Journal of Machine Learning Research 14 (2013) 2617-2640 Submitted 1/13; Revised 5/13; Published 9/13

Keep It Simple And Sparse: Real-Time Action Recognition

Sean Ryan Fanello[*] SEAN.FANELLO@IIT.IT
Ilaria Gori[*] ILARIA.GORI@IIT.IT
Giorgio Metta GIORGIO.METTA@IIT.IT
iCub Facility
Istituto Italiano di Tecnologia
Genova, Via Morego 30, 16163, Italia

Francesca Odone FRANCESCA.ODONE@UNIGE.IT
Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi
Università degli Studi di Genova
Genova, Via Dodecaneso 35, 16146, Italia

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

Sparsity has been showed to be one of the most important properties for visual recognition purposes. In this paper we show that sparse representation plays a fundamental role in achieving one-shot learning and real-time recognition of actions. We start off from RGBD images, combine motion and appearance cues and extract state-of-the-art features in a computationally efficient way. The proposed method relies on descriptors based on 3D Histograms of Scene Flow (3DHOFs) and Global Histograms of Oriented Gradient (GHOGs); adaptive sparse coding is applied to capture high-level patterns from data. We then propose a simultaneous on-line video segmentation and recognition of actions using linear SVMs. The main contribution of the paper is an effective real-time system for one-shot action modeling and recognition; the paper highlights the effectiveness of sparse coding techniques to represent 3D actions. We obtain very good results on three different data sets: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and a data set created for human-robot interaction purposes. Finally we demonstrate that our system is effective also in a human-robot interaction setting and propose a memory game, "All Gestures You Can", to be played against a humanoid robot.

Keywords: real-time action recognition, sparse representation, one-shot action learning, human robot interaction

1. Introduction

Action recognition as a general problem is a very fertile research theme due to its strong applicability in several real world domains, ranging from video-surveillance to content-based video retrieval and video classification. This paper refers specifically to action recognition in the context of Human-Machine Interaction (HMI), and therefore it focuses on whole-body actions performed by a human who is standing at a short distance from the sensor.

Imagine a system capable of understanding when to turn the TV on, or when to switch the lights off on the basis of a gesture; the main requirement of such a system is an easy and fast learn-

∗. S.R. Fanello and I. Gori contributed equally to this work.

ing and recognition procedure. Ideally, a single demonstration suffices to teach the system a new gesture. More importantly, gestures are powerful tools, through which languages can be built. In this regard, developing a system able to communicate with deaf people, or to understand paralyzed patients, would represent a great advance, with impact on the quality of life of impaired people. Nowadays these scenarios are likely as a result of the spread of imaging technologies providing real-time depth information at consumer's price (as for example the Kinect (Shotton et al., 2011) by Microsoft); these depth-based sensors are drastically changing the field of action recognition, enabling the achievement of high performance using fast algorithms.

Following this recent trend we propose a *complete system based on RGBD video sequences*, which models actions *from one example only*. Our main goal is to recognize actions in real-time with high accuracy; for this reason we design our system accounting for good performance as well as low computational complexity. The method we propose can be summarized as follows: after segmentation of the moving actor, we extract two types of features from each image, namely, Global Histograms of Oriented Gradient (GHOGs) to model the shape of the silhouette, and 3D Histograms of Flow (3DHOFs) to describe motion information. We then apply a sparse coding stage, which allows us to take care of noise and redundant information and produces a compact and stable representation of the image content. Subsequently, we summarize the action within adjacent frames by building feature vectors that describe the feature evolution over time. Finally, we train a Support Vector Machine (SVM) for each action class.

Our framework can segment and recognize actions accurately and in real-time, even though they are performed in different environments, at different speeds, or combined in sequences of multiple actions. Furthermore, thanks to the simultaneous appearance and motion description complemented by the sparse coding stage, the method provides a one-shot learning procedure. These functions are shown on three different experimental settings: a benchmark data set for one-shot action learning (the ChaLearn Gesture Data Set), an in-house data set acquired by a Kinect sensor including complex actions and gestures differing by small details, and an implementation of the method on a humanoid robot interacting with humans.

In order to demonstrate that our system can be efficiently engaged in real world scenarios, we developed a real-time memory game against a humanoid robot, called "All Gestures You Can" (Gori et al., 2012). Our objective in designing this interaction game is to stress the effectiveness of our gesture recognition system in complex and uncontrolled settings. Nevertheless, our long term goal is to consider more general contexts, which are beyond the game itself, such as rehabilitation and human assistance. Our game may be used also with children with memory impairment, for instance the Attention Deficit/Hyperactivity Disorder (ADHD) (Comoldi et al., 1999). These children cannot memorize items under different conditions, and have low performances during implicit and explicit memory tests (Burden and Mitchell, 2005). Interestingly, Comoldi et al. (1999) shows that when ADHD children were assisted in the use of an appropriate strategy, they performed the memory task as well as controls. The game proposed in this paper could be therefore used to train memory skills to children with attention problems, using the robot as main assistant. The interaction with the robot may increase their motivation to maintain attention and help with the construction of a correct strategy.

The paper is organized as follows: in Section 2 we briefly review the state of the art. In Section 3 sparse representation is presented; Section 4 describes the complete modeling and recognition pipeline. Section 5 validates the approach in different scenarios; Section 6 shows a real application

in the context of Human Robot Interaction (HRI). Finally, Section 7, presents future directions and possible improvements of the current implementation.

2. Related Work

The recent literature is rich of algorithms for gesture, action, and activity recognition—we refer the reader to Aggarwal and Ryoo (2011) and Poppe (2010) for a complete survey of the topic. Even though many theoretically sound, good performing and original algorithms have been proposed, to the best of our knowledge, none of them fulfills at the same time *real-time*, *one-shot learning* and *high accuracy* requirements, although such requirements are all equally important in real world application scenarios.

Gesture recognition algorithms differ in many aspects. A first classification may be done with respect to the overall structure of the adopted framework, that is, how the recognition problem is modeled. In particular, some approaches are based on machine learning techniques, where each action is described as a complex structure; in this class we find methods based on Hidden Markov Models (Malgireddy et al., 2012), Coupled Hidden Semi-Markov models (Natarajan and Nevatia, 2007), action graphs (Li et al., 2010) or Conditional Markov Fields (Chatzis et al., 2013). Other methods are based on matching: the recognition of actions is carried out through a similarity match with all the available data, and the most similar datum dictates the estimated class (Seo and Milanfar, 2012; Mahbub et al., 2011).

The two approaches are different in many ways. Machine learning methods tend to be more robust to intra-class variations, since they distill a model from different instances of the same gesture, while matching methods are more versatile and adapt more easily to one-shot learning, since they do not require a batch training procedure. From the point of view of data representation, the first class of methods usually extracts features from each frame, whereas matching-based methods try to summarize all information extracted from a video in a single feature vector. A recent and prototypical example of machine learning method can be found in Malgireddy et al. (2012), which proposes to extract local features (Histograms of Flow and Histograms of Oriented Gradient) on each frame and apply a bag-of-words step to obtain a global description of the frame. Each action is then modeled as a multi channel Hidden Markov Model (mcHMM). Although the presented algorithm leads to very good classification performance, it requires a computationally expensive offline learning phase that cannot be used in real-time for one-shot learning of new actions. Among the matching-based approaches, Seo and Milanfar (2012) is particularly interesting: the algorithm extract a new type of features, referred to as *3D LSKs*, from space-time regression kernels, particularly appropriate to identify the spatio-temporal geometric structure of the action; it then adopts the Matrix Cosine Similarity measure (Shneider and Borlund, 2007) to perform a robust matching. Another recent method following the trend of matching-based action recognition algorithms is Mahbub et al. (2011); in this work the main features are standard deviation on depth (STD), Motion History Image (MHI) (Bobick and Davis, 2001) and a 2D Fourier Transformation in order to map all information in the frequency domain. This procedure shows some benefits, for instance the invariance to camera shifts. For the matching step, a simple and standard correlation measure is employed. Considering this taxonomy, the work we propose falls within the machine learning approaches, but addresses specifically the problem of one-shot learning. To this end we leverage on the richness of the video signal used as a training example and on a dictionary learning approach to obtain an effective and distinctive representation of the action.

