Computer Science • 19(2) 2018 https://doi.org/10.7494/csci.2018.19.2.2718

Dawid Warchoł

HAND POSTURE RECOGNITION USING MODIFIED ENSEMBLE OF SHAPE FUNCTIONS AND GLOBAL RADIUS-BASED SURFACE DESCRIPTOR

Abstract *This paper presents an approach to the recognition of static hand gestures based on data acquired from 3D cameras and point cloud descriptors: Ensemble of Shape Functions and Global Radius-based Surface Descriptor. We describe a recognition algorithm consisting of hand segmentation, noise removal and downsampling of point clouds, dividing point cloud bounding boxes to cells, feature extraction and normalization, and gesture classification. Modifications to the descriptors are proposed in order to increase the hand posture recognition rates while decreasing the quantity of used features as well as the computational cost of the algorithm. Experiments performed on four challenging datasets using cross-validation tests prove the usefulness of our approach.*

Keywords hand posture recognition, point cloud, descriptor, 3D camera

Citation Computer Science 19(2) 2018: 115–137

1. Introduction

Nowadays, more and more emphasis is being placed on the communication of people with computers and machines in a natural way; e.g., using hand gestures. The hand is the most functional part of the human body by which people communicate when verbal communication is not possible. It is often use as an aid to verbal communication. There is, therefore, a need to develop methods of automatic gesture recognition in applications such as human-robot-interaction, controlling smart home devices, manipulating virtual objects, controlling television sets and other multimedia hardware or computer games, and interpreting sign language statements; e.g., for automatic translation to a spoken or written language.

The last of the listed applications is very important in the context of communication with deaf and hearing-impaired people in typical situations in which a translator is usually required. Applying an automatic sign language communicator could, therefore, make it easier for the deaf community to function in society.

Recognition of static hand gestures is a challenging issue. It can be assumed that the recognition of static hand postures is even more challenging than recognizing dynamic gestures (with motion) because, contrary to this, algorithms of static gesture recognition do not use information of moving hand trajectory (which is very useful in distinguishing between the gesture classes). The classification is, therefore, only based on hand shape features. Moreover, some hand shapes representing different gestures are very similar to each other; e.g., the letters M and N as well as S and T of the American Finger Alphabet.

In our work, we use data acquired from 3D cameras in the form of point clouds. Three-dimensional data (or depth data) has several advantages over standard color or gray images. One of these is the possibility of hand segmentation based on depth that can be more accurate than methods that rely solely on color information. Another advantage is the representation of objects by features based on the surface curvature in addition to the standard 2D shape features typical for color or gray images. Moreover, some models of 3D cameras (using time-of-flight technology) work correctly under poor lighting conditions or even in absolute dark, which can be another advantage.

This paper presents an approach to static hand posture recognition based on 3D data and two point cloud descriptors: Ensemble of Shape Functions (ESF) and the Global Radius-based Surface Descriptor (GRSD). These descriptors are available in the Point Cloud Library (PCL) for the C++ language, and they are used to describe the geometry of objects represented by 3D data in order to distinguish classes of objects. They are designed to recognize objects like tools, toys, fruits, etc. We propose to use them to describe hand postures and utilize this information in recognition methods. Modifications of these descriptors are proposed to better describe hand geometry and, therefore, improve hand posture recognition rates while decreasing the quantity of used features and the computational cost of the algorithm.

The proposed recognition method can be used with 3D depth cameras based on structural light or time-of-flight technologies. Some examples of such cameras are Kinect, Kinect 2, and MESA SR4000.

The remaining part of the paper is organized as follows. Section 2 contains an overview of related publications. Our proposed hand posture recognition method is presented in Section 3. Section 4 discusses two 3D point cloud descriptors and their proposed modifications. In Section 5, we describe the performed experiments and present their results. The conclusions and future works are included in Section 6.

2. Related work

In recent years, there has been increasing interest in the subject of object recognition using data from 3D cameras. Currently, methods of static hand posture recognition exist that use 3D data in the form of a point cloud or depth map. In some of these, the 3D information is used as an addition to color, gray, or skeletal data; e.g., [21, 33, 37].

