Feature descriptors for depth-based hand gesture recognition

Fabio Dominio, Giulio Marin, Mauro Piazza and Pietro Zanuttigh
Department of Information Engineering
University of Padova, Italy

***Abstract*—Depth data acquired by consumer depth cameras provide a very informative description of the hand pose that can be exploited for accurate gesture recognition. A typical hand gesture recognition pipeline requires to identify the hand, extract some relevant features and exploit a suitable machine learning technique to recognize the performed gesture. This chapter deals with the recognition of static poses. It starts by describing how the hand can be extracted from the scene exploiting depth and color data. Then several different features that can be extracted from the depth data are presented. Finally a multi-class SVM classifier is applied to the presented features in order to evaluate the performance of the various descriptors.**

I. Introduction

Hand gesture recognition is an intriguing problem for which many different approaches exist. Even if gloves and various wearable devices have been used in the past, vision-based methods able to capture the hand gestures without requiring to wear any physical device allow a more natural interaction with computers and many other devices. This problem is currently raising a high interest due to the rapid growth of application fields where it can be efficiently applied, as reported in recent surveys (e.g. [1], [2]). However hand gesture recognition from images or video data is a very challenging task. The hand and fingers can assume a huge variety of poses and there are often several inter-occlusions between the various fingers. Furthermore the skin has a relatively uniform color that does not help feature extraction and recognition schemes. For all these reasons it is difficult to recognize complex gestures from the 2D representation given by a single image or from a video.

The introduction of Microsoft's Kinect sensor has made the acquisition of 3D information available to the mass marker, thus paving the way for new solutions for many challenging computer vision problems, including object tracking and recognition, human activity analysis, indoor 3D mapping and also hand gesture recognition. A complete review of them is presented in [3]. In the computer gaming field, Microsoft's Kinect has already brought gesture interfaces to the mass market, but many new application fields are being considered. These include human-computer interaction, 3D navigation, robotics, gaming, sign-language recognition, vehicle control and many others. Hand gestures can be used to replace the mouse in computer interfaces and to allow a more natural interaction with mobile devices like smartphones and tablets, but also with newer wearable devices like the Google glasses and other similar tools. Besides controlling standard 2D interfaces, a very interesting field is the interaction with 3D virtual environments, that is much more natural if gestures performed in the 3D space are used. Another key application is automatic sign-language interpretation, that would allow hearing and speech impaired people to interact with computers and other electronic devices. Finally, the healthcare field is another field in which hand gesture recognition can be used for a more natural control of diagnostic data and surgical devices.

Depth data intrinsically contains a very informative three-dimensional description of the hand pose, which can be exploited for gesture recognition. The simplest approaches just use depth data to reliably extract the hand silhouette. This allows to directly apply many methods derived from color-based hand gesture recognition, but only exploits a limited amount of the information contained in depth data. This chapter shows how several features based on the three-dimensional data in the depth map, representing the hand shape and the finger posture, can be extracted and used to properly recognize complex gestures.

In order to show how depth data can be used in hand gesture recognition, it is assumed that a generic hand gesture recognition scheme encompasses three main steps:

- Hand extraction
- Feature extraction
- Gesture classification

The chapter follows this subdivision and the following sections describe the three main steps. Section III explains how the hand can be recognized both from depth and color data, by also considering its segmentation into arm, palm and fingers regions. Then the extraction of feature descriptors is analyzed in detail in Section IV. Several different possible descriptors are presented, including 3D distances from the palm center or from the palm plane, descriptors based on the contour of the hand (e.g., curvature features or approaches based on the perimeter length), convex hull based features, and other schemes. In order to compare the efficiency of the various features, a classification scheme based on Support Vector Machines (SVM) is presented in Section V and the obtained accuracies from a real dataset are reported in Section VI. Finally, Section VII draws the

conclusions.

II. Related work

Hand gesture recognition has been the subject of a large amount of research activity. Until a few years ago, computer vision approaches were only based on the images or videos framing the hand. These methods typically exploit the shape of the hand silhouette, color or motion information to extract relevant features to be used for the gesture recognition. A complete overview of these approaches is out of the scope of this chapter, which focuses on depth information. Complete reviews of this field can be found in [1] and in [4].

The recent availability of consumer depth cameras based on structured light, like the Kinect or the Asus Xtion, and of matricial Time-Of-Flight sensors, e.g., the Creative SENZ3D or Mesa's Swiss Ranger, has opened the way to a new family of approaches exploiting depth information. Depth data offer an accurate description of the hand pose and compared to images and videos there are several advantages. They include the possibility of capturing the 3D orientation of the hand parts, the availability of metric measures in the 3D space and much simpler options for the hand segmentation.

The basic pipeline of most methods consists in the three steps indicated in Section I. Hand segmentation is typically solved by a simple thresholding of the depth data [5], [6]. A common assumption used in this step is that the hand is the closest object to the sensor. After extracting the samples corresponding to the hand only, several different methods may be used for the following steps. A first family of approaches is based on the hand silhouette extracted from the depth data: in [7] silhouette and cell occupancy features are extracted from the depth map and used for building a shape descriptor that is then fed into a classifier based on action graphs. Histograms of the distance of the hand points from its center, extracted from the silhouette, are used in [6] and [8]. Other approaches in this family use features based on the convex hull and on the fingertips positions computed from the silhouette, as in [9] and [10]. The convex hull is also exploited in the open source library *XKin* [11], [12].

Another possibility is computing descriptors based on the volume occupied by the hand. In [13], 3D volumetric shape descriptors are extracted from the depth map and fed into a Support Vector Machine (SVM) classifier. A similar approach is exploited by [14].

Color data can also be used together with the depth data, as in [15], that is based on Randomized Decision Forests (RDFs). RDFs are also used by Keskin et Al [16].

All these approaches are focused on the recognition of static poses, other methods, instead, deal with dynamic gestures. For example, [17] exploits motion information, and in particular the trajectory of the hand centroid in the 3D space, for recognizing dynamic gestures. Depth and color information are used together in [18] to extract the trajectory that is then fed to a Dynamic Time Warping (DTW) algorithm. Finally, Wan et Al [19] exploit both the convex hull on a single frame and the trajectory of the gesture.

Among the various application of hand gesture recognition, sign language recognition is one of the most interesting. An approach for sign language recognition with the Kinect is proposed in [20].

A different but related problem is the extraction of the 3D hand pose, which can then be exploited for gesture recognition. Approaches exploiting depth data for this task are [21], [22] and [23].

III. Hand extraction

The first step of the considered gesture recognition pipeline, indicated in Section I, consists in segmenting the hand from the rest of the scene, since all the information of the performed gesture is entirely contained in the hand region. The region of the arm is usually discarded, as it does not contain any helpful information about a particular gesture and its shape and size are affected by the presence of sleeves and bracelets. Hand extraction is a crucial step because all the following processing is performed on the segmented region only.

Different methods have been used over the years to tackle this problem, exploiting different clues like appearance, shape, color, depth, context, and tracking. In order to take advantage of both the color and the depth map from the sensor, a joint calibration of color and depth camera [24] is required. Joint calibration, in fact, allows to associate a color and a depth value to each point in the framed scene .