An alternative to classifying gesture recognition algorithms is based on the data representation of gesture models. In this respect there is a predominance of features computed on local areas of single frames (local features), but also holistic features are often used on the whole image or on a region of interest. Among the most known methods, it is worth mentioning the spatio-temporal interesting points (Laptev and Lindeberg, 2003), spatio-temporal Hessian matrices (Willems et al., 2008), Gabor Filters (Bregonzio et al., 2009), Histograms of Flow (Fanello et al., 2010), Histograms of Oriented Gradient (Malgireddy et al., 2012), semi-local features (Wang et al., 2012), combination of multiple features (Laptev et al., 2008), Motion History Image (MHI) (Bobick and Davis, 2001), Space-Time shapes (Gorelick et al., 2007), Self-Similarity Matrices (Efros et al., 2003). Also, due to the recent diffusion of real-time 3D vision technology, 3D features have been recently employed (Gori et al., 2012). For computational reasons as well as the necessity of specific invariance properties, we adopt global descriptors, computed on a region of interest obtained through motion segmentation. We do not rely on a single cue but rather combine motion and appearance similarly to Malgireddy et al. (2012).

The most similar works to this paper are in the field of HMI as for example Lui (2012) and Wu et al. (2012): they both exploit depth information and aim at one-shot learning trying to achieve low computational cost. The first method employs a nonlinear regression framework on manifolds: actions are represented as tensors decomposed via Higher Order Singular Value Decomposition. The underlying geometry of tensor space is used. The second one extracts Extended-MHI as features and uses Maximum Correlation Coefficient (Hirschfeld, 1935) as classifier. Features from RBG and Depth streams are fused via a Multiview Spectral Embedding (MSE). Differently from these works, our approach aims specifically to obtain an accurate real-time recognition from one video example only.

We conclude the section with a reference to some works focusing on continuous action or activity recognition (Ali and Aggarwal, 2001; Green and Guan, 2004; Liao et al., 2006; Alon et al., 2009). In this case training and test videos contain many sequential gestures, therefore the temporal segmentation of videos becomes fundamental. Our work deals with continuous action recognition as well, indeed the proposed framework comprehends a novel and robust temporal segmentation algorithm.

3. Visual Recognition with Sparse Data

One-shot learning is a challenging requirement as the small quantity of training data makes the modeling phase extremely hard. For this reason, in one-shot learning settings a careful choice of the data representation is very important. In this work we rely on sparse coding to obtain a compact descriptor with a good discriminative power even if it is derived from very small data sets.

The main concept behind sparse coding is to approximate an input signal as a linear combination of a few components selected from a dictionary of basic elements, called atoms. We refer to *adaptive sparse coding* when the coding is driven by data. In this case, we require a *dictionary learning* stage, where the dictionary atoms are learnt (Olshausen and Fieldt, 1997; Yang et al., 2009; Wang et al., 2010).

The motivations behind the use of image coding arise from biology: there is evidence that similar signal coding happens in the neurons of the primary visual cortex (V1), which produces sparse and overcomplete activations (Olshausen and Fieldt, 1997). From the computational point of view the objective is to find an overcomplete model of images, unlike methods such as PCA, which

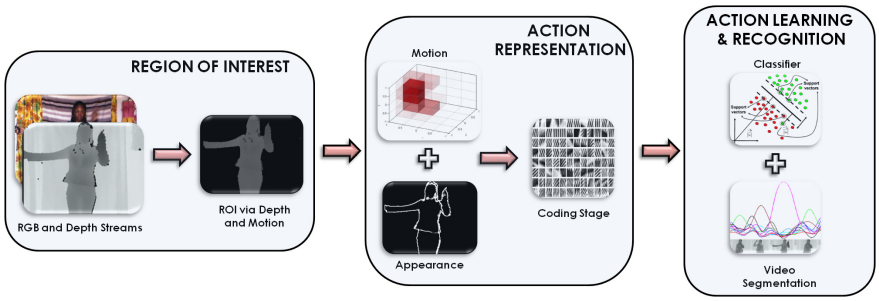

Figure 1: Overview of the recognition system, where video segmentation and classification are performed simultaneously.

aims at finding a number of components that is lower than the data dimensionality. Overcomplete representation techniques have become very popular in applications such as denoising, inpainting, super-resolution, segmentation (Elad and Aharon, 2006; Mairal et al., 2008b,a) and object recognition (Yang et al., 2009). In this work we assess their effectiveness also for gesture recognition. Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}^{n \times m}$ be the matrix whose m columns $\mathbf{x}_i \in \mathbb{R}^n$ are the feature vectors. The goal of adaptive sparse coding is to learn a dictionary $\mathbf{D}$ (a $n \times d$ matrix, with d the dictionary size and n the feature vector size) and a code $\mathbf{U}$ (a $d \times m$ matrix) that minimize the reconstruction error:

$$\min_{\mathbf{D},\mathbf{U}} \|\mathbf{X} - \mathbf{D}\mathbf{U}\|_F^2 + \lambda \|\mathbf{U}\|_1, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm. As for the sparsity, it is known that the L_1-norm yields to sparse results while being robust to signals perturbations. Other penalties such as the L_0-norm could be employed, however the problem of finding a solution becomes NP-hard and there is no guarantee that greedy algorithms reach the optimal solution. Notice that fixing $\mathbf{U}$, the above optimization reduces to a least square problem, whilst, given $\mathbf{D}$, it is equivalent to linear regression with the sparsifying norm L_1. The latter problem is referred to as a feature selection problem with a known dictionary (Lee et al., 2007). One of the most efficient algorithms that converges to the optimal solution of the problem in Equation 1 for a fixed $\mathbf{D}$, is the *feature-sign search* algorithm (Lee et al., 2007). This algorithm searches for the sign of the coefficients $\mathbf{U}$; indeed, considering only non-zero elements the problem is reduced to a standard unconstrained quadratic optimization problem (QP), which can be solved analytically. Moreover it performs a refinement of the signs if they are incorrect. For the complete procedure we refer the reader to Lee et al. (2007).

In the context of recognition tasks, it has been proved that a sparsification of the data representation improves the overall classification accuracy (see for instance Guyon and Elisseeff, 2003; Viola and Jones, 2004; Destrero et al., 2009 and references therein). In this case sparse coding is often cast into a *coding-pooling* scheme, which finds its root in the Bag of Words paradigm. In this scheme a *coding operator* is a function $f(\mathbf{x}_i) = \mathbf{u}_i$ that maps $\mathbf{x}_i$ to a new space $\mathbf{u}_i \in \mathbb{R}^k$; when $k > n$ the representation is called overcomplete. The action of coding is followed by a pooling stage, whose purpose is to aggregate multiple local descriptors in a single and global one. Common pooling operators are the max operator, the average operator, or the geometric L_p-norm pooling operator (Feng et al., 2011). More in general, a pooling operator takes the codes located in S regions—for

Figure 2: Region of Interest detection. Left: RGB video frames. Center: depth frames. Right: the detected ROI.

instance cells of the spatial pyramid, as in Yang et al. (2009)—and builds a succinct representation. We define as Y_s the set of locations within the region s. Defining the pooling operator as g, the resultant feature can be rewritten as: $\mathbf{p}_{(s)} = g_{(i \in Y_s)}(\mathbf{u}_{(i)})$. After this stage, a region s of the image is encoded with a single feature vector. The final descriptor of the image is the concatenation of the descriptors $\mathbf{p}_s$ among all the regions. Notice that the effectiveness of pooling is subject to the coding stage. Indeed, if applied on non-coded descriptors, pooling would bring to a drastic loss of information.

