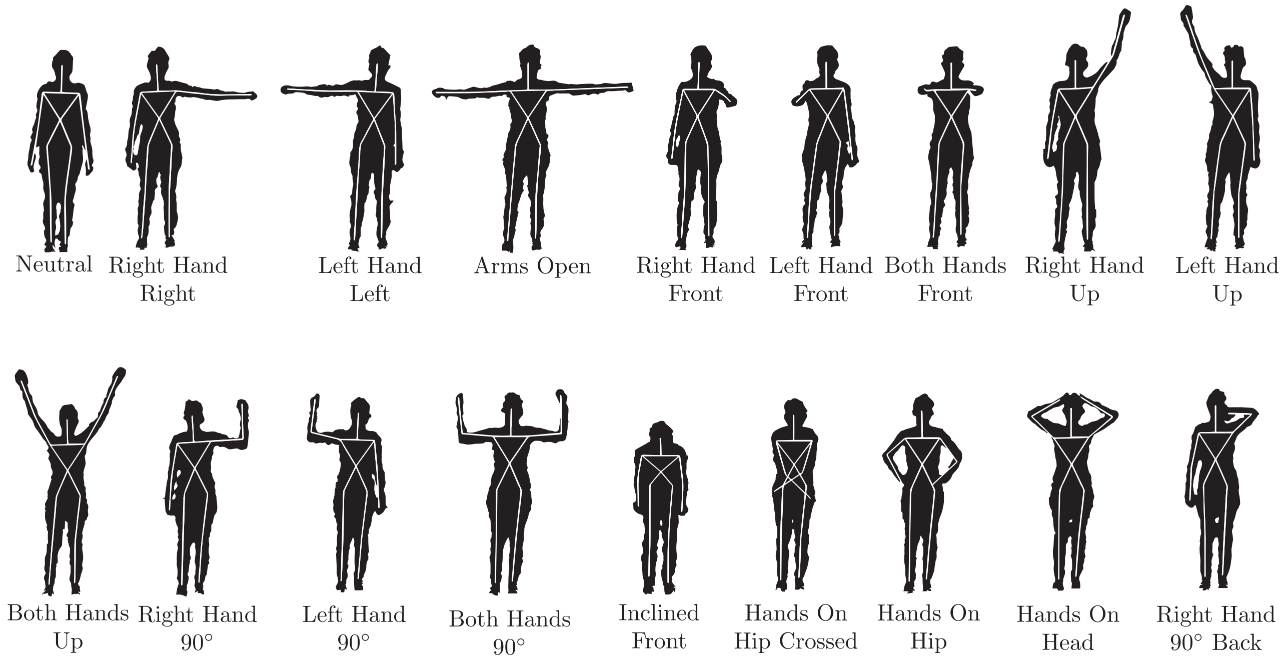

Figure 6: Reference key poses for training set $\mathcal{T}$.

was troublesome for the skeleton tracker, and consequently to our method, still leading to 86% recognition rate. However, when she tied her hair, closer to average results were achieved.

*Stability.* To verify the stability of the key pose learning machine, we performed out-of-sample tests. In this experiment, we removed at random 20% of the data from our training set $\mathcal{T}$, computed the SVM classifiers and tried to classify the removed poses. After executing this experiment 10 times, the average number of false classifications was only 4.16%, while 3.45% could not be classified.

### *6.3. Gesture recognition*

To check the robustness of the gesture learning machine, we verbally described the trained gestures to 10 individuals. Then, we measured the recognition rate of the machine when the individuals executed each trained gestures 10 times. Excellent results were obtained in the majority of the gestures, while more tricky gestures achieved satisfactory results, as shown in Table 1. In particular, time constrained gestures present the larger performance variations, although the recognition rates for the time constrained gestures $g_5$, $g_6$ and $g_7$ are above 80%. Comparing the results of gestures $g_6$ and $g_7$ (Figure 7), which are represented by the same key poses but performed at different speeds, we observe that our method achieves better recognition rates when gestures are performed slower.

