STAG: Smart Tools & Apps for Graphics (2015)
Silvia Biasotti and Marco Tarini and Andrea Giachetti (Editors)

Exploiting Silhouette Descriptors and Synthetic Data for Hand Gesture Recognition

A. Memo, L. Minto and P. Zanuttigh

Department of Information Engineering, University of Padova, Italy

Abstract

This paper proposes a novel real-time hand gesture recognition scheme explicitly targeted to depth data. The hand silhouette is firstly extracted from the acquired data and then two ad-hoc feature sets are computed from this representation. The first is based on the local curvature of the hand contour, while the second represents the thickness of the hand region close to each contour point using a distance transform. The two feature sets are rearranged in a three dimensional data structure representing the values of the two features at each contour location and then this representation is fed into a multi-class Support Vector Machine. The classifier is trained on a synthetic dataset generated with an ad-hoc rendering system developed for the purposes of this work. This approach allows a fast construction of the training set without the need of manually acquiring large training datasets. Experimental results on real data show how the approach is able to achieve a 90% *accuracy on a typical hand gesture recognition dataset with very limited computational resources.*

1. Introduction

Hand gesture recognition [WKSE11] is attracting a growing interest due to its applications in many different fields like human-computer interaction, robotics, computer gaming and automatic sign-language interpretation. The solution of this problem using image and video data [WKSE11] has always been very challenging but the introduction of low cost consumer depth cameras, like Time-Of-Flight and structured light cameras [DMZC12], has opened the way to novel approaches based on three-dimensional information. Depth data contains a very informative description of the hand pose that is very useful for gesture recognition applications.

This paper presents a novel real-time approach for gesture recognition based on depth data that can be used both for setups where the camera is facing the user or for head-mounted sensors. The proposed approach works on a single frame, thus making it suited for head mounted applications where the camera can move with respect to the arm. The hand is firstly extracted from the depth map and its silhouette is computed. Then two different descriptors are computed from the silhouette information. The first represents the local curvature along the hand contour, while the second is based on a distance transform and represent the thickness of the hand region close to each contour point. These descriptors are used to build a multi-dimensional feature set representing the local curvature and distance transform values at the various contour points. Finally a multi-class Support Vector Machine (SVM) classifier is used to recognize the performed gestures. The classifier is trained on a synthetic dataset produced with an ad-hoc rendering system, thus allowing the training on a large dataset without all the cumbersome work required for the construction of very large real datasets.

This paper introduces several novel contributions. Firstly two new sets of ad-hoc features for hand gesture recognition have been proposed. This work also introduces an efficient solution for the construction of large synthetic training datasets and demonstrates that training on synthetic data can be sufficient in order to effectively classify real gesture samples. This allows to avoid the cumbersome construction of large real data training datasets. As another major contribution, it introduces a new kind of three-dimensional structure for the combination of multiple features, which proved to generalize well to the real case yet requiring little computational power.

The paper is organized in the following way: Section 2 presents the related works. Section 3 introduces the general architecture of the proposed system. Section 4 presents the proposed feature extraction schemes. Then the classification scheme exploiting synthetic training is described in Section

5. Experimental results are in Section 6 and finally Section 7 draws the conclusions.

2. Related works

Several different hand gesture recognition approaches exploiting depth data have been introduced recently. There are various schemes but the most common basic pipeline consists in firstly extracting a set of relevant features from the depth data and then applying a suitable machine-learning technique in order to recognize the performed gesture. The approach of [KZL12] exploits silhouette and cell occupancy features to build a shape descriptor that is then fed to a classifier based on action graphs. Volumetric shape descriptors and a classifier based on Support Vector Machines are used both by [SSM10] and [WLC*12]. Another commonly used solution is to compare the histograms of the distance of hand edge points from the hand center in order to recognize the gestures, as in [RYZ11], [RMY11] and [DDM*13]. Four different types of features are extracted and fed into a SVM classifier in the approach of [DDZ14]. The approach has also been extended with different classification and feature selection schemes [NLD*13] and by exploiting also the data from the LeapMotion sensor [MDZ14]. The approach of [QZYJ14] exploits a convex shape decomposition method in order to recognize the palm, fingertips and hand skeleton and then use this data for gesture recognition. In [KOWW15] the hand point cloud is described using Viewpoint Feature Histograms (VFH) and Hidden Markov Models are used for the classification of static gestures while dynamic data is handled with Dynamic Time Warping (DTW). A novel descriptor representing the local distribution of 3D samples acquired by the depth sensor is proposed in [ZT15] and exploited for hand gesture recognition. Another recent work [DLK14] exploits a Random Forest Classifier trained on synthetic data to recognize the various hand parts and then uses this information for hand gesture recognition.

3. Overview of the proposed approach

The proposed approach encompass two main steps (the basic pipeline is shown in Figure 1). In the first step the hand is segmented from the rest of the scene and two sets of relevant features are extracted from the silhouette of the hand on the depth map. The first set of features captures the local curvature of the hand contour, while the second exploits the distance transform representation to get the thickness of the hand region corresponding to each contour point. In the second step the extracted features are used to build a three-dimensional vector representation that is fed to a classifier based on Support Vector Machines. The classifier is trained on a synthetic dataset constructed with the method of Section 5.1 and used to recognize gestures from a real dataset acquired with the Creative Senz3D consumer depth camera.