4. Action Recognition System

In this section we describe the versatile real-time action recognition system we propose. The system, depicted in Figure 1, consists of three layers, that can be summarized as follows:

- **Region Of Interest detection**: we detect a Region of Interest (ROI), where the human subject is actually performing the action. We use the combination of motion and depth to segment the subject from the background.
- **Action Representation**: each ROI within a frame is mapped into a feature space with a combination of 3D Histogram of Flow (3DHOF) and Global Histogram of Oriented Gradient (GHOG) on the depth map. The resultant 3DHOF+GHOG descriptor is processed via a sparse coding step to compute a compact and meaningful representation of the performed action.
- **Action Learning**: linear SVMs are used on frame buffers. A novel on-line video segmentation algorithm is proposed which allows isolating different actions while recognizing the action sequence.

4.1 Region Of Interest Segmentation

The first step of each action recognition system is to identify correctly where in the image the action is occurring. Most of the algorithms in the literature involve background modeling techniques

(Stauffer and Grimson, 1999), or space-time image filtering in order to extract the interesting spatio-temporal locations of the action (Laptev and Lindeberg, 2003). Other approaches require an *a priori* knowledge of the body pose (Lv and Nevatia, 2007). This task is greatly simplified in our architecture, since in human-machine interaction we can safely assume the human to stand in front of the camera sensors and that there is no other motion in the scene. For each video in the data set, we initially compute the frame differences within consecutive frames in a small buffer, obtaining the set P of pixels that are moving. Relying on this information, we compute the mean depth μ of the pixels belonging to P, which corresponds to the mean depth of the subject within the considered buffer. Thus, for the rest of the video sequence, we select the region of interest as $ROI(t) = \{p_{i,j}(t) : \mu - \varepsilon \leq d(p_{i,j}(t)) \leq \mu + \varepsilon\}$, where $d(p_{i,j}(t))$ is the depth of the pixel $p_{i,j}(t)$ at time t and ε is a tolerance value. In Figure 2 examples of segmentation are shown. We determined empirically that this segmentation procedure achieves better performance with respect to classic thresholding algorithms such as Otsu's method (Otsu, 1979).

4.2 Action Representation

Finding a suitable representation is the most crucial part of any recognition system. Ideally, an image representation should be both *discriminative* and *invariant* to image transformations. A discriminative descriptor should represent features belonging to the same class in a similar way, while it should show low similarity among data belonging to different classes. The invariance property, instead, ensures that image transformations such as rotation, translation, scaling do not affect the final representation. In practice, there is a trade-off between these two properties (Varma and Ray, 2007): for instance, image patches are highly discriminative but not invariant, whereas image histograms are invariant but not discriminative, since different images could be associated to the same representation. When a lot of training data is provided, one could focus on a more discriminative and less invariant descriptor. In our specific case however, where only one training example is provided, invariance is a necessary condition in order to provide discriminant features; this aspect is greatly considered in our method.

From the neuroscience literature it is known that body parts are represented already in the early stages of human development (Mumme, 2001) and that certainly adults have prior knowledge on the body appearance. Many suggests that motion alone can be used to recognize actions (Bisio et al., 2010). In artificial systems this developmental-scale experience is typically not available, although actions can still be represented from two main cues: motion and appearance (Giese and Poggio, 2003). Although many variants of complex features describing human actions have been proposed, many of them imply computationally expensive routines. Differently, we rely on simple features in order to fulfill real-time requirements, and we show that they still have a good discriminative power. In particular we show that a combination of 3D Histograms of Flow (3DHOFs) and Global Histograms of Gradient (GHOGs) models satisfactorily human actions. When a large number of training examples is available, these two features should be able to describe a wide variety of actions, however in one-shot learning scenarios with noisy inputs, they are not sufficient. In this respect, a sparse representation, which keeps only relevant and robust components of the feature vector, greatly simplifies the learning phase making it equally effective.

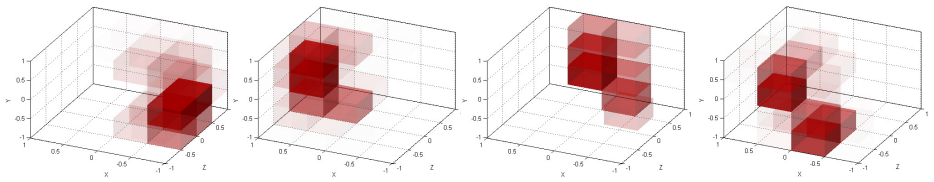

Figure 3: The figure illustrates high level statistics obtained by the proposed scene flow description (3D-HOFs). Starting from the left we show the histogram of the scene flow directions at time t, for a moving hand going on the *Right*, $Left$, *Forward*, *Backward* respectively. Each cuboid represents one bin of the histogram, for visualization purposes we divided the 3D space in $n \times n \times n$ bins with $n = 4$. Filled cuboids represent high density areas.

4.2.1 3D HISTOGRAM OF FLOW

Whereas 2D motion vector estimation has been largely investigated and various fast and effective methods are available today (Papenberg et al., 2006; Horn and Shunk, 1981), the scene flow computation (or 3D motion field estimation) is still an active research field due to the required additional binocular disparity estimation problem. The most promising works are the ones from Wedel et al. (2010), Huguet and Devernay (2007) and Cech et al. (2011); however these algorithms are computationally expensive and may require computation time in the range of 1.5 seconds per frame. This high computational cost is due to the fact that scene flow approaches try to estimate both the 2D motion field and disparity changes. Because of the real-time requirement, we opted for a simpler and faster method that produces a coarser estimation, but is effective for our purposes.

For each frame F_t we compute the 2D optical flow vectors $U(x,y,t)$ and $V(x,y,t)$ for the x and y components with respect to the previous frame F_{t-1}, via the Fanerbäck algorithm (Farnebäck, 2003). Each pixel (x_{t-1}, y_{t-1}) belonging to the ROI of the frame F_{t-1} is reprojected in 3D space $(X_{t-1}, Y_{t-1}, Z_{t-1})$ where the Z_{t-1} coordinate is measured through the depth sensor and X_{t-1}, Y_{t-1} are computed by:

$$\begin{pmatrix} X_{t-1} \\ Y_{t-1} \end{pmatrix} = \begin{pmatrix} \dfrac{(x_{t-1} - x_0) Z_{t-1}}{f} \\ \dfrac{(y_{t-1} - y_0) Z_{t-1}}{f} \end{pmatrix},$$

where f is the focal length and $(x_0, y_0)^T$ is the principal point of the sensor. Similarly, we can reproject the final point (x_t, y_t) of the 2D vector representing the flow, obtaining another 3D vector $(X_t, Y_t, Z_t)^T$. For each pixel of the ROI, we can define the scene flow as the difference of the two 3D vectors in two successive frames F_{t-1} and F_t:

$$\begin{aligned} \mathbf{D} &= (\dot{X}, \dot{Y}, \dot{Z})^T = \\ &= (X_t - X_{t-1}, Y_t - Y_{t-1}, Z_t - Z_{t-1})^T. \end{aligned}$$

Once the 3D flow for each pixel of the ROI at time t has been computed, we normalize it with respect to the $L2$-norm, so that the resulting descriptors $\mathbf{D_1}, \ldots, \mathbf{D_n}$ (n pixels of the ROI) are invariant to the overall speed of the action. In order to extract a compact representation we build a 3D Histogram

of Flow (3DHOF) $\mathbf{z}(t)$ of the 3D motion vectors, where $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ and $\sqrt[3]{n_1}$ is the quantization parameter of the space (i.e., the bin size). In addition we normalize each 3DHOF $\mathbf{z}(t)$ so that $\sum_j z_j(t) = 1$; hence we guarantee that these descriptors are invariant to the subject of interest's scale.

