Hand Gesture Recognition with Depth Data

Fabio Dominio
University of Padova
Via Gradenigo 6B
Padova, Italy
dominiof@dei.unipd.it

Mauro Donadeo
University of Padova
Via Gradenigo 6B
Padova, Italy
donadeom@dei.unipd.it

Giulio Marin
University of Padova
Via Gradenigo 6B
Padova, Italy
maringiu@dei.unipd.it

Pietro Zanuttigh
University of Padova
Via Gradenigo 6B
Padova, Italy
zanuttigh@dei.unipd.it

Guido Maria Cortelazzo
University of Padova
Via Gradenigo 6B
Padova, Italy
corte@dei.unipd.it

ABSTRACT

Depth data acquired by current low-cost real-time depth cameras provide a very informative description of the hand pose, that can be effectively exploited for gesture recognition purposes. This paper introduces a novel hand gesture recognition scheme based on depth data. The hand is firstly extracted from the acquired depth maps with the aid also of color information from the associated views. Then the hand is segmented into palm and finger regions. Next, two different set of feature descriptors are extracted, one based on the distances of the fingertips from the hand center and the other on the curvature of the hand contour. Finally, a multi-class SVM classifier is employed to recognize the performed gestures. The proposed scheme runs in real-time and is able to achieve a very high accuracy on depth data acquired with the Kinect.

Categories and Subject Descriptors

I.4.8 [**Computing Methodologies**]: Image processing and computer vision Scene Analysis Depth cues; I.2.10 [**Computing Methodologies**]: Artificial intelligence Vision and Scene Understanding 3D/stereo scene analysis

Keywords

Gesture recognition; depth; SVM; Kinect

ARTEMIS'13, October 21, 2013, Barcelona, Spain.

ACM 978-1-4503-2393-2/13/10 ...$15.00.
http://dx.doi.org/10.1145/2510650.2510651.

1. INTRODUCTION

Hand gesture recognition [15] is an intriguing problem that has many applications in different fields, such as human-computer interaction, robotics, computer gaming, automatic sign-language interpretation and so on. A certain number of hand gesture recognition approaches, based on the analysis of images and videos only, may be found in the literature [15, 7], but a bidimensional representation is not always sufficient to capture the complex movements and inter-occlusions characterizing hand gestures. The recent introduction of low cost consumer depth cameras, like Time-Of-Flight cameras and Microsoft's Kinect^{TM}[4], has made depth acquisition available to the mass market, thus opening the way to novel gesture recognition approaches based on depth information.

The most common approach is to recognize the gestures from depth data without computing the full pose, by applying machine learning techniques to some relevant features extracted from the depth information. In [9] silhouette and cell occupancy features are used to build a shape descriptor that is then feeded to a classifier based on action graphs. Suryanarayan et Al. [14] extract 3D volumetric shape descriptors from the hand depth and then exploit Support Vector Machines for the classification. Other volumetric features and a SVM classifier are also used by [16]. Finally, [13] and [12] compare the histograms of the distance of hand edge points from the hand center in order to recognize the gestures.

Another possibility is to estimate the complete 3D hand pose from depth data. Keskin et Al. [6] try to estimate the pose by segmenting the hand depth map into its different parts, adapting the machine learning approach used for full body tracking performed by the Kinect. Other approaches exploit multi-view setups [1], since approaches based on a single camera are affected by a large amount of occluded parts, making the pose estimation rather challenging.

This paper is part of the first family of approaches and presents a novel hand gesture recognition scheme combining two types of hand pose features, namely, distance features describing the positions of the fingertips and curvature features computed on the hand contour. The descriptors constructed on the basis of such features are combined together and fed into a SVM classifier in order to recognize the performed gestures.

The paper is organized in the following way: Section 2 introduces the general architecture of the proposed gesture recognition system. Section 3 explains how the hand region is extracted from the acquired depth data and divided into palm and finger areas. The computation of the proposed feature descriptors is presented in Section 4 and the classifying algorithm is described in Section 5. Section 6 contains the experimental results and finally Section 7 draws the conclusions.

2. PROPOSED GESTURE RECOGNITION SYSTEM

The proposed gesture recognition system (shown in Fig. 1) encompasses three main steps. In the first one the samples corresponding to the hand region are extracted from the depth map and further subdivided into palm and finger samples. Then, two sets of features are extracted: the first describes the distance of the fingertips from the center of the hand while the other depends on the curvature of the hand contour. Finally a multi-class Support Vector Machine classifier is used to recognize the different gestures.