The most common methods based on appearance exploit cascade classifiers based on Haar-like features [25]. However, differently from faces, which have fixed properties related to the position of the mouth, eyes and nose, hands have many degrees of freedom, and so this technique does not produce satisfactory results.

A common scenario is to have users facing the camera with their hand held in front of themselves. This allows to assume that the hand is the closest object to the sensor. In this case, a simple threshold in depth range can be used to isolate the hand. Additional geometric constraints in the hand aspect ratio and size may be used to refine the segmentation, as in [26]. Other approaches exploiting only the depth map, use clustering algorithms such as K-means, iterative seed fill or region growing to separate the hand region from the rest of the scene. In [27], the depth range is fixed and a flood fill algorithm is used to cluster contiguous points with the aim of separate the hand from the body. In [10] instead, the K-means algorithm with 2 clusters is used in a limited depth range to find the hands.

The assumption that the hand has to be the closest object in the scene can be relaxed by predicting hand depth according to the position of other body parts, such as the face. In addition, since the color image is available as well, skin color segmentation can be used to enforce the hand detection. In [28], skin color segmentation based on both

a model trained offline and a further online histogram-based refinement are used to get an initial guess for hand detection. Then, the user face is detected and all the points not belonging to a predefined region in front of the face are rejected. Once the hand is detected, the arm is removed by exploiting the depth and other geometrical constraints.

Color data can be combined with the depth in order to improve the accuracy and better segment the hand from the arm, as in [29]. Other approaches also exploit some physical aids, e.g. in [6], after thresholding the depth map, a black bracelet on the gesturing hand's wrist is recognized in the color image for an accurate hand detection.

More reliable approaches exploit the temporal redundancy to better find and segment the hand, reducing false positive detection. For example, [7] first divide depth map into a number of blobs using a connected-component labeling algorithm, then, for the biggest blob that is assumed to include the body and the hand, compute blob tracking, the blob with the highest track is associated to the hand. Additional geometric constraints are used to identify and remove points of the wrist region.

Once the hand has been extracted, it is also necessary to estimate its position and orientation to ensure that the recognition is rotational and scale invariant. Scale invariance can be easily obtained since, if calibration data are available, depth provide metric measures in the 3D space. The orientation can be computed by detecting the principal direction, e.g. with Principal Component Analysis (PCA) or by fitting a plane on the hand point cloud.

An example of efficient hand segmentation scheme that exploits both color and depth information is presented in [30] and is briefly recalled here. This approach also allows to extract useful information for features extraction that will be exploited in the next section. The acquired depth map $D(u,v)$ is firstly thresholded on the basis of color information. More specifically, the colors associated to the samples are converted into the CIELAB color space and compared with a reference skin color that has been previously acquired, e.g. from the face by using a standard face detector [25]. After the skin color thresholding, the hand region has a higher chance to be the nearest object to the camera.

Let $\mathbf{X}_i = \mathbf{X}_{u,v}$ denote a generic 3D point acquired by the depth camera, that is, the back-projection of the depth sample in position (u,v). A search for the sample with the minimum depth value D_{min} on the thresholded depth map, avoiding isolated artifact, is performed and the corresponding point $\mathbf{X}_{min}$ is chosen as the starting point for the hand detection procedure. Points belonging to the hand samples set $\mathcal{H}$ are the ones whose distance from $\mathbf{X}_{min}$ does not exceed a predefined threshold T_{max}:

$$\mathcal{H} = \{\mathbf{X}_i \mid ||\mathbf{X}_i - \mathbf{X}_{min}|| < T_{max}\} \quad (1)$$

This algorithm allows to reliably segment the hand samples from the scene objects and from the other body parts. An example of a thresholded depth map obtained with this approach is shown in Fig. 1c.

In order to extract some of the feature sets described in Section IV, it is first necessary to obtain some additional information such as the hand orientation, its centroid, the palm and fingers regions. A possible approach is to consider the binary mask of the hand samples in the depth image and filter it with a Gaussian kernel obtaining a result similar to the one in Fig. 1d. The resulting blurred image represents the hand samples density, where the luminance is directly proportional to the points density, and the point with the maximum intensity (that is, the one representing the maximum density) is chosen as initial value of the palm center. This value can then be refined by fitting a circle or an ellipse [31] whose center and size are updated iteratively by ensuring that most of the points belong to the hand samples. When the process converges, the points of the hand within the fitted shape will give an estimate of the palm region, while the 3D point corresponding to the center of the fitted shape will be the *centroid* $\mathbf{C}$ of the hand. Note that the *centroid* plays an important role in the features extraction, as most of them strongly depend from the centroid position. At this point, the hand region $\mathcal{H}$ can be segmented into three regions:

- $\mathcal{P}$ containing points corresponding to the hand palm.
- $\mathcal{W}$ containing the points of $\mathcal{H}$ lying on the sub-space below $\mathcal{P}$. Such samples typically belong to the wrist and forearm and are not considered in most gesture recognition schemes.
- $\mathcal{F}$ containing the points of $\mathcal{H} - \mathcal{P} - \mathcal{W}$, which corresponds to the fingers region.

It is useful also to define the set $\mathcal{H}_e = \mathcal{P} + \mathcal{F}$ containing the points of the palm and the fingers, and its projection $\mathcal{B}$ on the binary mask.

Once all the possible palm samples have been detected, a 3D plane π can be fitted on them by using SVD and RANSAC to get its position in 3D space. Principal Component Analysis (PCA) can be applied, instead, to the 3D points in $\mathcal{H}$ in order to extract the main axis that roughly corresponds to the direction $\mathbf{i}_x$ of the vector going from the wrist to the fingertips.

In order to build a 3D coordinate system centered on the palm centroid projected on π, i.e. $\mathbf{C}^\pi$, the axis $\mathbf{i}_x$ is projected on plane π obtaining $\mathbf{i}_x^\pi$, then $\mathbf{i}_z^\pi$ is the normal to plane π, and the remaining axis $\mathbf{i}_y^\pi$ is obtained by the cross-product of $\mathbf{i}_z^\pi$ and $\mathbf{i}_x^\pi$ thus forming a right-handed reference system $(\mathbf{i}_x^\pi, \mathbf{i}_y^\pi, \mathbf{i}_z^\pi)$, as depicted in Fig. 1g.

Finally, the information on the hand orientation and the palm radius can be used to remove all the wrist samples. Let R be the estimated palm radius and $\mathbf{X}_i^\pi = (x_i^\pi, y_i^\pi, z_i^\pi)$ the coordinates of an hand sample $\mathbf{X}_i$ with respect to the palm 3D coordinate system: it is possible to assume that $\mathbf{X}_i^\pi$ belongs to the wrist whenever $x_i^\pi < -R$ (recall that the x axis points form the palm center to the fingertips as shown in Fig.1g).