### *6.4. Performance*

During the preprocessing phase, the only, albeit small, bottleneck is computing the SVM binary classifiers functions. For a large training set with around 2 000 key pose examples of 18 classes, the 18 functions were computed in 3.9 secs, with an average of 105 support vectors per binary classifier. Note that these functions only need to be computed once, as long as the training set remains unchanged.

During training and recognition phases of our experiments, performance allowed real time interaction: the key pose learning machine, composed of several SVM binary classifiers, was easily capable of recognizing key poses at 30fps (the maximum Kinect sensor frame rate) on a Core 2 Duo laptop at 2.53$GHz$.

Also, on the one hand, most gestures are composed of just a few key poses, generating decision trees with very low depths. On the other hand, each trees width depends on the number of trained gestures, which is also a low number in most cases. The decision forest formulation leads to very low search complexity in practice and did not impact on the total execution time.

### *6.5. Comparison*

We compared our approach to two state of the art methods, and the conference version of this work that did not use the key pose kernel [15]. In the work of Li *et al.* [11], three subsets of eight gestures each are selected from a dataset of 20 gestures, as shown in Table 3.

The same dataset and subsets are used to compare with [11], [27] and our method, although we extract skeletons from the original depth maps. For each subset AS1, AS2 and AS3, we trained key poses and gestures using the skeletons of half of the subjects from the dataset. We manually labeled key poses for each performance, feeding the gestures learning machine from the resulting pose sequence. Then, we performed cross subject tests, trying to recog-