The recognition method proposed in [24] is based on Gabor filters and a random forest classifier. In [22], Gabor filters are also used along with a multi-class SVM (Support Vector Machine) classifier with the all-versus-all method. Wang et al. [35] described an algorithm that uses a shape descriptor developed by them based on the polar representation of hand images obtained using depth data. The Depth-projection-based Bag of Contour Fragments, the recognition algorithm presented by Feng et al. [6], is used to extract the hand region and describe its shape.

Molina et al. [20] proposed an algorithm of static and dynamic hand-gesture recognition based on the geodesic description of the 3D hand surface and k-nearest neighbor (KNN) classification. In [40], the Histogram of 3D Facets is described. It uses so-called 3D facets as local supporting surfaces related to each point of the cloud. The classification is performed by an SVM with a linear kernel function. In [11], hand postures are classified using multi-layered random forests. Another approach using 3D data and Hu moments is presented in [39]. Li [12] utilized information about the angles between extended fingers and a two-stage classification that eliminated gestures with different numbers of detected fingers in the first stage. Plouffe et al. [23] proposed a method of comparing data sequences (Dynamic Time Warping – DTW) to static hand posture recognition, although this technique is usually used for the classification of dynamic gestures (e.g., [28]). In [25], the authors developed a system of gesture recognition based on the convex shape decomposition method that uses Morse functions.

The algorithms presented in [5, 10] utilize 3D data from Kinect and synthetically generated hand models (3D [10] or 2D [5]) with separated particular hand parts. These models are used by the random forest classifier, which uses them to categorize each pixel, the dividing hand shape of a classified gesture into similar fragments. Then, a hierarchical mode-seeking method is used to designate hand joints. Finally, the classifier (SVM [10] or random forests [5]) makes a selection of the class using features based on the angular relationships of the joints.

Dominio et al. [4] proposed an SVM classification and four combined depth features: (i) the distances of the fingertips from the palm center; (ii) the distances of the fingertips from a plane fitted on the palm samples; (iii) the curvature of the contour of the hand region; and (iv) the shape of the palm region. In [14], depth features from [4] are used along with the features obtained from the skeletal data acquired by a Leap Motion camera. Ren et al. [26] described a method of measuring the differences between hand shapes on the basis of Finger-Earth Mover's Distance (FEMD) – a modified version of the Earth Mover's Distance (EMD) metric (modified to work better with hand shapes).

The recognition method from our work does not use any other information than depth (e.g., to detect skin color in order to separate the hand region from the background [4]) and does not require the user to wear particular clothing (e.g., long sleeves [13, 32]), use additional objects such as gloves (e.g., [8, 31]) or wrist bands (e.g., [26,32]), nor performing gestures on a simple uniform background (e.g., [7,32]).

3. Hand posture recognition method

Our recognition method consists of hand segmentation, noise removal and downsampling of the point cloud, dividing the point cloud bounding box to cells, feature extraction and normalization of gesture classification.

Two different segmentation methods were used depending on the used dataset (see Subsection 5). The first is the method of the "gesticulation area" that works well even with cameras featuring relatively inaccurate depth data (e.g., MESA SR4000). It requires the user to perform gestures in the fixed 3D area and separates the hand from the background based on the area boundaries and information about user's hand size (which should be measured by the user or the algorithm). This is a simple method that can roughly separate the forearm region from the hand if the requirements are met. It is, however, sufficient in the hand posture recognition that was shown in [9], where the method is described in detail.

The second segmentation method (called "precise segmentation") is much more complex, effective, and does not require gestures to be performed in a fixed area nor measure the user's hand. However, it requires 3D data with relatively accurate depth (e.g., from the Kinect or Kinect 2 camera) and occurrence of at least a small portion of the forearm within the point cloud. These requirements are met only in the case of two out of the four datasets used in the experiments, which is the reason for adopting two different segmentation methods. The algorithm is based on finding the hand center point using a Gaussian blur with a relatively large kernel on a binarized hand image and fitting the largest possible circle in the palm region. This is described in detail in [36].

Point clouds obtained from 3D cameras (especially time-of-flight types) often include noise in the form of isolated points. Therefore, we filter each cloud by removing the points that have a small number of neighboring points within the given radius.