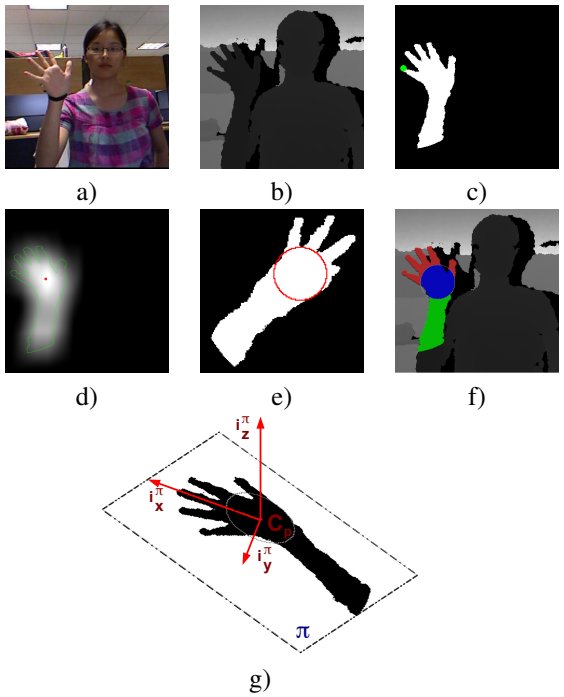


Fig. 1: Extraction of the hand and palm samples: a) Acquired color image; b) Acquired depth map; c) Extracted hand samples (the closest sample is depicted in green); d) Output of the Gaussian filter applied on the mask corresponding to $\mathcal{H}$ with the maximum in red; e) Circle fitted on the sample gesture; f) Palm (blue), finger (red) and wrist (green) regions subdivision; g) Reference system $(\mathbf{i}_x^\pi, \mathbf{i}_y^\pi, \mathbf{i}_z^\pi)$.

IV. Extraction of the Relevant Features

The next step consists in extracting feature sets from the segmented hand data which will be used for the classification of the performed gestures. Several different types of features have been proposed in the literature, as shown in Section II, based on both color and depth data. This section focuses on the information that can be extracted from depth data and presents a set of possible features to be used for reliable gesture recognition. The considered features are:

- **Distance features:** describe the Euclidean 3D distances of the fingertips from the estimated palm center.
- **Elevation features:** account for the Euclidean distances of the fingertips from a plane fitted on the palm samples. Such distances may also be considered as the *elevations* of the fingers with respect to the palm.
- **Contour histogram similarity features:** account for the similarity between the contour points distances in the performed gesture and in the reference acquisitions of the various gestures.
- **Curvature features:** describe the curvature of the contour of the palm and fingers regions.
- **Palm area features:** describe the shape of the palm region and helps to state whether each finger is raised or bent on the palm.
- **Convex hull features:** various feature can be extracted from the convex hull of the hand shape in the depth map. They include the ratios between the area or the perimeter of the hand shape to the one of its convex hull and the number of vertexes in the convex hull.
- **Connected components features:** they are based on the connected components in the difference between the hand shape and its convex hull. The dimensions and the number of the connected components can be used as features.

A. Distance features

The computation of this feature set starts from the construction of an histogram representing the distance of the edge samples in $\mathcal{F}$ from the hand centroid $\mathbf{C}^\pi$. Here a brief description of the approach of [30] and [29], that in turn extends the scheme of [6], is given.

Let R be the 3D radius of the circle back-projected to the plane π, if the more accurate fitting model with the ellipse is employed, R represents the distance from C^π to the edge of the ellipse and is not a constant value. For each 3D point $\mathbf{X}_i \in \mathcal{F}$, the normalized distance from the centroid $d_{\mathbf{X}_i} = \|\mathbf{X}_i - \mathbf{C}^\pi\| - R$ and the angle θ_{X_i} between vector $\mathbf{X}_i^\pi - \mathbf{C}^\pi$ and axis $\mathbf{i}_x^\pi$ on the palm plane π (where $\mathbf{X}_i^\pi$ is the projection of $\mathbf{X}_i$ on π) are computed. Then θ is quantized with a uniform quantization step Δ into a discrete set of values θ_q. Each θ_q thus corresponds to an angular sector $\mathcal{I}(\theta_q) = \theta_q - \frac{\Delta}{2} < \theta \leq \theta_q + \frac{\Delta}{2}$, and the farthest point inside each sector $\mathcal{I}(\theta_q)$ is selected thus producing a histogram $L(\theta)$:

$$L(\theta_q) = \max_{\mathcal{I}(\theta_q)} d_{\mathbf{X}_i} \tag{2}$$

For each gesture in the dataset, a reference histogram $L_g^r(\theta)$ of the type shown in Fig. 2 is built. A set of angular regions corresponding to the raised fingers intervals in each gesture (shown in Fig. 2) is also defined and will be used for computing the features.

The hand direction estimated by means of the PCA main axes is not very precise and furthermore is affected by several issues, e.g. the number of raised fingers in the performed gesture and the size of the retained wrist region after hand detection. The generated distance histogram therefore may not be aligned with the gesture templates, and a direct comparison of the histograms in this case is not possible. In order to compare the performed gesture histogram with each gesture template, they are firstly aligned by looking for the argument maximizing the cross-correlation between the acquired histogram and the translated version of the reference histogram of each gesture[1]. 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

[1]In Equations (3) and (4) L is considered as a periodic function with period 2π.

is also considered, by evaluating:

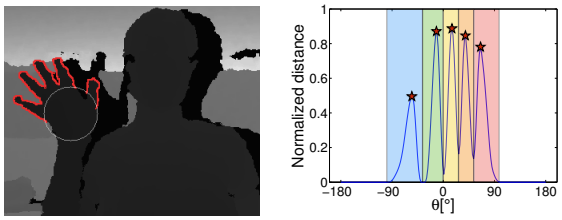
$$\Delta_g = \underset{\Delta}{\text{argmax}} \left(\rho\left(L(\theta), L_g^r(\theta+\Delta)\right)\right)$$
$$\Delta_g^{rev} = \underset{\Delta}{\text{argmax}} \left(\rho\left(L(-\theta), L_g^r(\theta+\Delta)\right)\right) \quad (3)$$

where symbol $\rho(a(\cdot), b(\cdot))$ denotes the value of the cross correlation between $a(\cdot)$ and $b(\cdot)$. The translational shift Δ that gives the maximum of the correlation of either $L(\theta)$ or $L(-\theta)$ in (3) is used to align the acquired histogram with the reference histograms of each gesture. Let $L_g(\theta)$ denote the histogram aligned to the gesture reference histogram $L_g^r(\theta)$. The translational shift to be applied to $L(\theta)$ will be either Δ_g and Δ_g^{rev} depending on the one maximizing the correlation, i.e. $L_g(\theta)$ is defined as:

$$L_g(\theta) = \begin{cases} L(\theta - \Delta_g) & \text{if } \rho\left(L(\theta), L_g^r(\theta+\Delta_g)\right) \geq \\ & \rho\left(L(-\theta), L_g^r(\theta+\Delta_g^{rev})\right) \\ L(-\theta - \Delta_g^{rev}) & \text{otherwise} \end{cases} \quad (4)$$

Note that there can be a different alignment Δ_g for each gesture, and that different regions can be defined in each gesture reference histogram corresponding to the various features of interest. This approach basically compensates for the limited accuracy of the direction computed by the PCA.