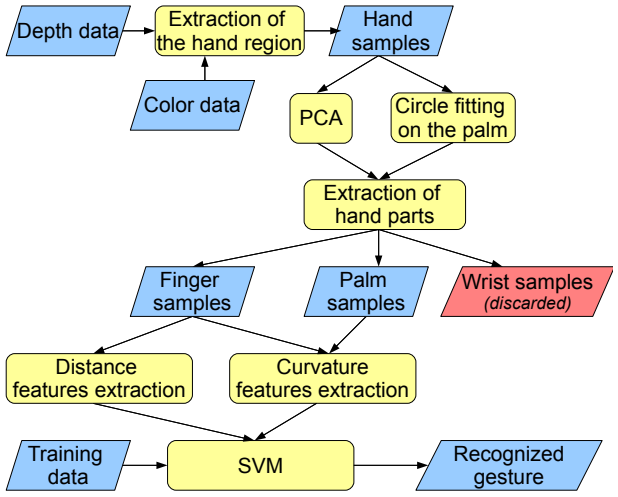


Figure 1: Architecture of the proposed gesture recognition system

3. EXTRACTION OF THE HAND AND OF ITS REGIONS

The first step, as pointed out in Section 1, is the extraction of the samples corresponding to the hand region from the depth map. The proposed method starts by detecting the point closest to the camera in the acquired depth map. Let us denote with $X_i, i = 1, .., N$ the 3D points acquired by the depth camera and with $D(X_i)$ the corresponding depth values, in particular $D(X_{min})$ will denote the depth of the closest sample. In order to avoid to select as the closest point an isolated artifact due to noise, the method verifies the presence of an adequate number of depth samples in a 5×5 region around the closest point. If the cardinality of the extracted set of point is below a given threshold we select the next closest point and repeat the check. Once the closest point is found, the set of all the points with depth within a threshold T_h from X_{min} and with a distance from X_{min} smaller than T_{h2} is computed:

$$\mathcal{H} = \{\mathbf{X_i} | D(X_i) < D(X_{min}) + T_h \wedge \|X_i - X_{min}\| < T_{h2}\} \quad (1)$$

The values of T_h and T_{h2} depend on the hand size (typical values are $T_h = 10cm$ and $T_{h2} = 30cm$). Two further checks are then performed on $\mathcal{H}$ in order to ensure that it corresponds to the hand: firstly its longest dimension must be bigger than $4cm$ in order to avoid to select small objects or isolated artifacts. Then the average color value in the CIELAB space must be compatible with the skin color (a simple thresholding is used for this step, while the reference skin color can be computed from a reference hand acquired from the user before starting). As in the previous case, if the point does not pass the checks a new starting point is selected and the procedure is repeated. Note how the color is used only for this step and not in all the subsequent phases of the algorithm. This step can be avoided if a depth-only approach is required. This approach allows to reliably divide the hand form the scene objects and from the other body parts (as shown in the example of Fig. 2c). However, since at this point one does not know the actual orientation of the hand and to which part of the hand the closest point belongs, it is necessary to use a relatively large threshold that will force the inclusion also of the wrist and the first part of the forearm into set $\mathcal{H}$. The wrist and forearm will be removed as described next.

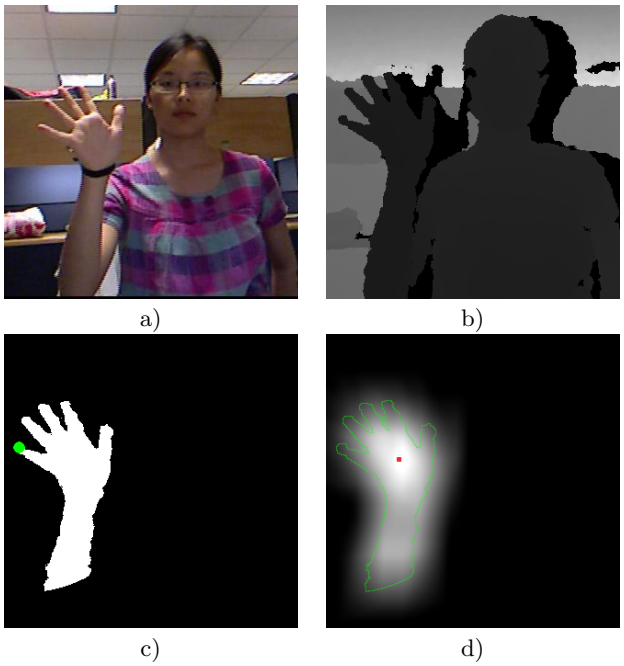


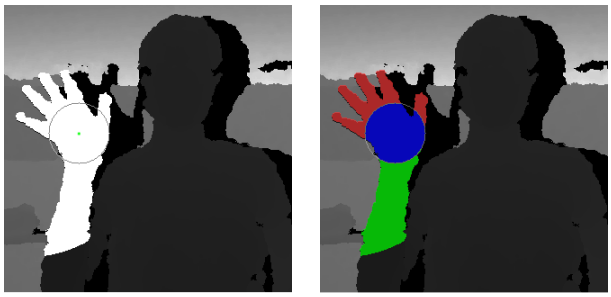
Figure 2: Extraction of the hand samples: a) Acquired color image; b) Acquired depth map; c) Extracted hand samples (the closest sample is depicted in green); d) Output of the Gaussian filter applied on the mask corresponding to $\mathcal{H}$ with the maximum (i.e., $\mathbf{C_g}$) in red; (*Best Viewed in colors*).

Let us define an indicator function on the points of $\mathcal{H}$:

$$h_m(X_i) = \begin{cases} 1 & X_i \in \mathcal{H} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The 2D mask H_{dm} in the depth image space corresponding to the samples for which $h_m(X_i) = 1$ is then filtered by a Gaussian filter with a very large standard deviation σ. The standard deviation σ depends on the distance of the hand from the Kinect, i.e., it is set to $\sigma = 150 \times 1[m]/D(X_{min})$. The maximum of the filtered image (corresponding to 3D coordinates $\mathbf{C_g}$) is then detected. As exemplified in Fig. 2d, since the Gaussian filter support is larger than the hand, and the palm is larger than the forearm and denser than the finger region, point C_g lies somewhere close to the center of the palm region. In case of multiple points with the same maximum value the closest to X_{min} is selected.