Figure 3 shows that the movements toward different directions reveal to be linearly separable, and the main directions are accurately represented: each cuboid represents one bin of the histogram, and the 3D space is divided in $n \times n \times n$ bins with $n = 4$. It is possible to notice how, in the *Right* direction for example, all the filled bins lay on the semi-space defined by $x < 0$. Similar observations apply all cases.

4.2.2 GLOBAL HISTOGRAM OF ORIENTED GRADIENT

In specific contexts, motion information is not sufficient to discriminate actions, and information on the pose or appearance becomes crucial. One notable example is the American Sign Language (ASL), whose lexicon is based mostly on the shape of the hand. In these cases modeling the shape of a gesture as well as its dynamics is very important. Thus we extend the motion descriptor with a shape feature computed on the depth map. If we assume the subject to be in front of the camera, it is unlikely that the perspective transformation would distort his/her pose, shape or appearance, therefore we can approximately work with invariance to translation and scale. We are interested in characterizing shapes, and the gradient of the depth stream shows the highest responses on the contours, thus studying the orientation of the gradient is a suitable choice. The classical Histograms of Oriented Gradient (HOGs) (Dalal and Triggs, 2005) have been designed for detection purposes and do not show the above-mentioned invariance; indeed dividing the image in cells makes each sub-histogram dependent on the location and the dimension of the object. Furthermore, HOGs exhibit a high spatial complexity, as the classical HOG descriptor belongs to $\mathbb{R}^{(ncells \times nblocks \times n_2)}$. Since we aim at preserving such invariance as well as limiting the computational complexity, we employed a simpler descriptor, the Global Histogram of Oriented Gradient (GHOG). This appearance descriptor produces an overall description of the appearance of the ROI without splitting the image in cells. We compute the histogram of gradient orientations of the pixels on the entire ROI obtained from the depth map to generate another descriptor $\mathbf{h}(t) \in \mathbb{R}^{n_2}$, where n_2 is the number of bins. The scale invariance property is preserved normalizing the descriptor so that $\sum_j h_j(t) = 1$. Computing this descriptor on the depth map is fundamental in order to remove texture information; in fact, in this context, the only visual properties we are interested in are related to shape.

4.2.3 SPARSE CODING

At this stage, each frame F_t is represented by two global descriptors: $\mathbf{z}(t) \in \mathbb{R}^{n_1}$ for the motion component and $\mathbf{h}(t) \in \mathbb{R}^{n_2}$ for the appearance component. Due to the high variability of human actions and to the simplicity of the descriptors, a feature selection stage is needed to catch the relevant information underlying the data and discarding the redundant ones such as background or body parts not involved in the action; to this aim we apply a sparse coding stage to our descriptor.

Given the set of the previously computed 3DHOFs $\mathbf{Z} = [\mathbf{z}(1), \ldots, \mathbf{z}(K)]$, where K is the number of all the frames in the training data, our goal is to learn one motion dictionary $\mathbf{D_M}$ (a $n_1 \times d_1$ matrix, with d_1 the dictionary size and n_1 the motion vector size) and the codes $\mathbf{U_M}$ (a $d_1 \times K$ matrix) that minimize the Equation 1, so that $\mathbf{z}(t) \sim \mathbf{D_M u_M}(t)$. In the same manner, we define the equal optimization problem for a dictionary $\mathbf{D_G}$ (a $n_2 \times d_2$ matrix) and the codes $\mathbf{U_G}$ (a $d_2 \times K$ matrix) for

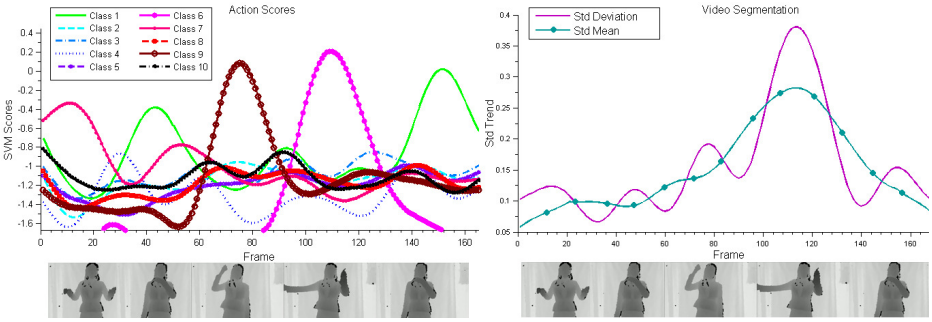

Figure 4: The figure illustrates on the left the SVMs scores (Equation 2) computed in real-time at each time step t over a sequence of 170 frames. On the right the standard deviation of the scores and its mean computed on a sliding window are depicted. The local minima of the standard deviation function are break points that define the end of an action and the beginning of another one. See Section 4.3.2 for details.

the set of GHOGs descriptors $\mathbf{H} = [\mathbf{h}(1), \ldots, \mathbf{h}(K)]$. Therefore, after the Sparse Coding stage, we can describe a frame as a code $\mathbf{u}(i)$, which is the concatenation of the motion and appearance codes: $\mathbf{u}(i) = [\mathbf{u_M}(i), \mathbf{u_G}(i)]$.

Notice that we rely on global features, thus we do not need any pooling operator, which is usually employed to summarize local features into a single one.

4.3 Learning and Recognition

The goal of this phase is to learn a model of a given action from data. Since we are implementing a one-shot action recognition system, the available training data amounts to one training sequence for each action of interest. In order to model the temporal extent of an action we extract sets of sub-sequences from a sequence, each one containing T adjacent frames. In particular, instead of using single frame descriptors (described in Section 4.2), we move to a concatenation of frames: a set of T frames is represented as a sequence $[\mathbf{u}(1), \ldots, \mathbf{u}(T)]$ of codes. This representation allows us to perform simultaneously detection and classification of actions.

The learning algorithm we adopt is the Support Vector Machine (SVM) (Vapnik, 1998). We employ linear SVMs, since they can be implemented with constant complexity during the test phase fulfilling real-time requirements (Fan et al., 2008). Additionally, recent advances in the object recognition field, such as Yang et al. (2009), showed that linear classifiers can effectively solve the classification problem if a preliminary sparse coding stage has previously been applied. Our experiments confirm these findings. Another advantage of linear SVMs is that they can be implemented with a linear complexity in training (Fan et al., 2008); given this property, we can provide a real-time one-shot learning procedure, extremely useful in real applications.

The remainder of the section describes in details the two phases of action learning and action recognition.

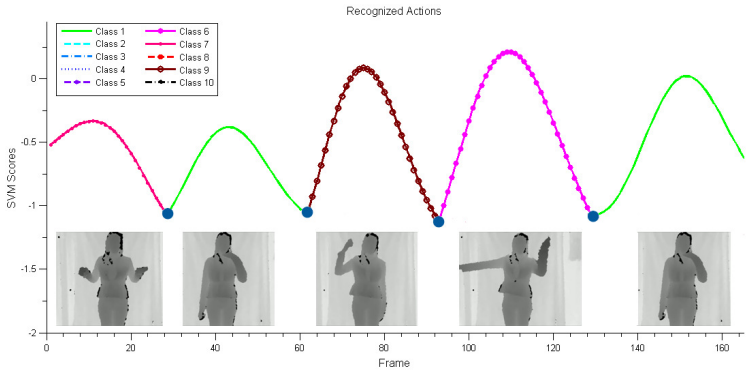

Figure 5: The figure illustrates only the scores of the recognized actions via the method described in Section 4.3.2. Blue dots are the break points computed by the video segmentation algorithm that indicate the end of an action and the beginning of a new one.