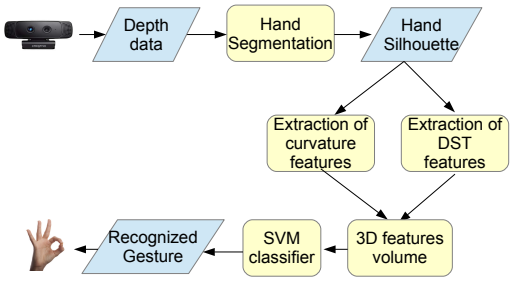


Figure 1: *Pipeline of the proposed approach.*

4. Feature Extraction

Both the proposed feature sets are based on the silhouette of the hand in the depth map. For this reason the hand is first extracted from the rest of the scene and then a binary mask representing the hand samples on the depth map is computed as detailed in Subsection 4.1. The two sets of features are finally extracted from the mask using the approaches described in Subsections 4.2 and 4.3.

4.1. Hand segmentation

The first step consists in the extraction of the samples corresponding to the hand region from the depth map. The process assumes that the hand is close to the camera and starts with a thresholding of the depth values that removes all the samples with a distance bigger than a pre-defined threshold that depends on the selected application (e.g., for the experimental results we used $0.5m$). In this way an initial hand mask is obtained (see Figure 2a): the set of all the points in this initial mask will be denoted with $\mathcal{H}'$. This assumption is verified in many typical hand gesture recognition settings, however more refined approaches can be used for more challenging situations, e.g., by using together color and depth information [DDZ14]. Typically after this operation the hand remains in the scene together with the first part of the arm and some other isolated objects or spots due to the noise of the sensor. The longest contour, that typically corresponds to the hand, is extracted from the binary mask and a second mask $\mathcal{H}''$ containing the area inside this contour is also generated. Notice that this mask contains only the hand without other objects. However, any internal contour or hole inside in the hand region will be filled (e.g., the area between the thumb and index of the "Ok" gesture in Figure 2). For this reason the masks $\mathcal{H}'$ and $\mathcal{H}''$ are intersected in order to get the mask $\mathcal{H}'''$:

$$\mathcal{H}''' = \mathcal{H}' \cap \mathcal{H}'' \tag{1}$$

which contains only the hand points together with the wrist and the first part of the forearm. In order to find the hand center, a distance transform (DST) is applied to the binary mask $\mathcal{H}'''$ and the maximum of the DST is located. More

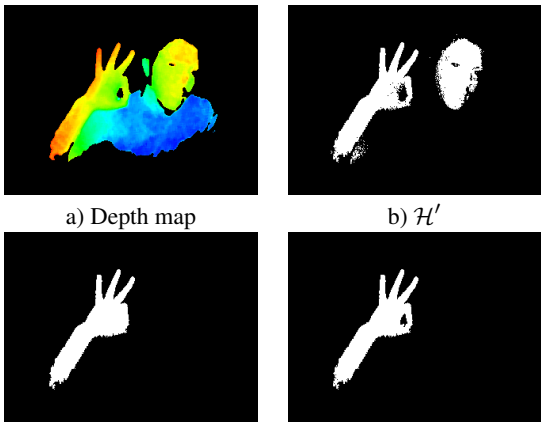
precisely a new data structure $D'(\mathbf{x}) = D'(u,v)$ containing the distance from each pixel $\mathbf{x} = (u,v)$ to the closest point not belonging to the mask is constructed, i.e.,

$$D'(u,v) = \min_{(j,k)\notin\mathcal{H}'''} (|u-j| + |v-k|) \tag{2}$$

where the Manhattan (i.e., d_1) distance has been used as the distance metric. The approach of [Bor86] has been used for the computation of the distance transform, an example of the results of the algorithm is shown in Figure 3a. The maximum of $D'(u,v)$,

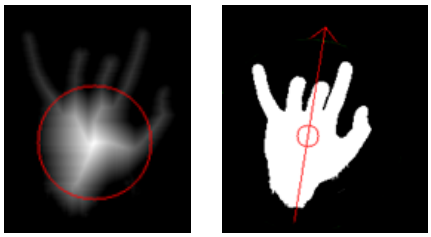
$$D'_{max} = \max_{(u,v)} \quad D'(u,v) \tag{3}$$

is computed and the point corresponding to the maximum of the DST is selected as the palm center C (if multiple pixels with the same highest distance transform value are present their barycenter is selected as the hand center). After locating the palm center, a circle of radius $R = 3D_{max}$ centered on C_H is computed and all the points outside the circle are excluded from the mask thus obtaining the final hand mask $H(u,v)$. Notice that this final step allows to remove the forearm if it has been included in the hand shape. The distance transform is also re-computed using this updated mask, thus obtaining a new set of distance values $D(u,v)$ that will be used for the construction of the distance features (see Subsection 4.3). The orientation of the hand is then obtained by computing the first two moments of the binary mask (see Figure 3b for an example). Finally the hand region is resampled to 100×100 pixels in order to normalize the data with respect to people with different hand sizes. The external contour of the mask H is then computed, leading to a sequence of points $\mathbf{b'}_1, ..., \mathbf{b'}_k$. The sequence is uniformly resampled into a sequence $\mathbf{b}_1, ..., \mathbf{b}_n$ with a constant number of samples n thus making the approach invariant to the number of pixels in the contour (for the results we used $n = 500$). This sequence will be used in the feature extraction step. The starting point $\mathbf{b}_1$ of the contour is selected as the one corresponding to the computed orientation, in order to make the approach also rotation invariant.