The next step of the proposed method is the detection of the largest circle, with center C and radius r, that can be fitted on the palm region (the fitting takes place in the depth map samples corresponding to the points in $\mathcal{H}$, i.e., on H_{dm}). The circle fitting uses the following procedure: a circle with initial center position $\mathbf{C} = \mathbf{C_g}$ is first expanded by increasing its radius r until 95% of the points inside it belong to $\mathcal{H}$ (we left a tolerance of 5% to account for errors due to noise or artifacts of the depth sensor). After the maximum radius value satisfying the threshold is found, the 2D coordinates $\mathbf{c}$ of point C are shifted towards the direction that maximizes the number of samples in $\mathcal{H}$ contained in the circle, with center C and radius r. Then, r is increased again and we continue to iterate the two phases until the largest possible circle has been fitted on the palm area, as shown in Fig. 3a. The final position of C, with 2D coordinates $\mathbf{c_f}$, matches the palm center. The corresponding 3D point, with coordinates $\mathbf{C_f}$, that we will call the *centroid* of the hand, will play an important role in the proposed algorithm together with the final radius value r_f. Note that if the palm is not parallel to the image plane the circle gets distorted by the projection from 3D space to the imaging plane. If the angle is relativelly small it is not a relevant issue but when the inclination is quite large better results can be obtained by fitting an ellipse in place of the circle, as proposed in [11].



a) b)

Figure 3: Extraction of the palm: a) Circle fitted on the hand with the point C_f in green; d) Palm (blue), finger (red) and wrist (green) regions subdivision. *(Best viewed in colors)*

At this point, the proposed method defines a new set $\mathcal{P}$, corresponding to the palm samples, and fits a plane π with normal $\mathbf{z_p}$ (denoted as *palm plane*) on them by a RANSAC approach. In our experiments, we set an inlier threshold of $10mm$ in order to take into account the average Kinect measurement noise at the operating distances. Principal Component Analysis (PCA) is then applied to $\mathcal{H}$ in order to extract the main axis $\mathbf{x}$ that roughly corresponds to the direction of the vector going from the wrist to the fingertips. Note how the direction computed in this way is not very precise, and depends on the position of the fingers in the performed gesture. It gives, however, a general indication of the hand orientation. The main axis direction will be refined as described in Section 4. Vector $\mathbf{x}$ is then projected on plane π and then a new reference system $(\mathbf{x_p}, \mathbf{y_p}, \mathbf{z_p})$ with origin in $\mathbf{C_f}$ is built as shown in Fig. 4. Its axes correspond to the projected first PCA eigenvector, to the plane π normal and to a third vector orthogonal to the first two.

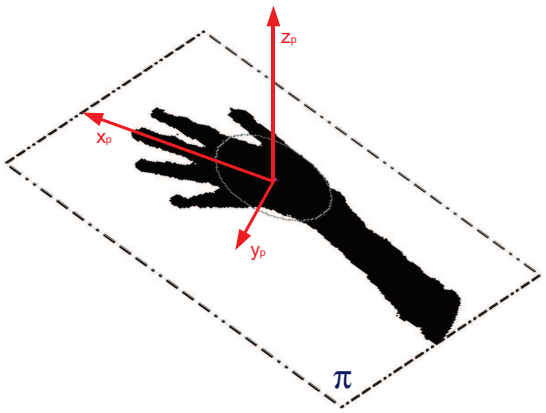


Figure 4: Reference system $(\mathbf{x_p}, \mathbf{y_p}, \mathbf{z_p})$ computed on the basis of the estimated plane and of the PCA output, used for the features extraction.

On the basis of the computed data, $\mathcal{H}$ is segmented in three sets as shown in Fig. 3:

- $\mathcal{P}$ containing points of the palm.
- $\mathcal{W}$ containing the points of $\mathcal{H}$ for which the x_p coordinate in the reference system $(\mathbf{x_p}, \mathbf{y_p}, \mathbf{z_p})$ satisfies the relationship $x_p \leq -r_f$. Such points belong to the wrist and the forearm, and will be discarded next. Note how the boundary plane has been placed at $x_p = -r_f$ and not at $x_p = 0$ since in some gestures the thumb can lie below the palm center.
- $\mathcal{F}$ containing the points of $\mathcal{H} \setminus (\mathcal{P} \cup \mathcal{W})$, corresponding to the fingers region.

Edge detection is then applied to the points of $(\mathcal{H} \setminus \mathcal{W})$ in order to build the set $\mathcal{E}$ containing the hand contour points. All the information needed by the proposed feature extraction scheme is now available.

4. EXTRACTION OF THE RELEVANT FEATURES

The proposed recognition scheme is based on two set of features, one describing the distance of the fingertips from the hand centroid and one describing the curvature of the hand contour.

4.1 Distance features

The computation of this feature set starts from the construction of a histogram representing the distance of the samples in $\mathcal{F}$ from the hand centroid C_f (note how the proposed scheme only considers finger samples, differently from other approaches using distance histograms like [13]). For each sample $\mathbf{X_i}$ in $\mathcal{F}$ we compute its distance $d_{\mathbf{X_i}} = \|\mathbf{X_i} - \mathbf{C_f}\|, \mathbf{X_i} \in \mathcal{F}$ from the centroid and the angle θ_{X_i} between the projections on the palm plane π of the PCA principal axis (i.e., $\mathbf{x_p}$) and of the vector $\mathbf{X_i} - \mathbf{C_f}$. We then quantize θ with a uniform quantization step Δ (in the current implementation we used $\Delta = 2^\circ$) into a discrete set of values θ_q. For each angular sector corresponding to a θ_q value we select the furthest point, thus producing an histogram $L(\theta)$:

$$L(\theta_q) = \max_{\theta_q - \frac{\Delta}{2} < \theta_{X_i} \leq \theta_q + \frac{\Delta}{2}} d_{\mathbf{X_i}} \tag{3}$$