The alignment procedure solves one of the main issues related to the direct application of the approach of [6]. Fig. 3 shows some examples of the computed histograms for three different gestures. Note that the fingers raised in the various gestures are clearly visible from the plots.



a) b)

Fig. 2: Histogram of the edge distances with the corresponding feature regions: a) finger edges $\mathcal{F}$; b) associated histogram $L(\theta)$ with the regions corresponding to the different features $f_{g,j}^l$ (feature points highlighted with red stars).

If the database has G different gestures to be recognized, the feature set contains a value for each finger $j \in \{1, .., 5\}$ in each gesture $g \in \{1, .., G\}$. The feature value $f_{g,j}^l$ associated to finger j in gesture g corresponds to the maximum of the aligned histogram within the angular region $\mathcal{I}(\theta_{g,j}) = \theta_{g,j}^{min} < \theta < \theta_{g,j}^{max}$ associated to finger j in gesture g (see Fig. 2), i.e. :

$$f_{g,j}^l = \frac{\max\limits_{\mathcal{I}(\theta_{g,j})} L_g(\theta)}{L_{max}} \quad (5)$$

All the features are normalized by the length L_{max} of the middle finger in order to scale them within range $[0, 1]$ and to account for the fact that hands of different people have different sizes. Note that there can be up to $G \times 5$ features, though their actual number is smaller since not all the fingers are raised in each gesture. The distance features are collected into feature vector $\mathbf{F}^l$.

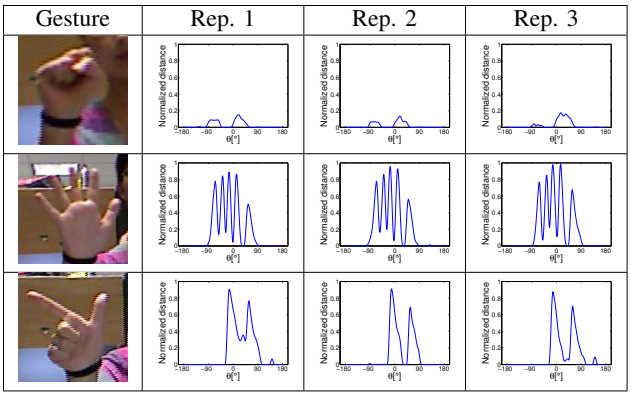


Fig. 3: Examples of aligned distance histogram $L_g(\theta)$ for 3 sample frames corresponding to different gestures.

B. Elevation features

The construction of the elevation features is analogous to the one employed for the distance features of Section IV-A. The process starts by building an histogram representing the distance of each sample in $\mathcal{F}$ from the palm plane π, namely, for each sample $\mathbf{X}_i$ in $\mathcal{F}$ its distance from plane π is computed:

$$e_{\mathbf{X}_i} = \text{sgn}\left((\mathbf{X}_i - \mathbf{X}_i^\pi) \cdot \mathbf{i}_y^\pi\right) |\mathbf{X}_i - \mathbf{X}_i^\pi| \quad (6)$$

where $\mathbf{X}_i^\pi$ is the projection of $\mathbf{X}_i$ on π. The sign of $e_{\mathbf{X}_i}$ accounts for the fact that $\mathbf{X}_i$ can belong to any of the two semi-spaces defined by π, that is, $\mathbf{X}_i$ can either be on the front or behind π.

Now, as for the distance features, for each angular sector corresponding to a θ_q, the point with greatest absolute distance from the plane is selected, thus producing an histogram:

$$E(\theta_q) = \begin{cases} \max\limits_{\mathcal{I}(\theta_q)} e_{\mathbf{X}_i} & \text{if } \left|\max\limits_{\mathcal{I}(\theta_q)} e_{\mathbf{X}_i}\right| > \left|\min\limits_{\mathcal{I}(\theta_q)} e_{\mathbf{X}_i}\right| \\ \min\limits_{\mathcal{I}(\theta_q)} e_{\mathbf{X}_i} & \text{otherwise} \end{cases} \quad (7)$$

Note that $E(\theta_q)$ uses the same regions computed in Section IV-A. The histogram $E(\theta)$ corresponding to the performed gesture is then aligned to the various reference gestures in G using the alignment information already computed in Section IV-A. Let $E^g(\theta)$ be the histogram $E(\theta)$ aligned with the template gesture g. The elevation features are then

computed according to:

$$f^e_{g,j} = \begin{cases} \frac{1}{L_{max}} \max\limits_{\mathcal{I}(\theta_{g,j})} E^g(\theta) & \text{if } \left|\max\limits_{\mathcal{I}(\theta_{g,j})} E^g(\theta)\right| > \\ & \left|\min\limits_{\mathcal{I}(\theta_{g,j})} E^g(\theta)\right| \\ \frac{1}{L_{max}} \min\limits_{\mathcal{I}(\theta_{g,j})} E^g(\theta) & \text{otherwise} \end{cases} \quad (8)$$

Note that the vector $\mathbf{F}^e$ of the elevation features has the same structure and number of elements of the vector $\mathbf{F}^l$ of the distance features.

C. Contour histogram similarity features

This feature set is based on the similarity between distance histograms of subsection IV-A. For each considered gesture, a reference acquisition is selected and the corresponding distance histogram is computed with the approach of Eq. 2, thus obtaining a set of reference histograms $L^r_g(\theta)$, where g is the considered gesture. The distance histogram of the acquired gesture $L(\theta_q)$ is also computed and the maximum of the correlation between the current histogram $L(\theta_q)$ and a shifted version of the reference histogram $L^r_g(\theta)$ is selected:

$$R_g = \max_\Delta \left[\rho\left(L(\theta), L^r_g(\theta+\Delta)\right), \rho\left(L(-\theta), L^r_g(\theta+\Delta)\right)\right] \quad (9)$$

where $g = 1, ..., G$. Note how the flipped histogram is also considered for the reasons already outlined in section IV-A. The computation is performed for each of the candidate gesture, thus obtaining a set $\mathbf{F}^\rho$ containing a different feature value f^ρ_g for each of them. As expected, ideally the correlation with the correct gesture should have a larger value than the other features.

An important aspect of this feature extraction method is the employed similarity metric, which strongly affects the results. A reasonable high accuracy can be obtained through the *zero-mean normalized cross-correlation* (ZNCC) between the histograms, since this measure is less affected by the different sizes of different hands. On the other side, this measure is sometimes not able to discriminate two histograms with a similar outline corresponding to different fingers, e.g. an histogram representing a gesture with a raised index only, may have an high correlation value with an histogram representing a gesture with a raised little only. This ambiguity can be often removed by using alternative similarity measurement values in place of or together with the ZNCC. A good alternative is the *sum of squared differences* (SSD) between the two histograms. The features obtained with this measure will be denoted with $\mathbf{F}^{SSD}$.