a) Depth map b) $\mathcal{H}'$ c) $\mathcal{H}''$ d) $\mathcal{H}'''$

Figure 2: *Computation of the initial hand mask: a) Depth map from the sensor; b) Binary mask from the thresholding operation ($H'(u,v)$); c) Area inside the longest contour ($H''(u,v)$); d) Intersection of the two masks ($H'''(u,v)$).*



a) b)

Figure 3: *a) Distance transform computed on a sample hand mask (the circle has a radius equivalent to the maximum of the distance transform); b) hand center and orientation obtained from the binary moments.*

4.2. Contour curvature features

The first set of features describes the curvature of the contour $\mathbf{b}_1, \ldots, \mathbf{b}_n$ computed in Subsection 4.1. The contour is analyzed and for each sample $\mathbf{b}_i$ the curvature is computed as the angle made between the set of the preceding samples and the set the subsequent ones. More precisely, let $\mathbf{p}_i$ and $\mathbf{s}_i$ be the barycenter of the k preceding samples and the barycenter of the k subsequent samples respectively (for the results we used $k = 5$), i.e.,:

$$\mathbf{p}_i = \frac{1}{k} \sum_{j=1}^{k} \mathbf{b}_{i-j} \tag{4}$$

$$\mathbf{s}_i = \frac{1}{k} \sum_{j=1}^{k} \mathbf{b}_{i+j} \tag{5}$$

Finally, the curvature c_i associated to the i-th sample is computed as the angle between the vectors $(\mathbf{b}_i - \mathbf{p}_i)$ and $(\mathbf{s}_i - \mathbf{b}_i)$, i.e.:

$$c_i = \arccos\left[(\mathbf{b}_i - \mathbf{p}_i) \cdot (\mathbf{s}_i - \mathbf{b}_i)\right] \tag{6}$$

As shown in Figure 4, the algorithm computes the deviation from the straight direction made by the contour in proximity of the i-th sample and thus each value c_i is included in the range $[-\pi, +\pi]$ (notice that also the orientation of the curvature is considered). Averaging over the k closest pixels in both directions makes the approach more stable with respect to the noise on the contour, specially when dealing with real data. The result of this computation is a sequence of values $\mathbf{c} = (c_1, \ldots, c_n)$ representing the local curvature at each location of the contour.

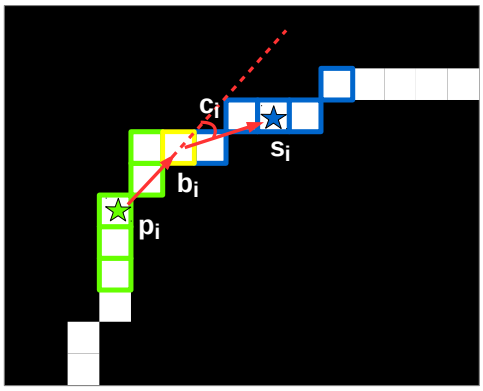


Figure 4: *Computation of the curvature value c_i at the location of sample $\mathbf{b}_i$. Point $\mathbf{p}_i$ (shown with a green star) is the barycenter of the preceding samples and $\mathbf{s}_i$ (shown with a blue star) is the barycenter of the subsequent ones.*

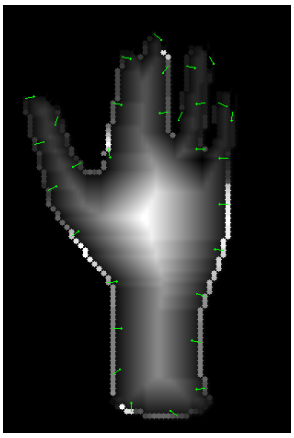


Figure 5: *Computation of the distance-based features: the green arrows represent the direction along which the maximum DST value is chosen.*

4.3. Distance-based features

The second descriptor represents the thickness of the hand region associated to each contour point. In order to compute this descriptor, the distance transform $D(u,v)$ computed in Subsection 4.1 on the hand mask $\mathcal{H}$ is exploited. For each contour sample $\mathbf{b}_i$ the direction $\mathbf{n}_i$ perpendicular to the contour is considered and the values of $D(\mathbf{x})$ along this direction are analyzed (see Fig. 5). The maximum value of the DST along the direction $\mathbf{n}_i$ going towards the center of the hand is then selected (the search stops when a maximum is found), i.e.,:

$$d_i = \max_k D(\mathbf{b}_i + \mathbf{n}_i \cdot k) \tag{7}$$

where k is the displacement from the contour in pixels that grows until a maximum is found (i.e., the values of D start to decrease).