For each gesture in the database a reference histogram $L_g^r(\theta)$ of the type of the one shown in Fig. 5 is built. A set of angular regions corresponding to the direction of the various fingers that are used in each gesture is defined as shown in Fig. 5. These regions correspond to each finger in each gesture and will be used for computing the distance features. For each acquired histogram we search for the maximum of the correlation $\rho(\cdot,\cdot)$ between the acquired histogram and the translated version of the reference histogram of each gesture[1]. We also consider the possibility of flipping the histogram to account for the fact that the hand could have either the palm or the dorsum facing the camera, evaluating:

$$\begin{aligned} \Delta_g &= \arg\max_{\Delta}(\rho(L(\theta), L_g^r(\theta + \Delta)) \\ \Delta_g^{rev} &= \arg\max_{\Delta}(\rho(L(-\theta), L_g^r(\theta + \Delta)) \end{aligned} \tag{4}$$

This gives us the translational shift aligning the acquired histogram L with the reference histograms L_g^r of each gesture g. We decide between Δ_g and Δ_g^{rev} on the basis of the one maximizing the correlation, i.e. we define the function:

$$L_a^g(\theta) = \begin{cases} L(\theta - \Delta_g) & \max_{\Delta} \rho(L(\theta), L_g^r(\theta + \Delta)) \geq \\ & \quad \max_{\Delta} \rho(L(-\theta), L_g^r(\theta + \Delta)) \\ L(-\theta - \Delta_g^{rev}) & \text{otherwise} \end{cases} \tag{5}$$

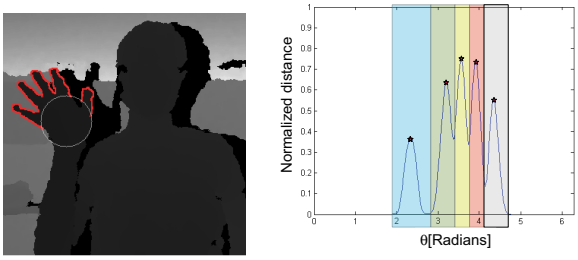
Note how there can be a different alignment Δ_g for each gesture. This approach basically compensates for the limited precision of the direction computed by the PCA, and allows to precisely align the reference and the computed histograms in order to define the regions corresponding to the various features of the gesture, thus solving one of the main issues of directly applying the approach of [13]. Fig. 6 shows some examples of the computed histograms for three different gestures. Note how the fingers raised in the various gestures (or the fact that there are no raised fingers for the fist) are clearly visible from the plots.

Let us denote with G the number of different gestures to be recognized. The set of features F^l can have a feature value for each finger $j \in \{1,..,5\}$ in each gesture $g \in \{1,..,G\}$ (not all the fingers are of interest in all the gestures). The feature value $f_{g,j}^l$ associated to finger j in gesture g is the maximum of the aligned histogram in the angular region $\theta_{g,j}^{min} < \theta < \theta_{g,j}^{max}$ corresponding to finger j in gesture g (see Fig. 5), i.e. :

$$f_{g,j}^l = \frac{\max_{\theta_{g,j}^{min} < \theta < \theta_{g,j}^{max}} L_a^g(\theta) - r_f}{L_{max}} \tag{6}$$

Note how the radius of the palm r_f is subtracted from all the features (otherwise feature values will jump from 0 to r_f as soon as edge points cross the circle), and that they are normalized by the length L_{max} of the middle finger in order to bring them within the $[0, 1]$ range and to account for the fact that the hands of different people have different sizes. Note how there can be up to $G \times 5$ features, but their actual number is smaller since not all the fingers are of interest in all the gestures (e.g. in the dataset used in the experimental results there are 10 different gestures and we used 24 features).

[1]In Eq. (4) and (5) L is considered as a periodic function with period 2π.



a) b)

Figure 5: Histogram of the edge distances with the features regions: a) finger edges $\mathcal{F}_e$ computed from $\mathcal{F}$; b) corresponding histogram $L(\theta)$ with the regions corresponding to the different features $f_{g,j}^l$ (feature points are highlighted with red stars).

4.2 Curvature features

The second employed descriptor is based on the curvature of the edges of the hand shape. Since data coming from real-time depth cameras usually are rather noisy and contain edge artifacts, we decided to avoid differential operators for curvature description. We instead exploited integral invariants [10, 8]. Our feature extractor algorithm takes as input the hand edge points $\mathcal{E}$ and the indicator function h_m defined in Eq. (2). Consider a set of S circular masks $M_s(\mathbf{X_i}), s = 1,..,S$ with radius r_s varying from $0.5cm$ to $5cm$ centred on each edge sample $X_i \in \mathcal{E}$. Let $V(\mathbf{X_i}, s)$ denote the number of samples of $\mathcal{H}$ inside $M_s(\mathbf{X_i})$, i.e.:

$$V(\mathbf{X_i}, s) = \frac{\sum_{X \in M_s(\mathbf{X_i})} h_m(\mathbf{X})}{|M_s(\mathbf{X_i})|} \tag{7}$$

Note how the radius value s actually corresponds to the scale level at which the feature extraction is performed. Differently from [8] and other approaches, the radius r_S is defined in metrical units and then converted into the corresponding pixel size on the basis of the distance between the camera and the hand, thus making the descriptor invariant with respect to the distance of the hand from the camera and consequently from the hand size in pixels in the ac-