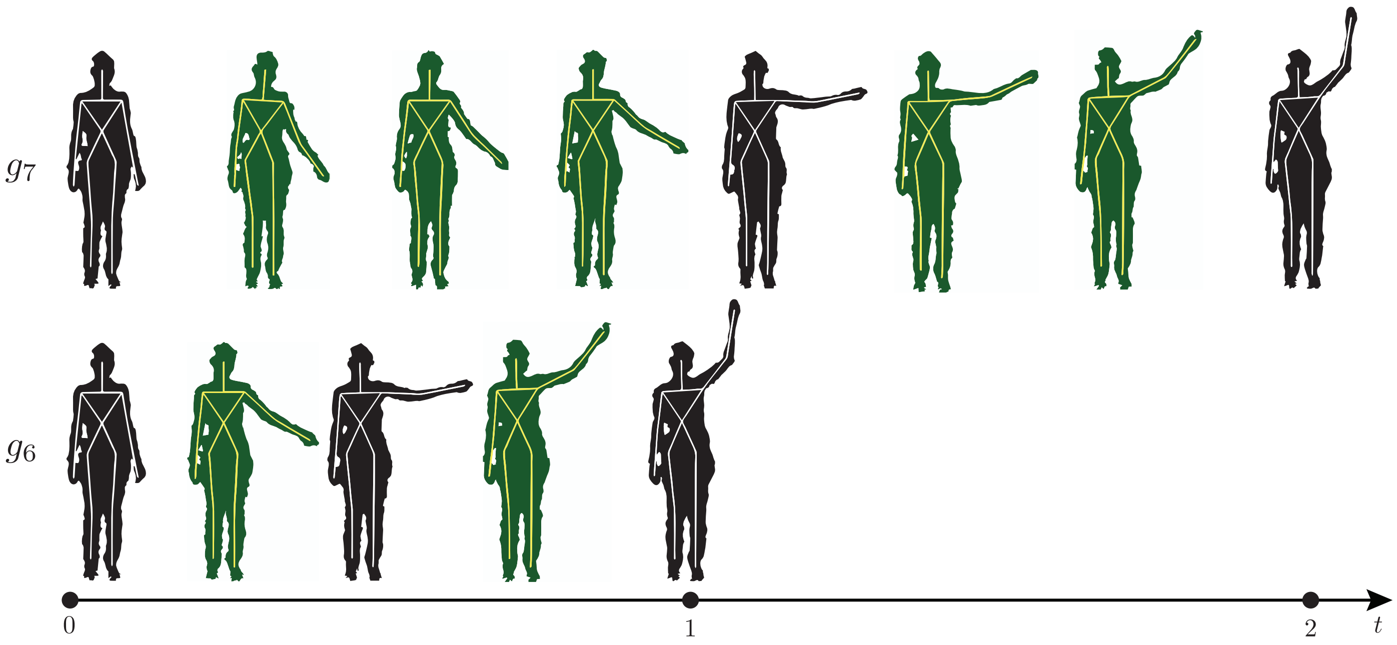

Figure 7: Time constrained gestures: the machine is capable of recognizing and distinguishing between gestures composed of the same key poses, but performed at different speeds. In our tests, a trainer taught the machine that a slow lateral raise of its right arm (top) is a different gesture than a quick raise (bottom). High recognition rates were obtained, as shown in Table 1.

Table 2: Number of correctly recognized key poses with 10 individuals, performing variations of each key pose 10 times. $u_1$ is the trainer user, $u_{10}^1$ is the female with long frizzy hair, showing the worst results, and $u_{10}^2$ is the same female with tied hair, achieving better results. All key poses were very well recognized by the classifiers, except for the last one, due to skeleton tracking issues with occlusion.

| **key pose** | **id** | **recognized key poses per user** | | | | | | | | | | | **total** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $u_1$ | $u_2$ | $u_3$ | $u_4$ | $u_5$ | $u_6$ | $u_7$ | $u_8$ | $u_9$ | $u_{10}^1$ | $u_{10}^2$ | **(%)** |
| Neutral | $k_1$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Right Hand Right | $k_2$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Left Hand Left | $k_3$ | 10 | 10 | 10 | 10 | 10 | 10 | 9 | 10 | 10 | 10 | 10 | **100.00** |
| Arms Open | $k_4$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 9 | 10 | 7 | 10 | **96.36** |
| Right Hand Front | $k_5$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 8 | 9 | **97.27** |
| Left Hand Front | $k_6$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 9 | 10 | **99.09** |
| Both Hands Front | $k_7$ | 10 | 10 | 9 | 10 | 10 | 10 | 10 | 10 | 10 | 9 | 10 | **98.18** |
| Right Hand Up | $k_8$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Left Hand Up | $k_9$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 9 | 10 | **99.09** |
| Both Hands Up | $k_{10}$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Right Hand 90° | $k_{11}$ | 10 | 9 | 9 | 10 | 10 | 10 | 10 | 10 | 10 | 8 | 10 | **96.36** |
| Left Hand 90° | $k_{12}$ | 10 | 10 | 10 | 10 | 10 | 8 | 10 | 10 | 10 | 7 | 10 | **95.45** |
| Both Hands 90° | $k_{13}$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Inclined Front | $k_{14}$ | 10 | 10 | 9 | 10 | 10 | 8 | 10 | 10 | 10 | 5 | 7 | **90.00** |
| Hands-on-Hip Crossed | $k_{15}$ | 9 | 8 | 8 | 10 | 8 | 10 | 10 | 9 | 8 | 8 | 9 | **88.18** |
| Hand-On-Hip | $k_{16}$ | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | **100.00** |
| Hands on Head | $k_{17}$ | 10 | 10 | 10 | 8 | 10 | 10 | 8 | 9 | 10 | 10 | 6 | **91.81** |
| Right Hand 90° Back | $k_{18}$ | 8 | 10 | 9 | 9 | 7 | 9 | 9 | 10 | 10 | 5 | 7 | **84.54** |
| **total (%)** | | **98.3** | **98.3** | **96.7** | **98.3** | **97.2** | **97.2** | **98.3** | **97.8** | **98.9** | **86.1** | **93.3** | |

nize non-trained gesture samples using the corresponding training set.