4.3.1 ACTION LEARNING

Given a video V_s of t_s frames, containing only one action A_s, we compute a set of descriptors $[\mathbf{u}(1),\ldots,\mathbf{u}(t_s)]$ as described in Section 4.2. Then, action learning is carried out on a set of data that are descriptions of a frame buffer $\mathbf{B}_T(t)$, where T is its length:

$$\mathbf{B}_T(t) = (\mathbf{u}(t-T),\ldots,\mathbf{u}(t-1),\mathbf{u}(t))^T.$$

We use a one-versus-all strategy to train a binary linear SVM for each class A_s, so that at the end of the training phase we obtain a set of N linear SVM classifiers $f_1(\bar{\mathbf{B}}),\ldots,f_N(\bar{\mathbf{B}})$, where N is the number of actions. In particular, in this one-shot learning pipeline, the set of buffers

$$\mathbf{B_s} = [\mathbf{B}_T(t_0),\ldots,\mathbf{B}_T(t_s)]$$

computed from the single video V_s of the class A_s are used as positive examples for the action A_s. All the buffers belonging to A_j with $j \neq s$ are the negative examples. Although we use only one example for each class, we benefit from the chosen representation: indeed, descriptors are computed per frame, therefore one single video of length t_s provides a number of examples equal to $t_s - T$ where T is the buffer size. Given the training data $\{\mathbf{B},\mathbf{y}\}$ where $\mathbf{B}$ is the set of positive and negative examples for the primitive A_s, $y_i = 1$ if the example is positive, $y_i = -1$ otherwise, the goal of SVM is to learn a linear function $(\mathbf{w}^T, b)$ such that a new test vector $\bar{\mathbf{B}}$ is predicted as:

$$y_{pred} = sign(f(\bar{\mathbf{B}})) = sign(\mathbf{w}^{\mathrm{T}}\bar{\mathbf{B}} + b).$$

4.3.2 ON-LINE RECOGNITION: VIDEO SEGMENTATION

Given a test video V, which may contain one or more known actions, the goal is to predict the sequence of the performed actions. The video is analyzed using a sliding window $\mathbf{B}_T(t)$ of size T. We compute the output score $f_i(\mathbf{B}_T(t))$ of the $i = 1,\ldots,N$ SVM machines for each test buffer $\mathbf{B}_T(t)$ and we filter these scores with a low-pass filter W that attenuates noise. Therefore the new score at

time t becomes:

$$H_i(\mathbf{B}_T(t)) = W \star f_i(\mathbf{B}_T(t)) \quad i = 1,\ldots,N, \tag{2}$$

where the $\star$ is the convolution operator. Figure 4 depicts an example of these scores computed in real-time. As long as the scores evolve we need to predict (on-line) when an action ends and another one begins; this is achieved computing the standard deviation $\sigma(H)$ for a fixed t over all the scores H_i^t (Figure 4, right chart). When an action ends we can expect all the SVM output scores to be similar, because no model should be predominant with respect to idle states; this brings to a local minimum in the function $\sigma(H)$. Therefore, each local minimum corresponds to the end of an action and the beginning of a new one. Let n be the number of local minima computed from the standard deviation function; there will be $n+1$ actions, and in particular actions with the highest score before and after each break point will be recognized. We can easily find these minima in real-time: we calculate the mean value of the standard deviation over time using a sliding window. When the standard deviation trend is below the mean, all the SVMs scores predict similar values, hence it is likely that an action has just ended. In Figure 5 the segmented and recognized actions are shown together with their scores.

5. Experiments

In this section we evaluate the performance of our system in three different settings:

- **ChaLearn Gesture Data Set**. The first experiment has been conducted on a publicly available data set, released by ChaLearn (, CGD2011). The main goal of the experiment is to compare our method with other techniques.

- **Kinect Data**. In the second experiment we discuss how to improve the recognition rate using all the functionalities of a real Kinect sensor. Gestures with high level of detail are easily caught by the system.

- **Human-Robot Interaction**. For the last experiment we considered a real HMI scenario: we implement the system on a real robot, the iCub humanoid robot (Metta et al., 2008), showing the applicability of our algorithm also in human-robot interaction settings.

For the computation of the accuracy between a sequence of estimated actions and the ground truth sequence we use the normalized Levenshtein Distance (Levenshtein, 1966), defined as:

$$TeLev = \frac{S+D+I}{M},$$

where each action is treated as a symbol in a sequence, S is the number of substitutions (misclassifications), D the number of deletions (false negatives), I the number of insertions (false positives) and M the length of the ground truth sequence. More specifically, this measure computes the minimum number of modifications that are required to transform a sequence of events in another one. It is widely used in speech recognition contexts, where each symbol represents an event. In action and gesture recognition, when sequences of gestures are to be evaluated, the Levenshtein Distance shows to be a particularly suitable metric, as it allows accounting not only for the single classifier accuracy, but also for the capability of the algorithm to accurately distinguish different gestures in a sequence (Minnen et al., 2006).

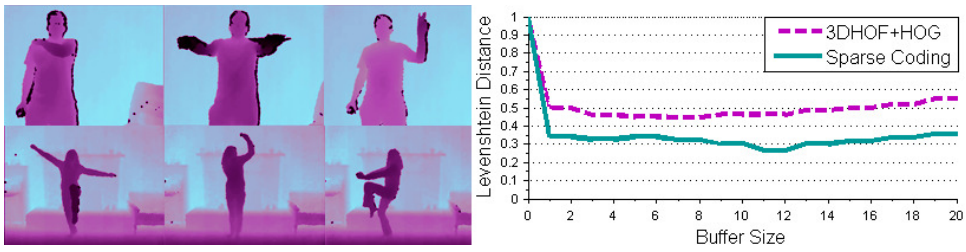

Figure 6: On the left examples of 2 different batches from the ChaLearn Data Set (, CGD2011). On the right the overall Levenshtein Distance computed in 20 batches with respect to the buffer size parameter is depicted for both 3DHOF+GHOG features and descriptors processed with sparse coding.

We empirically choose a quantization parameter for the 3DHOF, n_1 equal to 5, $n_2 = 64$ bins for the GHOG descriptor, and dictionary sizes d_1 and d_2 equal to 256 for both motion and appearance components. This led to a frame descriptor of size 189 for simple descriptors, which increases to 512 after the sparse coding processing. The whole system runs at 25fps on 2.4Ghz Core 2 Duo Processor.

5.1 ChaLearn Gesture Data Set

We firstly assess our method on the ChaLearn data set for the One-Shot Gesture Recognition Challenge (Guyon et al., 2012), see Figure 6. The data set is organized in batches, where each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures arbitrarily performed at different speeds. The gestures are drawn from a small vocabulary of 8 to 15 unique gestures called *lexicon*, which is defined within a batch. For each video both RGB and Depth streams are provided, but *only one example* is given for the training phase. In our experiments we do not use information on the body pose of the human. We consider the batches from *devel_01* to *devel_20*; each batch has 47 videos, where L (the lexicon size) videos are for training and the remaining are used as test data.

The main parameter of the system is the buffer size T, however in Figure 6 it is possible to notice that the parameter offers stable performances with a buffer range of $1-20$, so it does not represent a critical variable of our method. Furthermore, high performance for a wide buffer length range imply that our framework is able to handle different speeds implicitly. We compute the Levenshtein Distance as the average over all the batches, which is 25.11% for features processed with sparse coding, whereas simple 3DHOF+GHOG descriptors without sparse coding lead to a performance of 43.32%. Notably, each batch has its own lexicon and some of them are composed of only gestures performed by hand or fingers; in these cases, if the GHOG is computed on the entire ROI, the greatest contribution of the histogram comes from the body shape, whilst finger actions (see Figure 2, bottom row) represent a poor percentage of the final descriptor. If we consider batches where the lexicon is not composed of only hand/fingers gestures, the Levenshtein Distance reduces to 15%.