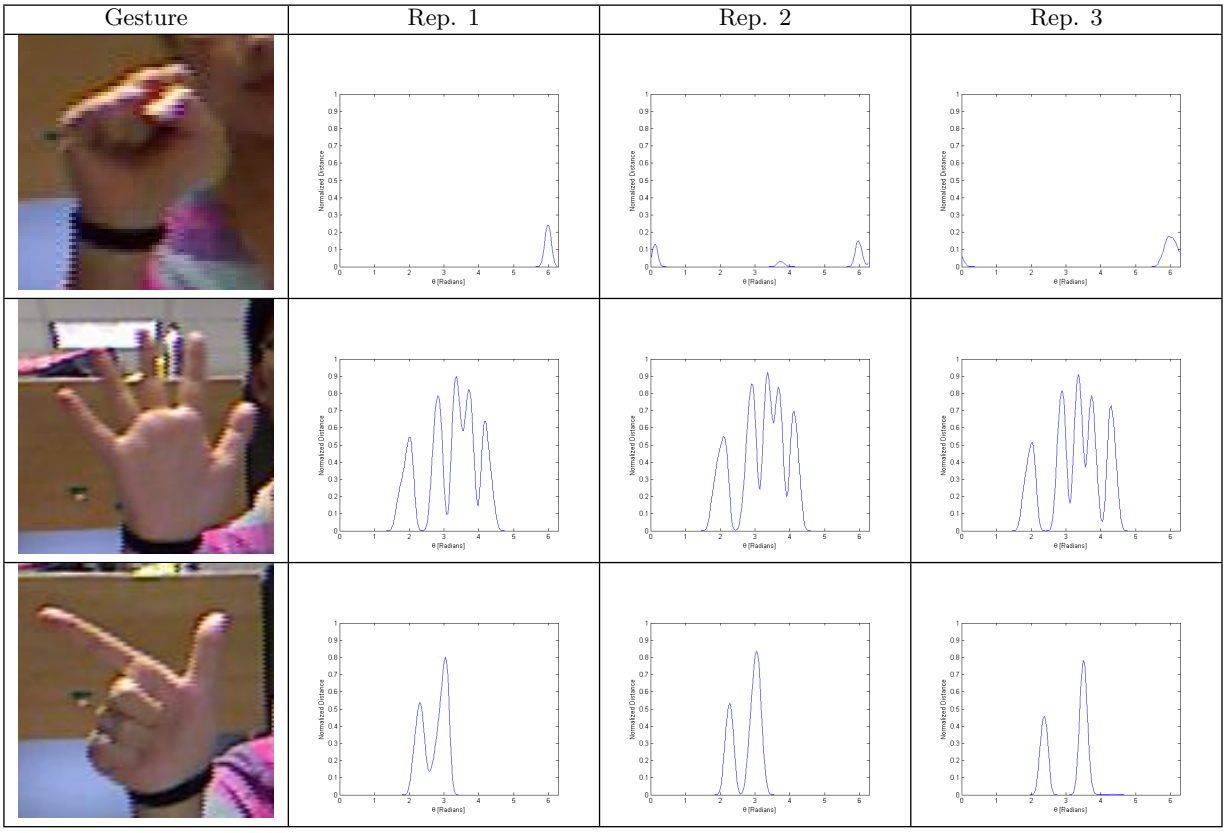


Figure 6: Examples of distance histogram for some sample frames from different gestures.

quired depth map. The values of $V(\mathbf{X_i}, s)$ range from 0 (extremely convex shape) to 1 (extremely concave shape) with $V(\mathbf{X_i}, s) = 0.5$ corresponding to a straight edge. We quantized the $[0, 1]$ interval into B bins of equal size $b_1, .., b_B$. Let $\mathcal{V}_{b,s}$ be the set of the finger edge points $\mathbf{X_i} \in \mathcal{E}$ with the corresponding value of $V(\mathbf{X_i}, s)$ falling in each bin b, namely:

$$\mathcal{V}_{b,s} = \{\mathbf{X_i} | \frac{(b-1)}{B} < V(\mathbf{X_i}, s) \leq \frac{b}{B}\} \tag{8}$$

We compute as curvature features the normalized cardinality of $\mathcal{V}_{b,s}$, for each bin b and radius value s, respect to the hand contour length, i.e.

$$f^c_{b,s} = \frac{|\mathcal{V}_{b,s}|}{|\mathcal{E}|} \tag{9}$$

In this way a second feature vector $\mathbf{F^c}$ containing $B \times S$ features is built, i.e., each feature value $f^c_{b,s}$ is the cardinality of set $V_{b,s}$ normalized by the cardinality of $\mathcal{E}$. As expected, the amount of curvature depends on the number of fingers not folded on the palm region and on their arrangement, thus giving an accurate description of the hand gesture.

5. GESTURE CLASSIFICATION WITH SUPPORT VECTOR MACHINES

In order to recognize the gestures from the feature vectors built in Section 4, we employed a multi-class Support Vector Machine classifier. Each acquired gesture is described by a feature vector $\mathbf{F} = [\mathbf{F^l}, \mathbf{F^c}]$ obtained by concatenating the distance feature vector $\mathbf{F^l}$ and the curvature one $\mathbf{F^c}$. The combined vector of features $\mathbf{F}$, describing the performed gesture, has the structure shown in Fig. 8; note how $\mathbf{F^l}$ actually contains all the distance features corresponding to the various possible *hypotheses* for the current gesture, while $\mathbf{F^c}$ basically contains the histograms of the curvature distribution for all the scale levels. The gesture recognition problem consists in classifying the vectors F with the distance and curvature features concatenated into G classes corresponding to the various gestures of the database. The employed algorithm is based on the *one-against-one* approach, i.e., a set of $G(G-1)/2$ binary SVM classifiers is used to test each class against each of the others and the output of each of them is chosen as a *vote* for a certain gesture. The gesture with the maximum number of votes is chosen as the recognized gesture. In particular, we used the implementation of SVMs contained in the LIBSVM package [3]. We chose non-linear Gaussian Radial Basis Function (RBF) kernel and tuned its parameters by grid search approach and cross-validation on the training set.