At the end of this process a sequence of values $\mathbf{d} = (d_1, \ldots, d_n)$ is obtained, each value d_i giving the distance value associated to the i-th sample. As shown in Figure 5, higher values correspond to the palm area and to groups of fingers joined together, while the DST takes smaller values in the fingers region.

5. Gesture Recognition

After computing the two feature descriptors, their values are fed to the classification algorithm. The employed classifier is a Support Vector Machine (SVM) trained on a synthetic dataset built using the approach of Subsection 5.1. In order to perform the recognition, feature data are firstly rearranged in a three dimensional structure and then fed to the SVM classifier as described in Subsection 5.2.

5.1. Construction of the synthetic training dataset

A typical issue of many machine learning techniques is that a large dataset is required to properly train the classifier. The acquisition of training data is a critical task that can require a huge amount of manual work. In order to solve this problem a possible solution is to perform the training on a computer generated dataset that simulates the data that would have been acquired in a real setup. The rendering of an accurate hand shape is not an easy task, because of its anatomy and because of the large variety of poses that it can assume. A commonly used approach for the rendering of the human and hand shape in different poses is linear blend skinning. In this work this model is exploited together with a 42-DOF skeleton (see Figure 6): starting from the open source library *LibHand* v0.9 [Š11], we developed a new rendering library (that is available at `http://lttm.dei.unipd.it/downloads/handposegenerator`). It shares the textured 3D model of the hand used by *LibHand* but exploits an ad-hoc OpenGL renderer for a fast rendering pipeline. The depth information is exported in order to simulate the data that is provided by the depth camera in the real setup. The software also allows to control all the parameters of the renderer in order to get results as similar as possible to the real data. For each gesture we considered several different hand positions and orientations, different inter-distances of the fingers and we generated a large number of training frames. The synthetic depth images are a good representation of the real ones (see Figure 7), and a large dataset containing 35200 samples has been generated. There are 3200 different samples for each of the 11 gestures considered for the experimental results (see Section 6) corresponding to different orientation and small variations in the position of the fingers. In particular 5 different inclinations in the x-axis direction, 4 in the y-axis one and 5 for the z-axis have

been considered, thus obtaining 100 possible orientations. Different small perturbations of the positions of the fingers (e.g., a bit closer one to the other) have also been considered and 32 different variations of each pose have been generated. By combining all possible positions and orientation, $100x32 = 3200$ samples for each gesture have been obtained.

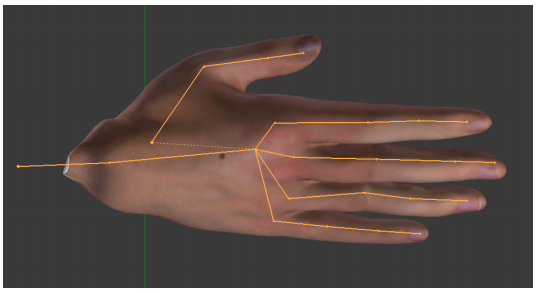


Figure 6: *42-DOF hand skeleton used by the rendering library.*

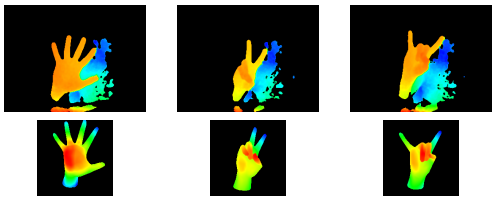


Figure 7: *Comparison of real data acquired by the ToF sensor (first row) and synthetic depth maps (second row) on 3 sample gestures.*

5.2. Classification of the feature data

Gesture recognition is performed by using a multi-class one-against-one SVM classifier, trained on the synthetic dataset presented in Subsection 5.1. For each input frame, a feature vector is computed starting from the contour curvature and distance-based features extracted as described in Section 4. An example of the content of the two feature vectors for a sample gesture is plotted in Figure 8, where the vectors **c** and **d** are shown by plotting each element value against its index. Rather than directly using the two vectors $\mathbf{c} = (c_1, \ldots, c_n)$ and $\mathbf{d} = (d_1, \ldots, d_n)$, the data have been rearranged in order to better capture significant correlations between the two descriptors.

A naïf approach is to feed the Support Vector Machine with a simple concatenation of **c** and **d**. Much of the effectiveness of this basic solution relies on how precisely the hand orientation is estimated for each frame, or at least on the invariance of the orientation information with respect to frames of the same gesture. Due to the not too precise estimation of the hand orientation this approach provided sub-optimal results in our experiments.