Another problem is that point clouds are often too redundantly dense for posture recognition. There is no need for such a large number of points to calculate representative descriptor features. For this reason, the cloud is downsampled, which means reducing the number of points and, therefore, speeding up the calculation of the descriptors. This operation is performed by the so-called voxel grid filter, which creates a 3D voxel grid over the input point cloud data. Each point situated within each voxel (3D cuboid) is approximated by its centroid. Voxel dimensions $V_x \times V_y \times V_z$ are the parameters of the filter. In the experiments performed in this work, cubic voxels are used. We experimentally found that the dimensions of $V_x = V_y = V_z = 4.5$ mm are sufficient to calculate the representative features of the descriptors. Larger dimensions cause worse recognition rates and their larger spread, especially if the value is greater than 7 mm.

In our method, the bounding box must be generated before calculating the descriptors. This can be defined as the smallest possible cuboid that entirely embraces the point cloud. Its walls are determined by the points situated furthest towards each main direction of the coordinate system. The front and back walls are parallel to the camera lens. The bounding box fits in the point cloud, making no spaces between the furthest points and the box wall; thus, the box size is matched to the cloud size. To increase the distinctiveness of the descriptor, we divide the bounding boxes into several cells of equal sizes, and the descriptor values are calculated for each of them. The experiments undertaken in [9, 36] showed that dividing the bounding box into three horizontal cells yields the best results in the hand posture recognition problem in most cases (nine cubic cells and three vertical cells were also verified). Therefore, the box is divided to three horizontal cells in this work (see Fig. 1).

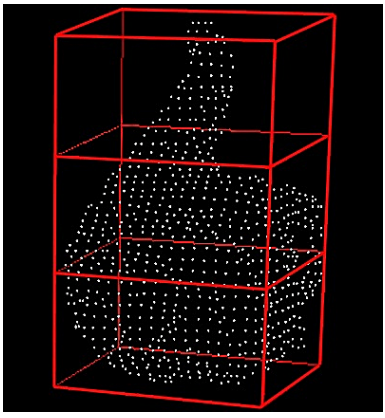

Figure 1. Filtered and downsampled point cloud corresponding to hand with extended thumb and its bounding box divided into three horizontal cells

For posture classification, we use the k-nearest neighbor method with the Euclidean metric and $k = 1$.

4. Descriptors and their modifications

The descriptor of a point cloud can be defined as the method of describing its geometric properties. There are two types of descriptors: local and global. The local descriptors are calculated for a single point and describe their local geometry within the neighborhood of a given radius or number of points. The global descriptors describe the geometry of the whole cluster representing an object [1]. In our work, we use two global descriptors (Ensemble of Shape Functions and Global Radius-based Surface Descriptor), and we propose their modification.

4.1. Ensemble of Shape Functions

Ensemble of Shape Functions (ESF) was first described in [38]. It is a combination of three different functions describing the following properties of a point cloud: distances between the points and areas as well as the angles formed by connecting them. ESF consists of 10 histograms of a size equal to 64 bins. The algorithm of calculating the histograms is repeated a fixed number of times. In each iteration, points $p1$, $p2$, $p3$ are randomly generated, for which the values of features $D2$, $D2-ratio$, $D3$, $A3$ are calculated. According to [38], the number of iterations should be equal to 40,000; however, in the ESF implementation in the Point Cloud Library [29], this number is set to 20,000. We propose to treat this value as parameter ESF_n and verify the possibility of decreasing it in order to make the algorithm faster and maintaining similar recognition rates. The experiments with the ESF_n parameter are presented in Subsection 5.

$D2$ is a feature consisting of three histograms. Their creation involves calculating the distances between points $p1$, $p2$, and $p3$. Then, for each pair, the algorithm determines whether the line connecting the points lies entirely inside the cloud surface, entirely outside the surface (excluding the first and last points), or partially inside and partially outside the surface. Depending on this, a value denoting the distance between the points is placed in one of three sub-histograms: IN, OUT, or MIXED.

The $D2-ratio$ is a feature represented by one histogram that describes the ratio between the lengths of the lines located inside and outside the surface. Only the lines placed inside the MIXED sub-histogram are taken into account.

$D3$ consists of three histograms. To generate them, the algorithm calculates the square root of the area of the triangle whose vertices are located at points $p1$, $p2$, and $p3$. These values are placed in three sub-histograms (similar to the $D2$ feature) depending on whether the triangle sides lie entirely inside the surface, entirely outside it, or both.