We compared our method with several approaches. First of all a Template Matching technique, where we used as descriptor the average of all depth frames for each action. The test video is split in

Method	TeLev	TeLen
Sparse Representation **(proposed)**	**25.11%**	**5.02%**
3DHOF + GHOG	43.32%	9.03%
Template Matching	62.56%	15.12%
DTW	49.41%	Manual
Manifold LSR (Lui, 2012)	28.73%	6.24%
MHI (Wu et al., 2012)	30.01%	NA
Extended-MHI (Wu et al., 2012)	26.00%	NA
BoVW (Wu et al., 2012)	72.32%	NA
2D FFT-MHI (Mahbub et al., 2011)	37.46%	NA
TBM+LDA (Malgireddy et al., 2012)	24.09%	NA

Table 1: Levenshtein Distance on the ChaLearn Gesture Data Set. For SVM classification we chose the appropriate buffer size for each batch according to the defined lexicon. TeLev is the Levenshtein Distance, TeLen is the average error (false positives + false negatives) made on the number of gestures (see text).

slices estimated using the average size of actions. In the recognition phase we classify each slice of the video comparing it with all the templates. The overall Levenshtein Distance becomes 62.56%. For the second comparison we employ Dynamic Time Warping (DTW) method (Sakoe and Chiba, 1978) with 3DHOF + GHOG features. We manually divided test videos in order to facilitate the recognition for DTW; nevertheless the global Levenshtein Distance is 49.41%. Finally we report the results presented in some recent works in the field, which exploit techniques based on manifolds (Lui, 2012), Motion History Image (MHI) (Wu et al., 2012), Bag of Visual Words (BoVW) (Wu et al., 2012), 2D FFT-MHI (Mahbub et al., 2011) and Temporal Bayesian Model (TBM) with Latent Dirichlet Allocation (LDA) (Malgireddy et al., 2012).

Table 1 shows that most of the compared approaches are outperformed by our method except for Malgireddy et al. (2012); however the method proposed by Malgireddy et al. (2012) has a training computational complexity of $O(n \times k^2)$ for each action class, where k is the number of HMM states and n the number of examples, while the testing computational complexity for a video frame is $O(k^2)$. Thanks to the sparse representation, we are able to use linear SVMs, which reduce the training complexity with respect to the number of training examples to $O(n \times d)$ for each SVM, where d is the descriptor size. In our case d is a constant value fixed a priori, and does not influence the scalability of the problem. Therefore we may approximate the asymptotic behavior of the SVM in training to $O(n)$. Similarly, in testing the complexity for each SVM is constant with respect to the number of training examples when considering a single frame, and it becomes $O(N)$ for the computation of all the N class scores. This allows us to provide real-time training and testing procedures with the considered lexicons.

Furthermore our on-line video segmentation algorithm shows excellent results with respect to the temporal segmentation used in the compared frameworks; in fact it is worth noting that the proposed algorithm leads to an action detection error rate $TeLen = \frac{FP+FN}{M}$ equal to 5.02%, where FP and FN are false positives and false negatives respectively, and M is the number of all test

gestures. Considering the final results of the ChaLearn Gesture Challenge (Round 1),[1] we placed 9th over 50 teams, but our method also fulfills real-time requirements for the entire pipeline, which was not a requirement of the challenge.

5.1.1 MOTION VS APPEARANCE

In this section we evaluate the contribution of the frame descriptors. In general we notice that the combination of both motion and appearance descriptors leads to the best results when the lexicon is composed of actions where both motion and appearance are equally important. To show this, we considered the 20 development batches from the ChaLearn Gesture Data Set. For this experiment, we used only coded descriptors, since we have already experienced that they obtain higher performance. Using only the motion component, the Levenshtein Distance is equal to 62.89%, whereas a descriptor based only on the appearance leads to an error of 34.15%. The error obtained using only the 3DHOF descriptors was expected, due to the nature of the lexicons chosen: indeed in most gestures the motion component has little significance. Considering instead batch *devel_01*, where motion is an important component in the gesture vocabulary, we have that 3DHOF descriptors lead to a Levenshtein Distance equal to 29.48%, the GHOG descriptors to 21.12% and the combination is equal to 9.11%. Results are consistent with previous findings, but in this specific case the gap between the motion and the appearance components is not critical.

5.1.2 LINEAR VS NON-LINEAR CLASSIFIERS

In this section we compare the performances of linear and non linear SVM for the action recognition task. The main advantage of a linear kernel is the computational time: non-linear SVMs have a worst case training computational complexity per class equal to $O(n^3 \times d)$ against the $O(n \times d)$ of linear SVMs, where n is the number of training examples, and d is the descriptor size. In testing, non linear SVMs show computational complexity of $O(n \times d)$ per frame, since the number of support vectors grows linearly with n. Moreover, non-linear classifiers usually require additional kernel parameter estimation, which especially in one-shot learning scenarios is not trivial. Contrarily, linear SVMs take $O(d)$ per frame. For this experiment we used coded features where both motion and appearance are employed. A non-linear SVM with RBF Kernel has been employed, where the kernel parameter and the SVM regularization term have been chosen empirically after 10 trials on a subset of the batches. The Levenshtein Distance among the 20 batches is 35.11%; this result confirms that linear classifiers are sufficient to obtain good results with low computational cost if an appropriate data representation, as the one offered by sparse coding, is adopted.

5.2 Kinect Data Set

In this section we assess the ability of our method to recognize more complex gestures captured by a Kinect for Xbox 360 sensor. In Section 5.1, we noted that the resolution of the proposed appearance descriptor is quite low and may not be ideal when actions differ by small details, especially on the hands, therefore a localization of the interesting parts to model would be effective. The simplest way to build in this specific information is to resort to a body part tracker; indeed, if a body tracker were available it would have been easy to extract descriptors from different limbs and then concatenate all the features to obtain the final frame representation. An excellent candidate to provide a reliable

1. The leaderboard website is: `https://www.kaggle.com`.

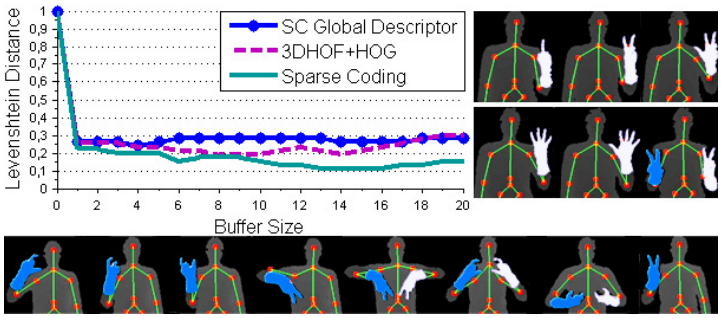

Figure 7: On the right and bottom the two vocabularies used in Section 5.2; these gestures are difficult to model without a proper body tracker, indeed the most contribution for the GHOG comes from the body shape rather than the hand. On the left the Levenshtein Distance.

body tracker is Microsoft Kinect SDK, which implements the method in Shotton et al. (2011). This tool retrieves the 20 principal body joints position and pose of the user's current posture. Given these positions, we assign each 3D point of the ROI to its nearest joint, so that it is possible to correctly isolate the two hands and the body from the rest of the scene (see Figure 7). Then, we slightly modify the approach, computing 3DHOF and GHOG descriptors on three different body parts (left/right hand and whole body shape); the final frame representation becomes the concatenation of all the part descriptors. As for the experiments we have acquired two different sets of data (see Figure 7): in the first one the lexicon is composed of numbers performed with fingers, in the second one we replicate the lexicons *devel_3* of the ChaLearn Gesture Data Set, the one where we obtained the poorest performances. In Figure 7 on the left the overall accuracy is shown; using sparse coding descriptors computed only on the body shape we obtain a Levenshtein Distance around 30%. By concatenating descriptors extracted from the hands the system achieves 10% for features enhanced with sparse coding and 20% for normal descriptors.