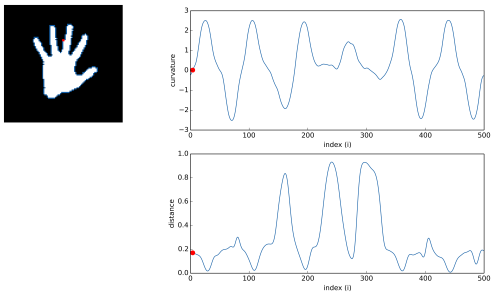


Figure 8: *Curvature and distance features for a sample gesture. The feature values have been plotted starting from the red point and proceeding in clockwise order along the hand contour.*

Another approach is to plot on a plane the discrete parametric curve given by (c_i, d_i), the parameter i taking values from 1 to n. An example of this representation is shown in Figure 9 that displays the curvature and distance couples (c_i, d_i) taking the index i as an implicit parameter. In this case, since the shape of the curve does not depend on which contour sample is chosen as the starting point, the orientation estimate plays no role in the descriptor computation performed with this approach. On the other hand, exploiting only this kind of features without exploiting the information about the variations of the curvature or of the distance with respect to the position of the samples may results in a significant drop in the prediction accuracy of the classifier, especially if a relatively good orientation estimate is available.

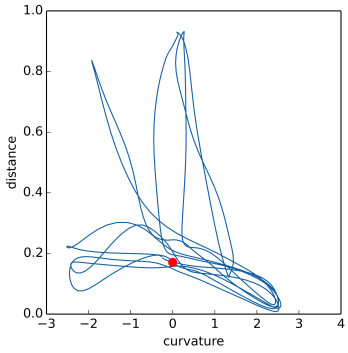


Figure 9: *Plot of the the parametric (c_i, d_i) curve for the sample gesture of Figure 8a.*

The idea exploited in this work is to make explicit the parameter i, or equivalently to look at the triples (c_i, d_i, i) for $i = 1, \ldots, n$ as coordinates in a three-dimensional space. An array representation of a quantized and smoothed version of the resulting 3D plot is computed and used as the

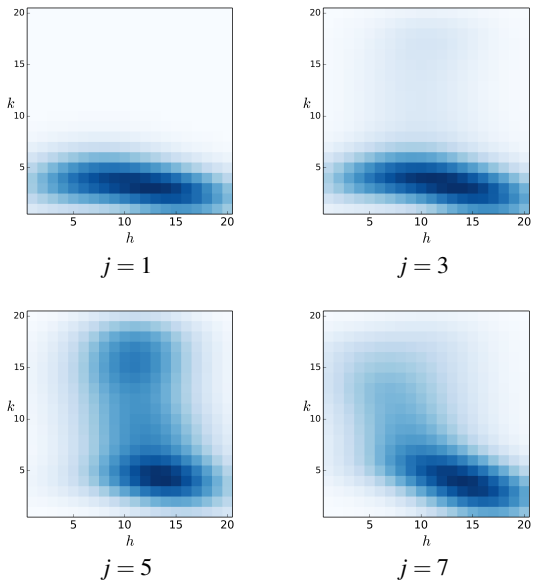


Figure 10: *Example of the three-dimensional array* **A** *corresponding to the gesture in Figure 8 for some sample values of the index j. The employed parameters are* $n_c = 20$, $n_d = 20$, $n_i = 10$, $\sigma_c = 4.0$, $\sigma_d = 1.5$, $\sigma_i = 1.0$.

feature vector to be given in input to the classifier. More precisely, the ranges of possible values of $c_i \in [0, 2\pi]$ (curvature values are shifted from $[-\pi, \pi]$ to $[0, 2\pi]$ in order to simplify the representation) , $d_i \in [0,1]$ and $i \in [1,n]$ are divided into three set of quantization intervals, i.e., $I_c = \{1,\ldots,n_c\}$, $I_d = \{1,\ldots,n_d\}$ and $I_i = \{1,\ldots,n_i\}$ respectively. A quantization function is then used mapping each triple (c_i, d_i, i) to the corresponding combination of quantization intervals, represented by a triple of integers (j,h,k) in $I_c \times I_d \times I_i$. A three dimensional array **A** of size $n_c \times n_d \times n_i$ is created where the entry $\mathbf{A}(j,h,k)$ is set to 1 if there exists at least one triplet (c_i, d_i, i) which is mapped to $(j,h,k) \in I_c \times I_d \times I_i$ by the quantization function, and it is set to 0 otherwise. Finally, a multi-dimensional Gaussian filter is applied to the array, using suitable standard deviations σ_c, σ_d, and σ_i for each dimension along which the filter is applied. In order to handle the boundaries, before applying the filter, the array is extended outside its borders with zeros along the first two dimensions (those relative to the intervals I_c and I_d), while wrapping is performed along the third dimension (the one accounting for intervals I_i). An example of the resulting representation is given in Figure 10, where four slices along the third dimension of the 3D array are shown. The array is computed from the same input gesture acquisition of Figure 8, and the different intensities of blue are used to represent the element values ranging from 0 (white) to 1 (dark blue).