D. Curvature features

The third proposed descriptor is based on the curvature of the hand shape edges. Since depth data coming from real-time depth cameras are usually rather noisy, a reasonable assumption is to avoid differential operators for curvature description, relying instead on integral invariants [32], [33].

The curvature feature extractor algorithm takes as input the hand edge points $\mathcal{H}_e$ and the binary mask $B(u, v)$. Let $\mathcal{H}_c = \partial\mathcal{H}_e$ be the boundary of $\mathcal{H}_e$, namely the subset of all the points $\mathbf{X}_i \in \mathcal{H}_e$ belonging to the hand contour only. Consider a set of S circular masks $M_s(\mathbf{X}_i),\ s = 1, .., S$ with radius r_s centered on each edge sample $\mathbf{X}_i \in \mathcal{H}_c$ (reasonable numbers are 25 masks with r_s varying from $0.5cm$ to $5cm$).

Let $V(\mathbf{X}_i, s)$ denote the curvature in $\mathbf{X}_i$, expressed as the ratio of the number of samples of $\mathcal{H}_e$ falling in the mask $M_s(\mathbf{X}_i)$ over $M_s(\mathbf{X}_i)$ size, namely:

$$V(\mathbf{X}_i, s) = \frac{\sum_{\mathbf{X}_i \in M_s(\mathbf{X}_i)} B(\mathbf{X}_i)}{|M_s(\mathbf{X}_i)|} \quad (10)$$

where $|M_s(\mathbf{X}_i)|$ denotes the cardinality of $M_s(\mathbf{X}_i)$ and $B(\mathbf{X}_i) = B(u_j, v_j)$, with (u_j, v_j) be the 2D coordinates corresponding to $\mathbf{X}_i$. Note that $V(\mathbf{X}_i, s)$ is computed for each sample $\mathbf{X}_i \in \mathcal{H}_c$. The value s corresponds to the scale level at which feature extraction is performed. Differently from [33] and other approaches, the radius r_s is defined in metrical units and is then converted to the corresponding pixel size on the basis of the distance between the camera and the hand. In this way the descriptor is invariant with respect to the distance between the hand and the camera.

For faster processing, the circular masks can be replaced with simpler square masks and then integral images can be used for fast computation of the samples in the mask. This approach, even if not perfectly rotation invariant, proved to be significantly faster and the performance loss is negligible.

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. The $[0, 1]$ interval is quantized into N bins of equal size $b_1, .., b_N$. The set $\mathcal{V}_{b,s}$ of the finger edge points $\mathbf{X}_i \in \mathcal{H}_c$ with the corresponding value of $V(\mathbf{X}_i, s)$ falling to bin b for the mask s is expressed as:

$$\mathcal{V}_{b,s} = \{\mathbf{X}_i | \frac{(b-1)}{B} < V(\mathbf{X}_i, s) \leq \frac{b}{B}\} \quad (11)$$

For each radius value s and for each bin b the chosen curvature feature, denoted by $f^c_{b,s}$, is the cardinality of the set $V(\mathbf{X}_i, s)$ normalized by the contour length $|\mathcal{H}_c|$:

$$f^c_{b,s} = \frac{|\mathcal{V}_{b,s}|}{|\mathcal{H}_c|} \quad (12)$$

Note that, because of the normalization, the curvature feature $f^c_{b,s}$ only takes values in $[0, 1]$, which is the same interval shared by both the distances and elevations feature. Finally, all the curvature features $f^c_{b,s}$ are collected within a feature vector $\mathbf{F}^c$ with $B \times S$ entries, ordered by increasing values of indexes $s = 1, 2, ...S$ and $b = 1, 2, ..., N$. By resizing $\mathbf{F}^c$ into a matrix with S rows and N columns, and by considering each $f^c_{b,s}$ as the value of the pixel with

coordinates (b, s) in a grayscale image, it is possible to graphically visualize the overall curvature descriptor $\mathbf{F}^c$ as exemplified in Fig. 4.

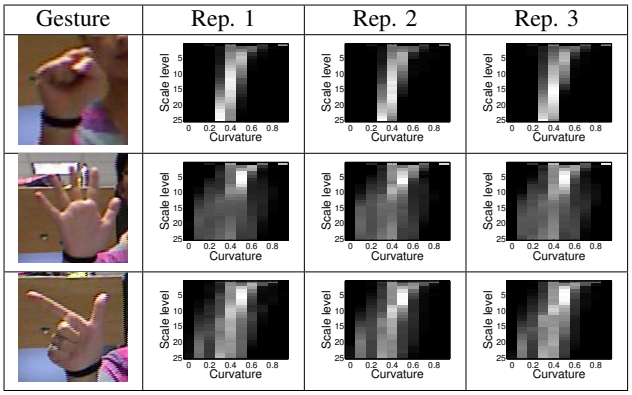


Fig. 4: Examples of curvature descriptors for 3 sample frames from different gestures.

E. Palm area features

Palm area set of features describes the displacement of the samples in the palm region $\mathcal{P}$. Note that $\mathcal{P}$ corresponds to the palm area, but it may also include finger samples if some fingers are folded over the palm. The idea is to subdivide the palm region into six different areas, defined over the plane π, as shown in Fig. 5. The circle or ellipse defining the palm area is divided into two parts: the lower half is used as a reference for the palm position, and a 3D plane π_p is firstly fitted to this region. The upper half is divided into 5 regions $\mathcal{A}_j, j = 1, .., 5$ roughly corresponding to the regions close to the different fingers as shown in Fig. 5, that is, each region corresponds to the area that is affected by the position of a finger. Note how the feature values account for the deformation the palm shape undergoes in the corresponding area when the related finger is folded or is moved. In particular, it is worth noting that the samples corresponding to the fingers folded over the palm are associated to $\mathcal{P}$ and are not captured by distance or elevation features, but they are used for the computation of palm area features. The areas positions on the plane strictly depend on the parameters defining the palm area (i.e. the center $\mathbf{C}^\pi$ and the radius R of the circle or the two axes of the ellipse), on the fingers width and on the direction $\mathbf{i}_x^\pi$ corresponding to $\theta = 0$. The alignment of the θ directions can be computed as in Section IV-A, thus obtaining an alignment for each gesture template. The areas aligned with the template of each gesture will be denoted with $\mathcal{A}_j^g$, where g indicates the corresponding gesture. In this way the set of points $\mathbf{X}_i$ in $\mathcal{P}$ associated to each of the regions $\mathcal{A}_j^g$ is computed. Then, each area $\mathcal{A}_j^g$ is considered and the distance between each sample $\mathbf{X}_i$ in $\mathcal{A}_j^g$ and π_p is computed. The average of the distances of the samples of the area A_j^g :

$$f_{g,j}^a = \frac{\sum_{\mathbf{X}_i \in \mathcal{A}_j^g} \|\mathbf{X}_i - \mathbf{X}_i^\pi\|}{|A_j^g|} \tag{13}$$

is taken as the feature corresponding to the area A_j^g. All the area features are collected within vector $\mathbf{F}^a$, made by $G \times 5$ area features, one for each finger in each possible gesture. The entries of $\mathbf{F}^a$ are finally scaled in order to assume values within range $[0, 1]$, as the other feature vectors.