The obtained results are shown in Table 4. Note that excellent results were obtained in AS1 and AS3, outperforming [11] and [27], while AS2 performed badly. The low performance of AS2 recognition was mainly due to three gestures composed of not very distinctive key poses: *draw x*, *draw circle* and *draw tick*. While these gestures required very subtle movements, many subjects executed them through slightly different poses.

### *6.6. Limitations*

Most robustness issues were mainly due to two reasons: skeleton tracking and small movements, like drawing an x. Using a single Kinect, the user must be in front of the sensor, since side positions can occlude joints, degrading

Table 3: Recognition rate comparison on the gesture dataset of Li *et al.* [11]

| **AS1** | **AS2** | **AS3** |
|---|---|---|
| Horizontal arm wave | High arm wave | High throw |
| Hammer | Hand catch | Forward kick |
| Forward punch | Draw x | Side kick |
| High throw | Draw tick | Jogging |
| Hand clap | Draw circle | Tennis swing |
| Bend | Two hand wave | Tennis serve |
| Pickup & throw | Side boxing | Pickup & throw |

Table 4: Comparison of recognition rate through a cross-subject test.

| **Gesture subset** | **Li** *et al.* [11] | **Vieira** *et al.* [27] | **Euclidean kernel** [15] | **Pose kernel** |
|---|---|---|---|---|
| AS1 | 72.9% | 84.7% | 93.5% | **96.0%** |
| AS2 | 71.9% | 81.3% | 52.0% | **57.1%** |
| AS3 | 79.2% | 88.4% | 95.4% | **97.3%** |
| Average | 74.7% | 84.8% | 80.3% | **83.5 %** |

the skeleton. Moreover, the skeleton tracker can generate different skeletons for different individuals performing the same pose due to the skeleton extraction we use [23]. These differences can degrade the invariance of pose descriptors, with frequent outlier skeletons. As opposed to the Action Graph [10], our method is limited to gestures composed of distinctive key poses. Gestures that requires subtle movements can be troublesome for our learning machines. Finally, relying on key pose design and training may not be the friendliest solution for an occasional user.

## 7. Future Work

As the skeleton extraction and tracking algorithms still cope with a large amount of noise from the sensor, robustness is a main issue for future work. A common problem for these algorithms is the 3D occlusion of some joint positions, requiring the user to face the camera to avoid outlier skeletons. Working with two or more Kinect sensors could be of help to robustly capture and process the skeleton stream.

The algorithm introduced here may be improved in two different directions. First, the use of time in recognizing gestures may be improved to distinguish complex gestures, using complementary SVM machines, maybe incorporating velocity of the joints. Second, automatic key pose generation from unrecognized gestures may greatly ease the usability of the interface. In this setting, the computer should be able to decide the best set of key poses to train, in order to get good results in gesture recognition.

The authors would like to thank Brazilian institutions CNPq, FAPEAL, FAPERJ and FAPEMIG for partially financing this research.

## References

[1] Bobick, A., Davis, J., 2001. The recognition of human movement using temporal templates. TPAMI 23.

[2] Cao, L., Liu, Z., Huang, T., 2010. Cross-dataset action detection. In: CVPR.

[3] Chen, D.-Y., Liao, H.-Y. M., Shih, S.-W., 2007. Human action recognition using 2-D spatio-temporal templates. In: Multimedia and Expo.

[4] Davis, J. W., Tyagi, A., 2006. Minimal-latency human action recognition using reliable-inference. Image and Vision Computing 24.

[5] Forbes, K., Fiu, E., 2005. An efficient search algorithm for motion data using weighted pca. In: SCA. pp. 67–76.

[6] Kovar, L., 2004. Automated extraction and parameterization of motions in large data sets. In: Siggraph. Vol. 23. pp. 559–568.

[7] Kovashka, A., Grauman, K., 2010. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In: CVPR.

[8] Lab, C. M. G., 2001. Carnegie mellon university motion capture database.
URL `http://mocap.cs.cmu.edu`

[9] Laptev, I., Lindeberg, T., 2003. Space-time interest points. In: ICCV.