By varying both the granularity of the quantization intervals I_i and the amount of Gaussian smoothing applied along

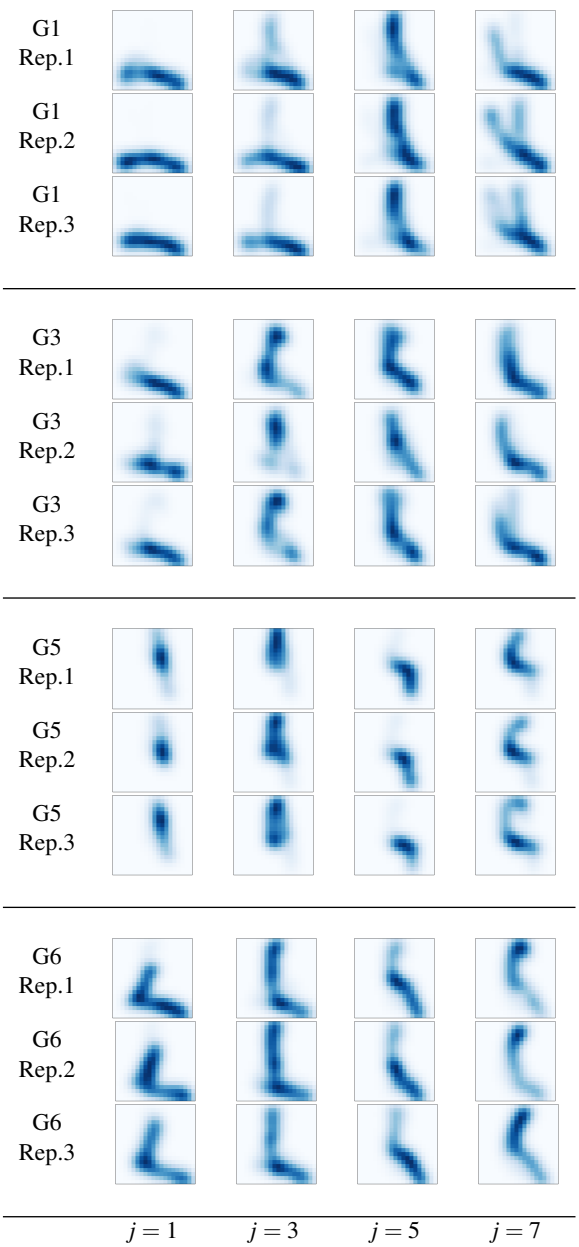


Figure 11: *The figures shows 4 different slices of the feature array* **A** *for 3 different repetitions of 4 different gestures (i.e., gestures G1, G3, G5 and G6). Notice how the slices are similar for different repetitions of the same gesture and very different among different gestures.*

the third dimension (i.e., varying σ_i), it is possible to implicitly reduce the weight that the position information of the feature values (and consequently the orientation information) has in describing the different gestures. It is also possible to vary the importance of curvature features over distance features by conveniently adjusting both the number of intervals n_c, n_d and the standard deviations σ_c, σ_d relative to the first and second dimension.

For the experimental results $n_c = 20$, $n_d = 20$ and $n_i = 10$ have been used to perform quantization. The standard deviations of the Gaussian filter have been set to $\sigma_c = 1.5$, $\sigma_d = 1.5$ and $\sigma_i = 1.0$. An additional smoothing has also been applied separately to both curvature and distance vectors $\mathbf{c}$ and $\mathbf{d}$ as a pre-processing step before computing the three-dimensional array. In particular, a 1-dimensional Gaussian filter with standard deviation $\sigma = 2.0$ has been used to smooth curvature values. For the distance vector a standard deviation $\sigma = 4.0$ has instead been used.

In Figure 11 the computed features are reported for three different repetitions of 4 different gestures (gestures G1, G3, G5 and G6). For each repetition, four slices of $\mathbf{A}$ are displayed corresponding to values $j = 1,3,5,7$, notice how the basic shape of the descriptor remains the same along the various repetitions. By looking also at the different gestures in the figure it is possible to realize that the shapes are characteristic of the performed gesture and allow a very good discrimination between the different gestures.

6. Experimental Results

In order to evaluate the proposed approach an hand gesture dataset has been acquired with a Creative Senz3D camera (also known as SoftKinetic DepthSense 325).The device has both a RGB camera and a Time-Of-Flight (ToF) depth sensor. The depth resolution is 320 per 240 pixels, with a frame rate ranging from 6 to 60. The automatic confidence threshold function of the sensor has also been used to obtain a better starting depth map while color data has not been used for this work. The dataset contains 11 different gestures performed by 4 different people. A sample color and depth frame for each gesture is shown in Figure 12, notice how the dataset contains different gestures with the same number of raised fingers, gestures with fingers very close each other and with fingertips touching each other. Each gesture has been repeated by each user 30 times for a total of 1320 acquisitions. The dataset is available at `http://lttm.dei.unipd.it/downloads/gesture2`.

Table 1 shows the results obtained by training the classifier with the approach of Section 5.1 on the synthetic dataset and then performing the testing on real data contained in the acquired dataset. Most of the gestures are correctly recognized and an average accuracy of 90% has been obtained. This is a quite remarkable result considering that the training has been performed on a synthetic dataset without any real acquisition. The synthetic data proved to be very similar to real acquisitions and the extracted features are stable across different repetitions of the same gesture but at the same time they are able to discriminate different gestures. Looking more in detail at the results in Table 1 it is possible to notice how the accuracy is above 80% for all the considered gestures and very close or above 90% for most of them. The most critical gestures are G2, G7 and G9. G2 is sometimes confused with G3 since they differ only for the position of the thumb that in some acquisitions is not very easy to detect from the silhouette. G9 is another challenging gesture due to the touching fingers. It is recognized 82% of the times, a good result considering that many approaches based on fingertips detection are typically completely unable to handle this gesture. G7 and G11 both have a single raised finger and sometimes are confused each other. This is a typical problem for many gesture recognition approaches since orientation information from PCA or moments computation, that should be the key to disambiguate such configurations, is usually not very accurate. The three-dimensional data structure allows to improve performances on such configurations, but there is still room for further improvements.