Fig. 5: Regions corresponding to the various area features shown over a sample gesture.

F. Convex hull features

The convex hull of the 2D hand shape in the depth map is another source of information that can be exploited for the construction of descriptors of the hand pose [11]. Given the set of points on the hand mask $\mathcal{B}$, the corresponding convex hull is computed. Due to the noise of the sensors, the set of vertexes may contains some artifacts that should be removed before exploiting it for gesture recognition:

1) The hand shape can contain some small holes due to noise that must be removed
2) There can be close vertexes due to the irregular shape of the acquired hand contour (see Fig. 6a). The length of the convex hull edges is computed and in case there were edges shorter than a pre-defined threshold they must be removed and the corresponding vertexes collapsed into a single point (Fig. 6b).
3) Also the presence of angles close to $180°$ (i.e. two subsequent edges are almost on the same line) is an hint of extra edges due to acquisition artifacts. In this case the vertex is removed as well and the two edges are combined into a single one, as shown in Fig. 6c.

In this way a simplified convex hull $C_{hull}(\mathcal{B})$ with vertexes P_{chull} can be built.

1) Convex hull vertexes: A first possible feature is the number of vertexes of the convex hull $F^{pch} = |P_{chull}|$. This value is an hint of the hand pose and in particular of the number of raised fingers. Note how, ideally, the desired convex hull presents a vertex for each fingertip and a few other vertexes delimiting the palm area. These vertexes, together with the distances histograms computed in Section IV-A, may also aid the fingertips position estimation: assuming that, for each histogram value the related 3D point is known, since the histogram local maxima correspond

a) b) c)

Fig. 6: Computation of the vertexes of the convex hull: a) Convex hull of $B(u, v)$; b) Convex hull after the removal of short edges; c) Convex hull after the removal of short edges and of wide angles.

to the hand shape boundary (which contains the fingertips as well), by pairing the convex hull vertexes with them it is possible to detect which local maxima are likely to be associated to fingertips and which ones, instead, are probably due to noise or associated to the palm edges.

2) Convex hull area ratio: Another good feature is the ratio between the area of the hand shape and the area of the convex hull enclosing it, that is:

$$F^{ch} = \frac{|\mathcal{B}|}{area(C_{hull}(\mathcal{B}))} \tag{14}$$

3) Convex hull perimeter ratio: The ratio between the perimeter of the hand shape and the perimeter of the convex hull:

$$F^{Rp} = \frac{perimeter(\mathcal{B})}{perimeter(C_{hull}(\mathcal{B}))} \tag{15}$$

is another useful clue. Gestures with bended fingers typically correspond to perimeter ratios close to 1 while when several fingers are pointing out of the hand this ratio is usually smaller. Fig. 7 shows a couple of examples; note how in the first case the two perimeters are quite different, while in the single finger gesture of the second row the two perimeters are more similar.

G. Connected components features

One of the other relevant clues that can be extracted from the comparison between the convex hull and the hand shape is the shape of the regions within the convex hull region but not belonging to the hand. These typically correspond to the empty regions between the fingers and are a good indicator of the fingers arrangement. Let $S = C_{hull}(\mathcal{B}) - \mathcal{B}$ be the difference between the convex hull and the hand shape. An example of the region S is shown in Fig. 8a. The region S is typically made of a set of connected components. The various connected components S_i are extracted and then the ones that are smaller than a threshold T_{cc} are discarded in order to avoid considering small components due to noise (e.g. the region on the right in the second row of Fig. 8). In this way the set $\mathcal{S} = \{S_i : S_i > T_{cc}\}$ is built as shown in Fig. 8b.

A first feature that can be extracted from this process is the number of connected components $N_{cc} = |\mathcal{S}|$ bigger

a) b)

Fig. 7: Perimeter of the hand contour and of the convex hull; a): convex hull perimeter. b) hand perimeter.

a) b)

Fig. 8: Area of the connected components. a) Connected components in set S; b) Connected components in set $\mathcal{S}$ highlighted in green.

than the threshold. Another feature set is, instead, given by the ratio of the areas of the various connected components with the convex hull area, that is:

$$f_i^{cc} = \frac{area(S_i | S_i \in \mathcal{S})}{area(C_{hull}(\mathcal{B})))} \tag{16}$$

where the areas are sorted according to the angle of their centroid with the axes computed by the PCA (i.e. from the thumb to the little).

V. Gesture classification with Support Vector Machines

The third and last step of the Section I pipeline consists in applying an appropriate machine learning technique to classify the features extracted in Section IV, in order to

recognize the performed gestures. Approaches based on Support Vector Machines, Randomized Decision Forests, Neural Networks and many others have been proposed in literature. Presenting the various machine learning methods is beyond the scopes of this chapter that focuses on feature extraction, although the classification with Support Vector Machines (SVM) is here briefly recalled only for clarity sake and used to compare the efficiency of the various features.

The feature extraction approaches of Section IV provide different feature vectors describing relevant properties of the hand samples. Let $\mathbf{F}$ denote a suitable feature vector describing the performed gesture, e.g. $\mathbf{F} = \mathbf{F}^l$ for distance features. The gesture recognition problem consists in classifying the vectors $\mathbf{F}$ into G different classes corresponding to the considered gestures. A multi-class SVM classifier based on the *one-against-one* approach can be used, and corresponds to a set of $G(G-1)/2$ binary SVM classifiers used to test each class (i.e., gesture) against each other. Each classification output is chosen as a *vote* for a certain gesture and the gesture with the maximum number of votes is the result of the recognition process. Different kernels can be used for the SVM, but the most common choice is the non-linear Gaussian Radial Basis Function (RBF) kernel. In order to set the classifier parameters, a training set containing data from N users is assumed to be available. A naive grid search with cross-validation approach is viable, although not efficient: the space of parameters (C, γ) of the RBF kernel is subdivided with a regular grid and for each couple of parameters the training set is divided into two parts, one containing $N-1$ users for training and the other with the remaining user for validation. The performances are evaluated and the procedure is repeated by changing each time the user used for the validation. The couple of parameters that give the best accuracy on average is finally selected. The SVM can then be trained on all the N users of the training set with the optimal parameters.

This approach can also be used to perform the recognition with multiple feature descriptors together. The simplest solution is to concatenate into vector $\mathbf{F}$ the different feature vectors of the descriptors that are going to be used. E.g. to combine distance, elevation and curvature descriptors a combined feature vector $\mathbf{F} = [\mathbf{F}^l, \mathbf{F}^e, \mathbf{F}^c]$ can be fed to the SVM classifier. More refined combination schemes can be exploited, e.g. as in [34].