[10] Li, W., Zhang, Z., Liu, Z., 2008. Expandable data-driven graphical modeling of human actions based on salient postures. Circuits and Systems for Video Technology 18 (11).

[11] Li, W., Zhang, Z., Liu, Z., 2010. Action recognition based on a bag of 3d points. In: CVPR Workshop for Human Communicative Behavior Analysis.

[12] Liu, Z., 2011. MSR action recognition datasets and codes.
URL `http://research.microsoft.com/ zliu/ActionRecoRsrc`

[13] Lv, F., Nevatia, R., 2007. Single view human action recognition using key pose matching and Viterbi path searching. In: CVPR. pp. 1–8.

[14] Mercier, G., Lennon, M., 2003. Support vector machines for hyperspectral image classification with spectral-based kernels. In: GRSS. Vol. 1. pp. 288–290.

[15] Miranda, L., Vieira, T., Martínez, D., Lewiner, T., Vieira, A. W., Campos, M. F. M., 2012. Real-time gesture recognition from depth data through key poses learning and decision forests. In: Sibgrapi. IEEE, Ouro Preto, MG, pp. 268–275.

[16] Müller, M., Baak, A., Seidel, H.-P., 2009. Efficient and robust annotation of motion capture data. In: SCA. pp. 17–26.

[17] Müller, M., Röder, T., 2006. Motion templates for automatic classification and retrieval of motion capture data. In: SCA. pp. 137–146.

[18] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., Weber, A., 2007. Documentation mocap database HDM05. Tech. Rep. CG-2007-2, Universität Bonn.

[19] Niebles, J. C., Chen, C. W., Fei-Fei, L., 2010. Modeling temporal structure of decomposable motion segments for activity classification. In: ECCV.

[20] Raptis, M., Kirovski, D., Hoppe, H., 2011. Real-time classification of dance gestures from skeleton animation. In: SCA. pp. 147–156.

[21] Reyes, M., Domínguez, G., Escalera, S., 2011. Feature weighting in dynamic time warping for gesture recognition in depth data. In: ICCV Workshop on Consomer Depth Cameras for Computer Vision.

[22] Schölkopf, B., Smola, A. J., 2002. Learning with Kernels. MIT.

[23] Shotton, J., Fitzgibbon, A. W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A., 2011. Real-time human pose recognition in parts from single depth images. In: CVPR. pp. 1297–1304.

[24] Sun, J., Wu, X., Yan, S., Cheong, L., Chua, T., Li, J., 2009. Hierarchical spatio-temporal context modeling for action recognition. In: CVPR.

[25] Vapnik, V., 2000. The Nature of Statistical Learning Theory. Springer.

[26] Vieira, A. W., Lewiner, T., Schwartz, W., Campos, M. F. M.,

2012. Distance matrices as invariant features for classifying mocap data. In: ICPR. IEEE, Tsukuba Science City, Japan.

[27] Vieira, A. W., Nascimento, E. R., Oliveira, G. L., Liu, Z., Campos, M. M., 2012. STOP: Space-time occupancy patterns for 3d action recognition from depth map sequences. In: CIARP.

[28] Vieira, T., Bordignon, A. L., Peixoto, A., Tavares, G., Lopes, H., Velho, L., Lewiner, T., 2009. Learning good views through intelligent galleries. Computer Graphics Forum 28 (2), 717–726.

[29] Weinland, D., Boyer, E., 2005. Action recognition using exemplar-based embedding. In: CVPR.

[30] Yang, X., Tian, Y., 2012. Eigenjoints-based action recognition using naïve-Bayes-nearest-neighbor. In: CVPR Workshops. IEEE, pp. 14–19.

[31] Yang, X., Zhang, C., Tian, Y., 2012. Recognizing actions using depth motion maps-based histograms of oriented gradients. In: Multimedia. ACM, pp. 1057–1060.

[32] Zhang, J., Gong, S., 2010. Action categorization with modified hidden conditional random field. Pattern Recognition 43 (1), 197–203.