With regard to the effectiveness of this arrangement of the features, it is interesting to look at Table 2 which reports, in the left column, the per-gesture accuracies obtained by feeding the SVM with a simple concatenation of the curvature features vector $\mathbf{c}$ with the distance-based feature vector $\mathbf{d}$. These results are compared with those achieved by the proposed approach (that exploits the three-dimensional structure to combine the two type of features), which are listed in the rightmost column. The table clearly shows how the proposed scheme allows to significantly improve the recognition accuracy. The effectiveness of both $\mathbf{c}$ and $\mathbf{d}$ descriptors depends on how well the hand orientation is estimated, since the two vectors are not invariant to rotations. Small variations in these estimates for samples capturing the same gesture may lead to poor classification results, as it can be noticed from the table.

The proposed approach has also very limited computational requirements (see Table 3). The hand segmentation requires $3.1ms$ on average on a standard PC (we performed the tests on an Intel i5-2430 CPU running at a $2.4Ghz$). The extraction of the features can be computed in only $0.83ms$ while SVM classification takes around $1ms$ on average. Summing up the three steps, the average total execution time is less than $5ms$ for each analyzed frame. This corresponds to a frame rate of $200fps$ that makes the approach very well-suited for real-time gesture recognition tasks. Furthermore the proposed approach does not exploit multi-core architectures or the GPU, even better performances can be obtained with a fully optimized implementation.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11
G1	**0.94**	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00
G2	0.00	**0.84**	0.12	0.00	0.00	0.03	0.00	0.00	0.01	0.00	0.00
G3	0.00	0.00	**1.00**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
G4	0.00	0.00	0.00	**0.88**	0.09	0.00	0.02	0.00	0.00	0.00	0.02
G5	0.00	0.00	0.00	0.06	**0.88**	0.00	0.00	0.00	0.00	0.06	0.00
G6	0.00	0.03	0.00	0.00	0.00	**0.88**	0.00	0.00	0.04	0.00	0.05
G7	0.00	0.00	0.01	0.00	0.00	0.00	**0.80**	0.00	0.00	0.00	0.19
G8	0.02	0.02	0.00	0.00	0.00	0.00	0.00	**0.93**	0.04	0.00	0.00
G9	0.00	0.12	0.00	0.00	0.00	0.00	0.00	0.05	**0.82**	0.00	0.01
G10	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	**0.92**	0.00
G11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**

Table 1: *Performance of the proposed approach on the acquired dataset. The Table shows the confusion matrix for the 11 different gestures contained in the dataset. Due to rounding effects some rows can sum up to* 0.99.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	All
$\mathbf{c}+\mathbf{d}$	**0.97**	0.53	0.84	0.72	0.66	0.77	0.77	0.28	0.17	**0.93**	0.37	0.64
3D array	0.94	**0.84**	**1.00**	**0.88**	**0.88**	**0.88**	**0.80**	**0.93**	**0.82**	0.92	**1.00**	**0.90**

Table 2: *Per-gesture accuracies achieved by exploiting the concatenation feature vector* $\mathbf{c}+\mathbf{d}$ *(first row) and by the proposed three-dimensional feature array (second row). Results in the first row were obtained using the best configuration returned by a grid search over a number of parameters controlling sub-sampling and smoothing of* $\mathbf{c}$ *and* $\mathbf{d}$*.*

Step	*Execution Time*
Hand Segmentation	3.1 ms
Feature Extraction	0.83 ms
SVM Classification	1 ms
Total	**4.93 ms**

Table 3: *Average execution time of the various steps of the proposed approach. The test have been performed on a standard PC with an Intel i5 processor.*

7. Conclusions

In this paper a novel real-time hand gesture recognition scheme has been presented. Two different feature sets have been developed ad-hoc for this problem, based on the contour curvature and on the shape thickness represented through a distance transform. The extracted features have proved to be discriminative and at the same time stable across different repetition of the same gesture. By arranging them in a smoothed three-dimensional data structure a representation suited to be used in an SVM classifier has been obtained. The critical problem of the classifier training has been solved by developing a synthetic rendering application able to create realistic synthetic depth maps that can be used for the training. Experimental results showed how the classifier trained on synthetic data is able to generalize to real data obtaining a 90% accuracy on a challenging real dataset. Furthermore the proposed approach has very limited computational requirement and is able to analyze each frame in less than 5*ms* on average. Further research will be devoted to the introduction of novel feature descriptors and to the extension of the proposed approach to the recognition of dynamic gestures exploiting also the temporal information.