VI. Experimental results

This chapter analyzes the classification performances of the various feature descriptor described in Section IV. The experiments were performed on a gesture dataset acquired in the Multimedia Technology and Telecommunications Lab of the University of Padova. Such dataset is a sub-set of the American Manual Alphabet, and contains 10 repetitions of 12 different gestures performed by 14 different people. A representative picture for each gesture is shown in Fig. 9 while the complete dataset is made available at the url http://lttm.dei.unipd.it/paper_data/gesture, and provides both the RGB image and the depth map for all the frames.

For each gesture, one of the repetitions in the training set was used for the computation of the reference histogram of Eq. (3), required for the extraction of distances, elevations, correlations and contour histogram similarity features.

The classification model was computed through the SVM implementation provided by the OpenCV library exploiting a variation, tailored for the particular data nature, of the grid-search approach described in Section V. Differently from the classic grid-search method, which performs random subsamplings of the feature vectors for splitting the dataset in a train and a test set, or use a *leave-one-out* approach, the method employed for the experiments relies on an extension of the *leave-one-out* which may be named as *leave-one person-out*. Namely, all the feature vectors referred to a specific person belong either to the test or the train set of the current dataset split. Fig. 10 exemplifies the *leave-one person-out* approach employed for the experiments. The interested reader may find a more detailed discussion on this classification method in [30].

Table I shows the results obtained from the considered dataset by just using each single feature alone. Distance features (**D**) alone provide an accuracy of about 68%. Note that distance descriptors are very good in capturing the fact that the various fingers are folded over the palm or raised, an important element in the recognition of many gestures. Elevation features (**E**) have lower performance as well (46.0%); this is due both to the fact that in most gestures in the dataset the fingers lay very close to the palm plane, and to the varying accuracy in the plane fitting described in Section III. Contour histogram similarity features (**R**) are slightly less performing, since histogram correlations are just a starting point for the distance features extraction and, moreover, due to the zero-mean cross correlation algorithm exploited, they are not always able to discriminate gestures with the same number of fingers. Sum of squared differences (**SSD**) provide, instead, poor performances (37.8%). Such features are anyway helpful when combined with other features as showed in Table II. The curvature-based classifier (**C**) allows to obtain the best performance (89.5%), thanks also to the characteristic concavities and convexities of each gesture profile. It is important to note that curvatures do not rely on the computation of the hand orientation or on the positions of the centroid and palm plane. For this reason, curvature-based classifiers are more performing in complex configurations where the estimation of the hand orientation is not always highly accurate. Area based features (**A**) allow to obtain, again, a relatively low accuracy (45.4%), as they are affected by problems similar to the elevations: the different quality in the plane fitting and the fact that in most gestures many fingers lay on the palm plane. Convex hull area (**CA**) and perimeter ratios (**CP**) are the least performing features, probably because their vectors are, single scalar numbers, and hence not sufficient to separate a relatively high number of classes.

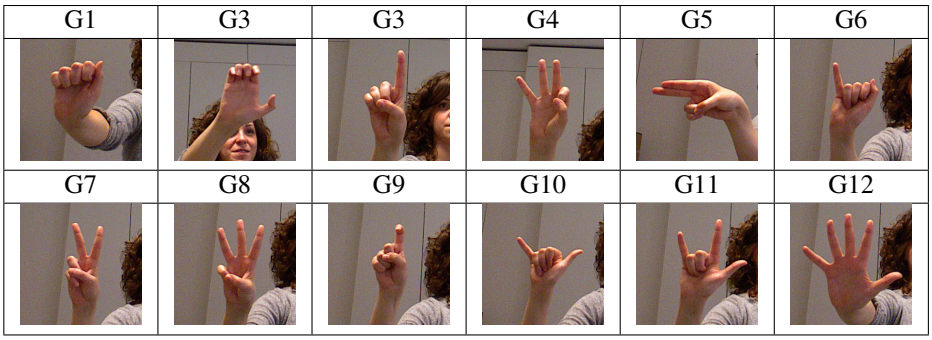


Fig. 9: Gestures from the American Manual Alphabet contained in the experimental dataset.

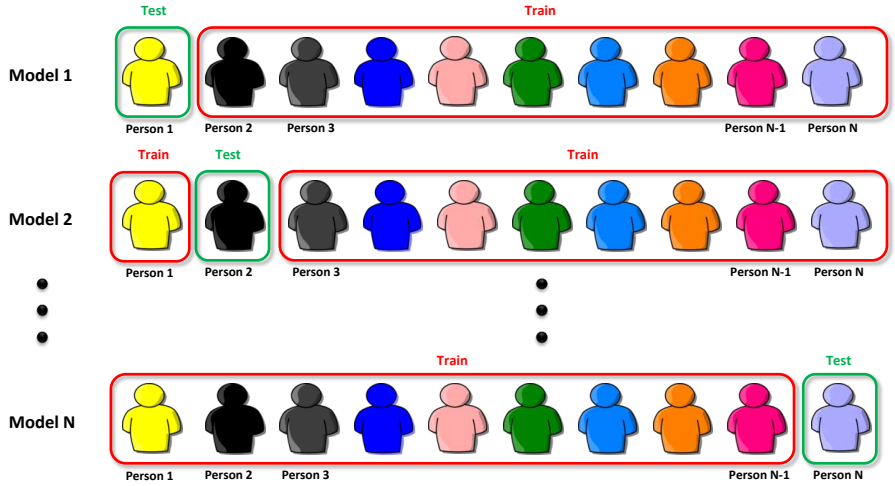


Fig. 10: Exemplification of *leave-one person-out* method.

Finally, the convex hull connected component area (**CC**) features are ones of the most performing, although their number is sensibly low respect to the curvatures (6 vs 240 features).

Type of features	Accuracy
Distances (D)	68.4 %
Elevations (E)	46.0 %
Correlations (R)	60.4 %
Sum of squared distances (SSD)	37.8 %
Curvatures (C)	89.5 %
Areas (A)	45.4 %
Convex hull area ratio (CA)	29.6 %
Convex hull perimeter ratio (CP)	37.0 %
Convex hull conn. comp. area ratio (CC)	72.0 %

TABLE I: Performance of single features extracted from the considered dataset.

A noteworthy characteristic of the described features is their complementarity, that is, although many of them are not so performing alone, when joined together they are able to increase the overall classification accuracy as one feature type is able to capture properties not detectable by another feature type and viceversa. Table II shows some examples of the performances that can be obtained combining multiple features. In particular, the distance only classifier is able to recognize some of the gestures that curvatures only can not handle, and their joint usage raises slightly the classification performance (91.4%). The combination of correlation and sum of squared differences features, instead, leads to a dramatic improvement in accuracy (78.3% vs 60.4% and 37.8% respectively). Even by combining the convex hull area and perimeter ratios it is possible a significant overall improvement (51.2% vs 29.6% and 37.0% respectively).

The second part of Table II shows that by combining 3 feature types a modest overall accuracy improvement can be obtained. This is due to the fact that the accuracy values are already quite high, and not always the added feature carry a sufficient amount of novel information to solve the most critical classification ambiguities. The best accuracy has been obtained by combining 4 feature types, i.e., distances, elevation, area based and curvature features. Finally, it is important to be aware that the addition of an increasing number of feature types may not lead to an overall improvement in accuracy and, sometimes, may also lead to overfitting with detrimental effects on the classifier

performance.