We compared our method with two previously mentioned techniques: a Template Matching algorithm and an implementation of the Dynamic Time Warping approach (Sakoe and Chiba, 1978). The resulted Levenshtein Distance is respectively 52.47% and 42.36%.

5.3 Human-Robot Interaction

The action recognition system has been implemented and tested on the iCub, a 53 degrees of freedom humanoid robot developed by the RobotCub Consortium (Metta et al., 2008). The robot is equipped with force sensors and gyroscopes, and it resembles a 3-years old child. It mounts two Dragonfly cameras, providing the basis for 3D vision, thus after an offline camera calibration procedure we can rely on a full stereo vision system; here the depth map is computed following Hirschmuller (2008). In this setting the action recognition system can be used for more general purposes such as Human-Robot-Interaction (HRI) or learning by imitation tasks. In particular our goal is to teach iCub how to perform simple manipulation tasks, such as move/grasp an object. In this sense, we are interested in recognizing actions related to the arm-hand movements of the robot. We define 8 actions, as shown in Figure 8, bottom row, according to the robot manipulation capabilities.

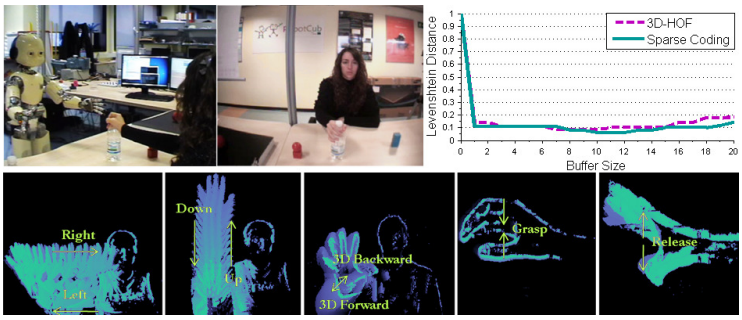

Figure 8: Accuracy for actions sequences (see bottom row). We evaluated the performance on more than 100 actions composed of sequences of 1 to 6 actions.

Each action is modeled using only the motion component (3DHOF), since we want the descriptor to be independent on the particular object shape used.

In Figure 8 we show the accuracy based on the Levenshtein Distance; this measure has been calculated on more than 100 actions composed of sequences of 1 to 6 actions. Notably the error is less than 10%; these good results were expected due to the high discriminative power of the 3DHOFs (Figure 3) on the chosen lexicon, which leads to a linearly separable set.

6. All Gestures You Can: a Real Application

As pointed out in the previous sections, our approach was designed for real applications where real-time requirements need to be fulfilled. We developed and implemented a "game" against a humanoid robot, showing the effectiveness of our system in a real HRI setting: "All Gestures You Can" (Gori et al., 2012), a game aiming at improving memory skills, visual association and concentration. Our game takes inspiration from the classic "Simon" game; nevertheless, since the original version has been often defined as "visually boring", we developed a revisited version, based on gesture recognition, which involves a "less boring" opponent: the iCub (Metta et al., 2008). Both the human and the robot have to take turns and perform the longest possible sequence of gestures by adding one gesture at each turn: one player starts performing a gesture, the opponent has to recognize the gesture, imitate it and add another gesture to the sequence. The game is carried on until one of the two players loses: the human player can lose because of limited memory skills, whereas the robot can lose because the gesture recognition system fails. As described in the previous sections, the system has been designed for one-shot learning; however, Kinect does not provide information about finger configuration, therefore a direct mapping between human fingers and the iCub's ones is not immediate. Thus we set a predefined pool of 8 gestures (see Figure 9, on the left). The typical game setting is shown in Figure 10: the player stays in front of the robot while performing gestures that are recognized with Kinect. Importantly, hand gestures cannot be learned exploiting the Skeleton Data of Kinect: the body tracker detects the position of the hand and it is not enough to discriminate more complicate actions,—for example, see gesture classes 1 and 5 or 2 and 6 in Figure 9.

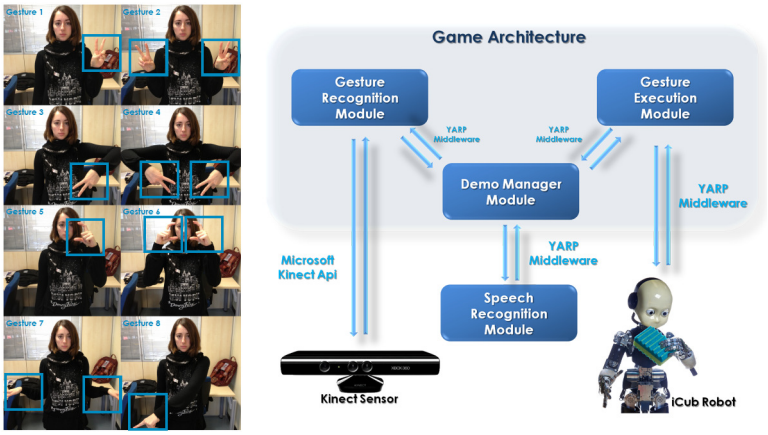

Figure 9: On the left the hand gestures. The vision system has been trained using 8 different actors performing each gesture class for 3 times. On the right the game architecture. There are three main modules that take care of recognizing the action sequence, defining the game rules and making the robot gestures.

The system is simple and modularized as it is organized in three components (see Figure 9) based on the iCub middleware, YARP (Metta et al., 2006), which manages the communication between sensors, processors, and modules. The efficiency of the proposed implementation is assured by its multithreading architecture, which also contributes to real-time performances. The software presented in this section is available in the iCub repository.[2]

The proposed game has been played by more than 30 different players during the ChaLearn Kinect Demonstration Competition at CVPR 2012.[3] Most of them were completely naive without prior knowledge about the gestures. They were asked to play using a lexicon that had been trained specifically for the competition (Figure 9). After 50 matches we had 75% of robot victories. This result indicates that the recognition system is robust also to different players performing variable gestures at various speeds. 15% of the matches have been won by humans and usually they finished during the first 3-4 turns of the game; this always occurred when players performed very different gestures with respect to the trained ones. A few players (10% of matches) succeeded in playing more than 8 turns, and they won due to recognition errors. "All Gestures You Can" ranked 2nd in the ChaLearn Kinect Demonstration Competition.

7. Discussion

This paper presented the design and implementation of a complete action recognition system to be used in real world applications such as HMI. We designed each step of the recognition pipeline to function in real-time while maximizing the overall accuracy. We showed how a sparse action repre-

2. Code available at `https://svn.code.sf.net/p/robotcub/code/trunk/iCub/contrib/src/demoGestureRecognition`.
3. The competition website is `http://gesture.chalearn.org/`
A YouTube video of our game is available at `http://youtu.be/U_JLoe_fT3I`.

Figure 10: The first two turns of a match. Left: the human player performs the first gesture of the sequence. Center: iCub recognized the gesture and imitates it. Right: iCub adds a new random gesture to the sequence.

sentation could be effectively used for one-shot learning of actions in combination with conventional machine learning algorithms (i.e., SVM), even if the latter would normally require a larger set of training data. The comprehensive evaluation of the proposed approach showed that we achieve good trade-off between accuracy and computation time. The main strengths of our learning and recognition pipeline can be summarized as follows:

1. **One-Shot Learning**: one example is sufficient to teach an new action to the system; this is mainly due to the effective per-frame representation.