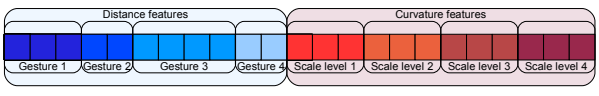


Figure 8: Feature vector

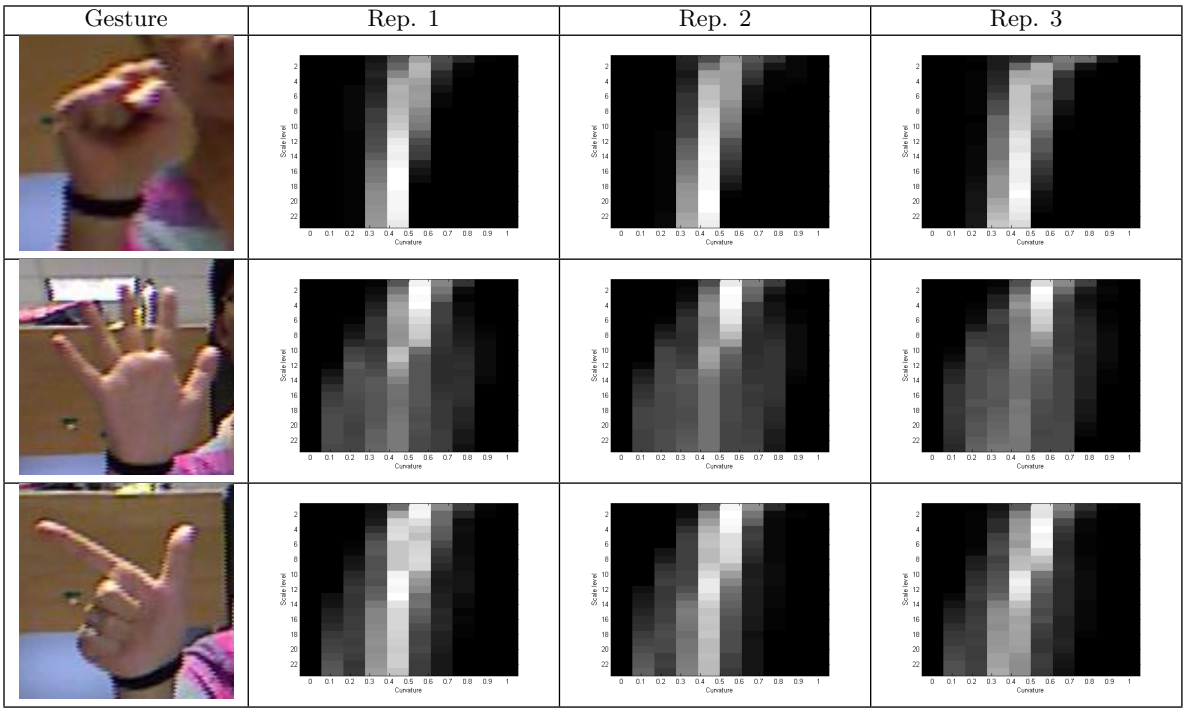


Figure 7: Examples of curvature descriptors for some sample frames from different gestures.

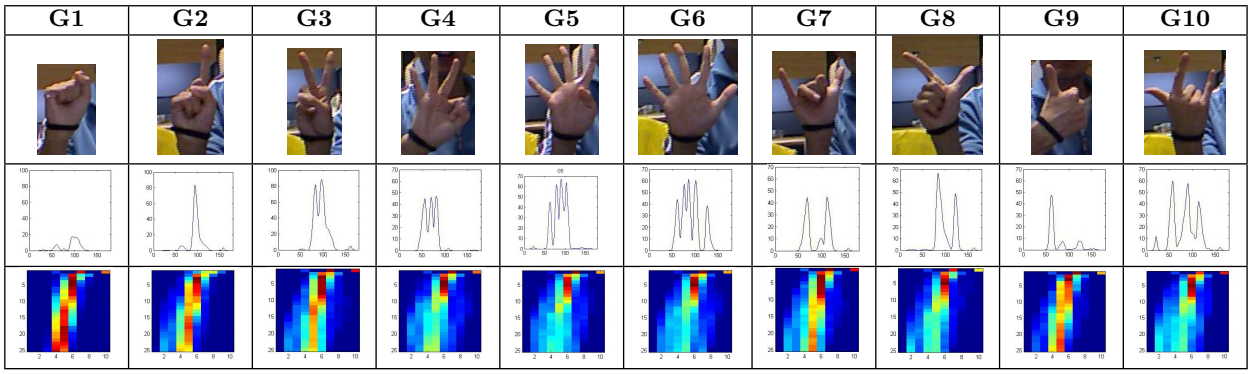


Table 1: Different gestures contained in the database used for the experimental results and corresponding feature descriptors. The first row contains the color images depicting the different considered gestures; the second shows the distance histograms computed from the depth maps corresponding to the images in the first row; the last row instead shows the curvature descriptors (for each plot the horizontal axis correspond to the curvature value and the vertical one to the scale level and the color is proportional to the amount of samples falling in the corresponding curvature bin).