Type of features	Accuracy
D + C	91.4 %
R + SSD	78.3 %
CA + CP	51.2 %
D + E + C	93.6 %
D + A + C	92.0 %
DA + CP + CC	73.2 %
D + E + A + C	93.5 %
R + SSD + CA + CP + CC	86.8 %

TABLE II: Performance of the features combination.

VII. Conclusions

In this chapter effective solutions for the exploitation of depth data in static hand gesture recognition have been presented. The hand recognition and extraction step, a very challenging task with color information, can be effectively solved with simple approaches if depth data is available. Furthermore depth can be combined with the color data for an even more reliable recognition. Several depth-based feature descriptors have been presented. Depth-based descriptors are typically based on measures in the 3D space and are more robust to issues like the position of the hand, its orientation, lighting and many others than color-based descriptors. Furthermore they are able to better capture the pose of the hand and of the various fingers, making the gesture recognition task easier. A comparison of the performances of the various features have been presented in Section VI. Some features, most notably the curvatures, but also distance and correlation features, have better performances and allow alone to obtain a reliable gesture recognition. However, notice how the different features capture relevant complementary properties of the hand gestures. For this reason the combination of multiple features allows to obtain better performance than each feature alone.

References

[1] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," *Commun. ACM*, vol. 54, no. 2, pp. 60–71, Feb. 2011.

[2] P. Garg, N. Aggarwal, and S. Sofat, "Vision based hand gesture recognition," *World Academy of Science, Engineering and Technology*, vol. 49, no. 1, pp. 972–977, 2009.

[3] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *Cybernetics, IEEE Transactions on*, vol. 43, no. 5, pp. 1318–1334, Oct 2013.

[4] X. Zabulis, H. Baltzakis, and A. Argyros, "Vision-based hand gesture recognition for human computer interaction," in *The Universal Access Handbook*, ser. Human Factors and Ergonomics. Lawrence Erlbaum Associates, Inc. (LEA), June 2009, ch. 34, pp. 34.1 – 34.30.

[5] Z. Mo and U. Neumann, "Real-time hand pose recognition using low-resolution depth images," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 2006, pp. 1499–1505.

[6] 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 the 19th ACM international conference on Multimedia*, ser. MM '11. New York, NY, USA: ACM, 2011, pp. 1093–1096. [Online]. Available: http://doi.acm.org/10.1145/2072298.2071946

[7] 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.

[8] Z. Ren, J. Meng, and J. Yuan, "Depth camera based hand gesture recognition and its applications in human-computer-interaction," in *Proc. of Int. conference on Information, Communications and Signal Processing (ICICS)*, dec. 2011, pp. 1 –5.

[9] Y. Wen, C. Hu, G. Yu, and C. Wang, "A robust method of detecting hand gestures using depth sensors," in *Haptic Audio Visual Environments and Games (HAVE), 2012 IEEE International Workshop on*, 2012, pp. 72–77.

[10] Y. Li, "Hand gesture recognition using kinect," in *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*, June 2012, pp. 196 –199.

[11] F. Pedersoli, N. Adami, S. Benini, and R. Leonardi, "Xkin - extendable hand pose and gesture recognition library for kinect," in *In: Proceedings of ACM Conference on Multimedia 2012 - Open Source Competition*, Nara, Japan, Oct. 2012.

[12] F. Pedersoli, S. Benini, N. Adami, and R. Leonardi, "Xkin: an open source framework for hand pose and gesture recognition using kinect," *The Visual Computer*, pp. 1–16, 2014.

[13] P. Suryanarayan, A. Subramanian, and D. Mandalapu, "Dynamic hand pose recognition using depth data," in *Proc. of Int. Conference on Pattern Recognition (ICPR)*, aug. 2010, pp. 3105 –3108.

[14] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, "Robust 3d action recognition with random occupancy patterns," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2012.

[15] N. Pugeault and R. Bowden, "Spelling it out: Real-time asl fingerspelling recognition," in *Proceedings of the 1st IEEE Workshop on Consumer Depth Cameras for Computer Vision*, 2011, pp. 1114–1119.

[16] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Hand pose estimation and hand shape classification using multi-layered randomized decision forests," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2012, pp. 852–863.

[17] K. Biswas and S. Basu, "Gesture recognition using microsoft kinect," in *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, Dec. 2011, pp. 100 –103.

[18] P. Doliotis, A. Stefan, C. McMurrough, D. Eckhard, and V. Athitsos, "Comparing gesture recognition accuracy using color and depth information," in *Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments(PETRA'11)*, 2011, pp. 20:1–20:7.

[19] T. Wan, Y. Wang, and J. Li, "Hand gesture recognition system using depth data," in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, April 2012, pp. 1063 –1066.

[20] C. Sun, T. Zhang, B.-K. Bao, C. Xu, and T. Mei, "Discriminative exemplar coding for sign language recognition with kinect," *Cybernetics, IEEE Transactions on*, vol. 43, no. 5, pp. 1418–1428, Oct 2013.

[21] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect," in *Proceedings of the 22nd British Machine Vision Conference (BMVC 2011)*, aug. 2011.

[22] 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 the European Conference on Computer Vision (ECCV)*, Firenze, October 2012.

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

[24] D. Herrera, J. Kannala, and J. Heikkilä, "Joint depth and color camera calibration with distortion correction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 10, pp. 2058–2064, 2012.

[25] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

[26] X. Liu and K. Fujimura, "Hand gesture recognition using depth data," in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, May 2004, pp. 529–534.

[27] E. Kollorz, J. Penne, J. Hornegger, and A. Barke, "Gesture recognition with a time-of-flight camera," *Int. J. Intell. Syst. Technol. Appl.*, vol. 5, no. 3/4, pp. 334–343, Nov. 2008. [Online]. Available: http://dx.doi.org/10.1504/IJISTA.2008.021296

[28] M. Van den Bergh and L. Van Gool, "Combining rgb and tof cameras for real-time 3d hand gesture interaction," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, Jan 2011, pp. 66–72.

[29] F. Dominio, M. Donadeo, G. Marin, P. Zanuttigh, and G. M. Cortelazzo, "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*. ACM, 2013, pp. 9–16.

[30] F. Dominio, M. Donadeo, and P. Zanuttigh, "Combining multiple depth-based descriptors for hand gesture recognition," *Pattern Recognition Letters*, 2013.

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

[32] S. Manay, D. Cremers, B.-W. Hong, A. Yezzi, and S. Soatto, "Integral invariants for shape matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1602 –1618, October 2006.

[33] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. Lopez, and J. Soares, "Leafsnap: A computer vision system for automatic plant species identification," in *Proc. of the European Conference on Computer Vision (ECCV)*, October 2012.

[34] L. Nanni, A. Lumini, F. Dominio, M. Donadeo, and P. Zanuttigh, "Ensemble to improve gesture recognition," *International Journal of Automated Identification Technology (to appear)*, 2014.