2. **Sparse Frame Representation**: starting from a simple and computationally inexpensive description that combines global motion (3DHOF) and appearance (GHOG) information over a ROI, subsequently filtered through sparse coding, we obtained a sparse representation at each frame. We showed that these global descriptors are appropriate to model actions of the upper body of a person.

3. **On-line Video Segmentation**: we propose a new, effective, reliable and on-line video segmentation algorithm that achieved a 5% error rate on action detection on a set of 2000 actions grouped in sequences of 1 to 5 gestures. This segmentation procedure works concurrently with the recognition process, thus a sequence of actions is simultaneously segmented and recognized.

4. **Real-time Performances**: the proposed system can be used in real-time applications, as it does require neither a complex features processing nor a computationally expensive training and testing phases. From the computational point of view the proposed approach scales well even for large vocabularies of actions.

5. **Effectiveness in Real Scenarios**: our method achieves good performances in a Human-Robot Interaction setting, where the RGBD images are obtained through binocular vision and disparity estimation. For testing purposes, we proposed a memory game, called "All Gestures You Can", where a person can challenge the iCub robot on action recognition and sequencing. The system ranked 2nd at the Kinect Demonstration Competition.[4]

4. The competition website is `http://gesture.chalearn.org/`.

We stress here the simplicity of the learning and recognition pipeline: each stage is easy to implement and fast to compute. It is shown to be adequate to solve the problem of gesture recognition; we obtained high-quality results while fulfilling real-time requirements. The approach is competitive against many of the state-of-the-art methods for action recognition.

We are currently working on a more precise appearance description at frame level still under the severe constraint of real-time performance; this would enable the use of more complex actions even when the body tracker is not available.

Acknowledgments

This work was supported by the European FP7 ICT projects N. 270490 (EFAA) and N. 270273 (Xperience).

References

J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 2011.

A. Ali and J.K. Aggarwal. Segmentation and recognition of continuous human activity. *IEEE Workshop on Detection and Recognition of Events in Video*, 2001.

J. Alon, V. Athitsos, Y. Quan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

A. Bisio, N. Stucchi, M. Jacono, L. Fadiga, and T. Pozzo. Automatic versus voluntary motor imitation: Effect of visual context and stimulus velocity. *PLoS ONE*, 2010.

A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001.

M. Bregonzio, S. Gong, and T. Xiang. Recognising action as clouds of space-time interest points. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

M. J. Burden and D. B. Mitchell. Implicit memory development in school-aged children with attention deficit hyperactivity disorder (adhd): Conceptual priming deficit? In *Developmental Neurophysiology*, 2005.

J. Cech, J. Sanchez-Riera, and R. Horaud. Scene flow estimation by growing correspondence seeds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

ChaLearn Gesture Dataset (CGD2011). `http://gesture.chalearn.org/data`, 2011.

S.P. Chatzis, D. Kosmopoulos, and P. Doliotis. A conditional random field-based model for joint sequence segmentation and classification. In *Pattern Recognition*, 2013.

C. Comoldi, A. Barbieri, C. Gaiani, and S. Zocchi. Strategic memory deficits in attention deficit disorder with hyperactivity participants: The role of executive processes. In *Developmental Neurophysiology*, 1999.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

A. Destrero, C. De Mol, F. Odone, and Verri A. A sparsity-enforcing method for learning face features. *IEEE Transactions on Image Processing*, 18:188–201, 2009.

A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *International Conference on Computer Vision*, 2003.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 2006.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9, 2008.

S. R. Fanello, I. Gori, and F. Pirri. Arm-hand behaviours modelling: from attention to imitation. In *International Symposium on Visual Computing*, 2010.

G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, 2003.

J. Feng, B. Ni, Q. Tian, and S. Yan. Geometric lp-norm feature pooling for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

M. A. Giese and T. Poggio. Neural mechanisms for the recognition of biological movements. *Nature reviews. Neuroscience*, 2003.

L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 2007.

I. Gori, S. R. Fanello, F. Odone, and G. Metta. All gestures you can: a memory game against a humanoid robot. *IEEE-RAS International Conference on Humanoid Robots*, 2012.

R.D. Green and L. Guan. Continuous human activity recognition. *Control, Automation, Robotics and Vision Conference*, 2004.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *International Journal of Machine Learning Research*, 3:1157–1182, 2003.

I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hammer, and H. J. E. Balderas. Chalearn gesture challenge: Design and first results. In *Computer Vision and Pattern Recognition Workshops*, 2012.

H. O. Hirschfeld. A connection between correlation and contingency. In *Mathematical Proceedings of the Cambridge Philosophical Society*, 1935.

H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

B. K. P. Horn and B. G. Shunk. Determining optical flow. *Journal of Artificial Intelligence*, 1981.

F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *International Conference on Computer Vision*, 2007.

I. Laptev and T. Lindeberg. Space-time interest points. In *IEEE International Conference on Computer Vision*, 2003.

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Conference on Neural Information Processing Systems*, 2007.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics - Doklady*, 1966.

W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops*, 2010.

H.-Y.M. Liao, D-Y. Chen, and S.-W Shih. Continuous human action segmentation and recognition using a spatio-temporal probabilistic framework. *IEEE International Symposium on Multimedia*, 2006.

Y. M. Lui. A least squares regression framework on manifolds and its application to gesture recognition. In *Computer Vision and Pattern Recognition Workshops*, 2012.

F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

U. Mahbub, H. Imtiaz, T. Roy, S. Rahman, and A. R. Ahad. Action recognition from one example. *Pattern Recognition Letters*, 2011.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008a.

J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, pages 53–69, 2008b.

M. R. Malgireddy, I. Inwogu, and V. Govindaraju. A temporal Bayesian model for classifying, detecting and localizing activities in video sequences. *Computer Vision and Pattern Recognition Workshops*, 2012.

G. Metta, P. Fitzpatrick, and L. Natale. YARP: Yet Another Robot Platform. *International Journal of Advanced Robotic Systems*, 2006.

G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The icub humanoid robot: an open platform for research in embodied cognition. In *Workshop on Performance Metrics for Intelligent Systems*, 2008.

D. Minnen, T. Westeyn, and T. Starner. Performance metrics and evaluation issues for continuous activity recognition. In *Performance Metrics for Intelligent Systems Workshop*, 2006.

D. L. Mumme. Early social cognition: understanding others in the first months of life. *Journal of Infant and Child Development*, 2001.

P. Natarajan and R. Nevatia. Coupled hidden semi markov models for activity recognition. In *Workshop Motion and Video Computing*, 2007.

B. A. Olshausen and D. J. Fieldt. Sparse coding with an overcomplete basis set: a strategy employed by v1. *Vision Research*, 1997.

N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 1979.

N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *International Journal of Computer Vision*, 2006.

R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 2010.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1978.

H. J. Seo and P. Milanfar. A template matching approach of one-shot-learning gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

J. W. Shneider and P. Borlund. Matrix comparison, part 1: Motivation and important issues for measuring the resemblance between proximity measures or ordination results. In *Journal of the American Society for Information Science and Technology*, 2007.

J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, Inc., 1998.

M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE International Conference on Computer Vision*, 2007.

P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. *European Conference on Computer Vision*, 2012.

A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision*, 2010.

G. Willems, T. Tuytelaars, and L. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *European Conference on Computer Vision*, 2008.

D. Wu, F. Zhu, and L. Shao. One shot learning gesture recognition from rgbd images. In *Computer Vision and Pattern Recognition Workshops*, 2012.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.