6. EXPERIMENTAL RESULTS

In order to evaluate the performances of the proposed approach, we used the database provided by Ren et Al. [13]. This database has 10 different gestures (a sample image of each gesture is shown in Table 1) performed by 10 different people and repeated each 10 times, for a total of 1000 different depth maps.

We considered two different situations. In the first we randomly split the database into 2 parts, of which one made by 800 depth maps was used to train the SVM classifier and the other made by the remaining 200 models was used as test set. For each gesture, one of the repetitions in the training set was used for the computation of the reference histogram used in Eq. (4). The complete training set was then used to train 3 different SVM classifiers. The first uses the distance features only, the second the curvature features only and the third one jointly uses both features. The first 3 lines of Table 2 and the confusion matrices in Tables 3, 4 and 5 show the accuracy of the three classifiers on the considered database. Distance features only provides an accuracy of about 96%, almost the same achieved by the curvature-based classifier (97.5%). Note how the distance only classifier is able to recognize some of the gestures that curvature only one can not handle, and viceversa. The performance of the combined classifiers are, instead, the best ones, reaching an accuracy of 99.5%. This is because the 2 classifiers have complementary characteristics since the two descriptors are based on totally different clues.

Method	Mean Accuracy	Conf. Matrix
Distance features (tr. with users)	96 %	(3)
Curvature features (tr. with users)	97.5 %	(4)
Dist.+curv. (tr. with users)	**99.5** %	(5)
Distance features (generic tr.)	92.5 %	(6)
Curvature features (generic tr.)	92 %	(7)
Dist.+curv. (generic tr.)	**98.5** %	(8)
Shape context [2]	83.2%	
Near-convex Dec.+FEMD [13]	90.6%	
Thresholding Dec..+FEMD [13]	93.9%	

Table 2: Performance comparison on the database from [13]. The last column contains the number of the table with the corresponding confusion matrix.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	0.95	0	0	0	0	0	0.05	0	0
G3	0	0	0.95	0	0	0	0	0	0	0.05
G4	0	0	0	0.95	0	0.05	0	0	0	0
G5	0	0	0.05	0	0.95	0	0	0	0	0
G6	0	0	0	0	0.05	0.95	0	0	0	0
G7	0	0	0	0	0	0	0.9	0	0.1	0
G8	0	0	0	0	0	0	0	1	0	0
G9	0	0	0	0.05	0	0	0	0	0.95	0
G10	0	0	0	0	0	0	0	0	0	1

Table 3: Confusion matrix for gesture recognition with distance features (training with users). Each cell contains the fraction of samples of the input gesture (each input correspond to a row of the matix) recognized as the gesture corresponding to the column of the matrix.

This subdivision of the database corresponds to having gestures from all the 10 subjects in both the train and test sets. Sometimes it is necessary to have a system able to recognize the gestures performed by new people without any training on that particular user, but just using a "generic" training performed on different people. We modeled this situation by dividing the database into two parts in a different way: 8 people have been used for the training while the remaining 2 constitute the test set. This is a more difficult situation since there is no training on the people that will then use the system, but just a generic training that should work with everyone. The performance of each of the two descriptors alone in this case (rows 4 and 5 of Table 2) is quite lower than in the previous case, but by combining them the proposed approach is able to achieve performances very close to the ones of the previous configuration, as shown in row 6 of Table 2 and by the confusion matrix in Table 8. By comparing the confusion matrices in Tables 6, 7 and 8 it is possible to note how, even if when employing each of two descriptors alone there is a non-neglegible number of errors, the errors of the two classifiers are quite complementary and by fusing them it is possible to largely improve the results. For example notice that gesture 4 is quite critical for distance features but is almost correctly handled by curvature data, while on the opposite side distance descriptors achieve a 100% accuracy on gesture 8 that instead is the most critical for curvature data with an error rate of 30%.

The last three rows of Table 2 also compare the results with the ones obtained by [13], and from the application of the shape comparison method of [2] over the shape extracted from the depth maps[2]. It is possible to note how by combining both features the proposed scheme is able to outperform all the compared approaches. Furthermore, note how [13] exploits a black bracelet that all people in the database had to wear in order to locate and align the hand shapes, while our approach does not exploit this aid and does not require to wear any glove, bracelet or other sort of marker, thus being more suited to be used in practical applications. Finally, note how our approach does not involve complex computa-

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	1	0	0	0	0	0	0	0	0
G3	0	0	1	0	0	0	0	0	0	0
G4	0	0	0	1	0	0	0	0	0	0
G5	0	0	0	0.1	0.9	0	0	0	0	0
G6	0	0	0	0	0	1	0	0	0	0
G7	0	0	0	0	0	0	0.95	0	0.05	0
G8	0	0	0	0	0	0	0.1	0.9	0	0
G9	0	0	0	0	0	0	0	0	1	0
G10	0	0	0	0	0	0	0	0	0	1

Table 4: Confusion matrix for gesture recognition with curvature features (training with users).

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	1	0	0	0	0	0	0	0	0
G3	0	0	1	0	0	0	0	0	0	0
G4	0	0	0	1	0	0	0	0	0	0
G5	0	0	0	0	1	0	0	0	0	0
G6	0	0	0	0	0	1	0	0	0	0
G7	0	0	0	0	0	0	0.95	0.05	0	0
G8	0	0	0	0	0	0	0	1	0	0
G9	0	0	0	0	0	0	0	0	1	0
G10	0	0	0	0	0	0	0	0	0	1

Table 5: Confusion matrix for gesture recognition with both features (training with users).