References

[Bor86] BORGEFORS G.: Distance transformations in digital images. *Computer vision, graphics, and image processing 34*, 3 (1986), 344–371. 3

[DDM*13] DOMINIO F., DONADEO M., MARIN G., ZANUTTIGH P., CORTELAZZO G. M.: Hand gesture recognition with depth data. In *Proceedings of the 4th ACM/IEEE international workshop on Analysis and retrieval of tracked events and motion in imagery stream* (2013), ACM, pp. 9–16. 2

[DDZ14] DOMINIO F., DONADEO M., ZANUTTIGH P.: Combining multiple depth-based descriptors for hand gesture recognition. *Pattern Recognition Letters* (2014), 101–111. 2

[DLK14] DINH D.-L., LEE S., KIM T.-S.: Hand number gesture recognition using recognized hand parts in depth images. *Multimedia Tools and Applications* (2014), 1–16. URL: http://dx.doi.org/10.1007/s11042-014-2370-y, doi:10.1007/s11042-014-2370-y. 2

[DMZC12] DAL MUTTO C., ZANUTTIGH P., CORTELAZZO G. M.: *Time-of-Flight Cameras and Microsoft Kinect*. SpringerBriefs in Electrical and Computer Engineering. Springer, 2012. 1

[KOWW15] KAPUSCINSKI T., OSZUST M., WYSOCKI M., WARCHOL D.: Recognition of hand gestures observed by depth cameras. *Int J Adv Robot Syst 12* (2015), 36. 2

[KZL12] KURAKIN A., ZHANG Z., LIU Z.: A real-time system for dynamic hand gesture recognition with a depth sensor. In *Proc. of EUSIPCO* (2012). 2

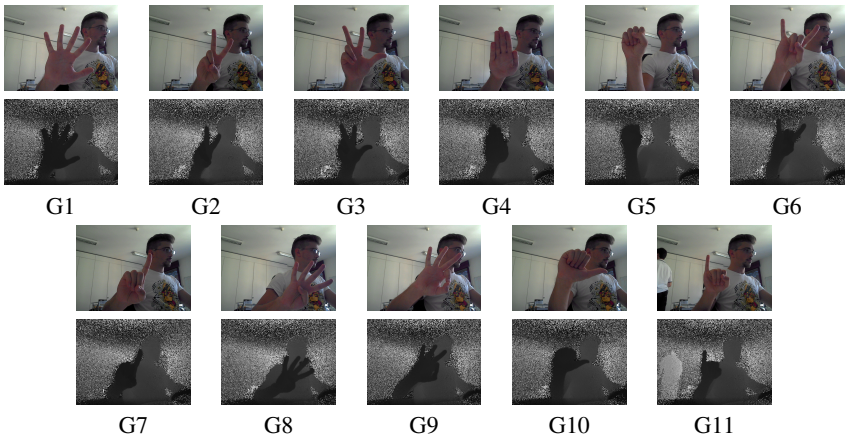


Figure 12: *Sample color and depth frame for each of the gestures in the experimental results dataset.*

[MDZ14] MARIN G., DOMINIO F., ZANUTTIGH P.: Hand gesture recognition with leap motion and kinect devices. In *Proceedings of IEEE International Conference on Image Processing (ICIP)* (2014), IEEE, pp. 1565–1569. 2

[NLD*13] NANNI L., LUMINI A., DOMINIO F., DONADEO M., ZANUTTIGH P.: Ensemble to improve gesture recognition. *International Journal of Automated Identification Technology*, 5 (2013), 47–56. 2

[QZYJ14] QIN S., ZHU X., YANG Y., JIANG Y.: Real-time hand gesture recognition from depth images using convex shape decomposition method. *Journal of Signal Processing Systems 74*, 1 (2014), 47–58. doi:10.1007/s11265-013-0778-7. 2

[RMY11] REN Z., MENG J., YUAN J.: Depth camera based hand gesture recognition and its applications in human-computer-interaction. In *Proc. of ICICS* (2011), pp. 1 –5. 2

[RYZ11] REN Z., YUAN J., ZHANG Z.: Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proc. of ACM Conference on Multimedia* (2011), ACM, pp. 1093–1096. 2

[SSM10] SURYANARAYAN P., SUBRAMANIAN A., MANDALAPU D.: Dynamic hand pose recognition using depth data. In *Proc. of ICPR* (aug. 2010), pp. 3105 –3108. 2

[Š11] ŠARIĆ M.: Libhand: A library for hand articulation, 2011. Version 0.9. URL: http://www.libhand.org/. 4

[WKSE11] WACHS J. P., KÖLSCH M., STERN H., EDAN Y.: Vision-based hand-gesture applications. *Commun. ACM 54*, 2 (Feb. 2011), 60–71. 1

[WLC*12] WANG J., LIU Z., CHOROWSKI J., CHEN Z., WU Y.: Robust 3d action recognition with random occupancy patterns. In *Proc. of ECCV* (2012). 2

[ZT15] ZHANG C., TIAN Y.: Histogram of 3d facets: A depth descriptor for human action and hand gesture recognition. *Computer Vision and Image Understanding* (2015). 2