[2]Results for [2] on this dataset have been extracted from [12]

tions and it runs in real-time. The current implementation has not been fully optimized, however we employed efficient implementations for some of the most computationally intensive steps, e.g., the PCA for orientation detection, the SVD used for the plane fitting and the Gaussian filtering for initial point detection.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	1	0	0	0	0	0	0	0	0
G3	0	0.05	0.95	0	0	0	0	0	0	0
G4	0	0	0	0.8	0.05	0.15	0	0	0	0
G5	0	0	0	0	1	0	0	0	0	0
G6	0	0	0	0	0.05	0.95	0	0	0	0
G7	0	0	0.05	0	0	0	0.85	0	0.1	0
G8	0	0	0	0	0	0	0	1	0	0
G9	0	0	0.05	0	0	0	0	0.1	0.85	0
G10	0	0	0	0.05	0.1	0	0	0	0	0.85

Table 6: Confusion matrix for gesture recognition with distance features (generic training).

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	1	0	0	0	0	0	0	0	0
G3	0	0	0.8	0	0	0	0.15	0.05	0	0
G4	0	0	0	0.95	0.05	0	0	0	0	0
G5	0	0	0	0.05	0.95	0	0	0	0	0
G6	0	0	0	0	0	1	0	0	0	0
G7	0	0	0	0	0	0	0.95	0	0.05	0
G8	0	0	0.25	0	0	0	0.05	0.7	0	0
G9	0	0.1	0	0	0	0	0	0	0.9	0
G10	0	0	0.05	0	0	0	0	0	0	0.95

Table 7: Confusion matrix for gesture recognition with curvature features (generic training).

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1	0	0	0	0	0	0	0	0	0
G2	0	1	0	0	0	0	0	0	0	0
G3	0	0	1	0	0	0	0	0	0	0
G4	0	0	0	0.9	0.1	0	0	0	0	0
G5	0	0	0	0	1	0	0	0	0	0
G6	0	0	0	0	0.05	0.95	0	0	0	0
G7	0	0	0	0	0	0	1	0	0	0
G8	0	0	5	0	0	0	0	1	0	0
G9	0	0	0	0	0	0	0	0	1	0
G10	0	0	0	0	0	0	0	0	0	1

Table 8: Confusion matrix for gesture recognition with both features (generic training).

7. CONCLUSIONS

This paper presents an effective way of exploiting depth information for hand gesture recognition purposes. It is worth noting how the palm and finger regions can be reliably extracted from depth data, a task usually rather challenging with color data only. The paper then introduces two different kinds of features representing distances of the fingers from the hand centroid and the curvature of the hand shape. Both types of features alone allow for reliable hand gesture recognition, but their combined usage can significantly enhance the recognition performances. Further research will be devoted to the introduction of new features into the proposed approach, in order to better describe the fingers when they are closed on the palm, and to the exploitation of color data in order to extract additional features. More complex machine learning strategies will also be considered, including deep learning techniques [5]. Then, since many gestures are characterized by a dynamic evolution over time, we are planning to extend the approach from the analysis of single frames to the analysis of video sequences considering also time-dependent features. Finallly we are planning to apply the proposed approach to the automatic recognition of the sign language and for this task we are creating a large database of sign language gestures.

8. REFERENCES

[1] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In *Proc. of ECCV*, October 2012.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on PAMI*, 24(4):509 –522, apr 2002.

[3] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Trans. on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[4] C. Dal Mutto, P. Zanuttigh, and G. M. Cortelazzo. *Time-of-Flight Cameras and Microsoft Kinect*. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012.

[5] N. Doulamis and A. Doulamis. Fast and adaptive deep fusion learning for detecting visual objects. In *Proceedings of ECCV 2012*, 2012.

[6] C. Keskin, F. Kirac, Y. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. In *ICCV Workshops*, pages 1228 –1234, nov. 2011.

[7] D. Kosmopoulos, A. Doulamis, and N. Doulamis. Gesture-based video summarization. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–1220–3, 2005.

[8] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. Lopez, and J. V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In *Proc. of ECCV*, October 2012.

[9] A. Kurakin, Z. Zhang, and Z. Liu. A real-time system for dynamic hand gesture recognition with a depth sensor. In *Proc. of EUSIPCO*, 2012.

[10] S. Manay, D. Cremers, B.-W. Hong, A. Yezzi, and S. Soatto. Integral invariants for shape matching. *IEEE Trans. on PAMI*, 28(10):1602 –1618, 2006.

[11] G. Marin, M. Fraccaro, M. Donadeo, F. Dominio, and P. Zanuttigh. Palm area detection for reliable hand gesture recognition. In *Proceedings of MMSP 2013*, 2013.

[12] Z. Ren, J. Meng, and J. Yuan. Depth camera based hand gesture recognition and its applications in human-computer-interaction. In *Proc. of ICICS*, pages 1 –5, 2011.

[13] Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proc. of ACM Conference on Multimedia*, pages 1093–1096. ACM, 2011.

[14] P. Suryanarayan, A. Subramanian, and D. Mandalapu. Dynamic hand pose recognition using depth data. In *Proc. of ICPR*, pages 3105 –3108, aug. 2010.

[15] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan. Vision-based hand-gesture applications. *Commun. ACM*, 54(2):60–71, Feb. 2011.

[16] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3d action recognition with random occupancy patterns. In *Proc. of ECCV*, 2012.