The $A3$ feature is represented by three histograms that are created based on calculating the angles between the lines connecting points $p1$, $p2$, and $p3$. These values are placed in three sub-histograms depending on whether the line opposite to the angle lies entirely inside the surface, entirely outside it, or both.

Figure 2 illustrates the method of calculating particular histograms. The authors of ESF claim that the descriptor is able to cope with camera characteristic differences, thus enabling the robust classification of 3D objects.

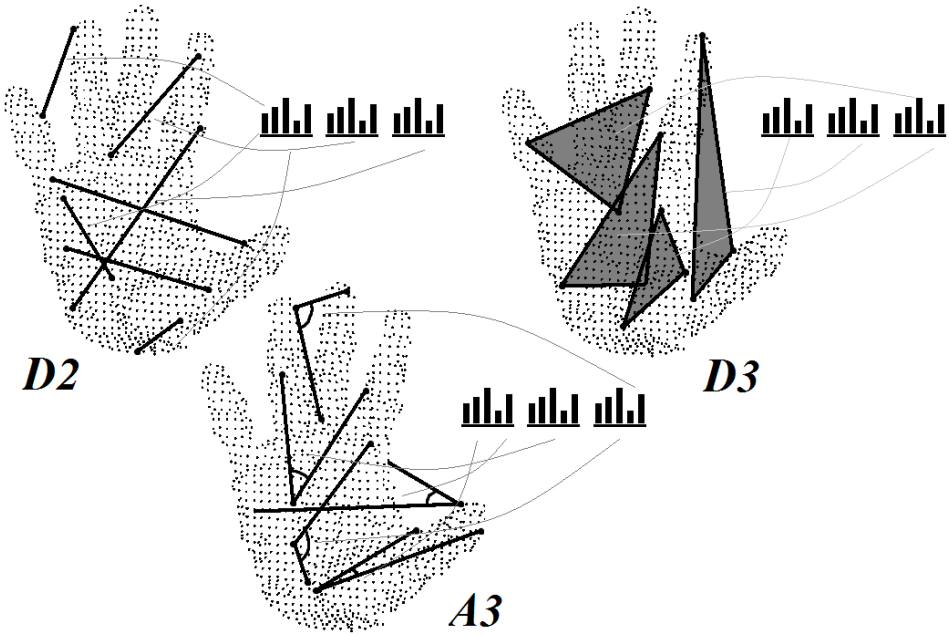

Figure 2. Calculation of ESF features

It is worth noting that the values of the $D2$, $D3$, and $A3$ features are placed in the IN sub-histogram only if the surface coinciding with the lines is almost completely flat, and the values are placed in the OUT sub-histogram if the surface is very convex. Such situations are relatively rare, which results in "flat" sub-histograms IN and OUT (generated by a small number of values) that are unrepresentative. Our proposition is a modification of the descriptor consisting of replacing the IN and OUT sub-histograms with MOSTLY-IN and MOSTLY-OUT cumulating values, for which the lines lie mostly inside the cloud surface or mostly outside it. For example, in the case of the $D2$ feature, if no more than $ESF2_{mo}\%$ of the line length lies inside the surface, this value is placed in MOSTLY-OUT; whereas, if no less than 100-$ESF2_{mi}\%$ of the line length lies inside the surface, the value is placed in MOSTLY-IN. $ESF2_{mo}$ and $ESF2_{mi}$ are the parameters of the method. The experiments performed for several datasets let us set these values as follows: $ESF2_{mo} = 35$, $ESF2_{mi} = 20$ for the $A3$ and $D3$ features, and $ESF2_{mo} = 25$, $ESF2_{mi} = 35$ for the $D - 2$ and $D2 - ratio$ features.

In Section 5, the results of the experiments verifying the recognition rates using the original as well as the modified version (called ESF2) are presented.

To illustrate the effects of ESF and ESF2, we show the histograms calculated for two different hand postures in Figure 3. Please note "flat" sub-histograms IN and OUT, which justify the proposed modifications of the ESF descriptor. Sub-histograms MOSTLY-IN and MOSTLY-OUT of the modified ESF2 descriptor visibly contain more values.

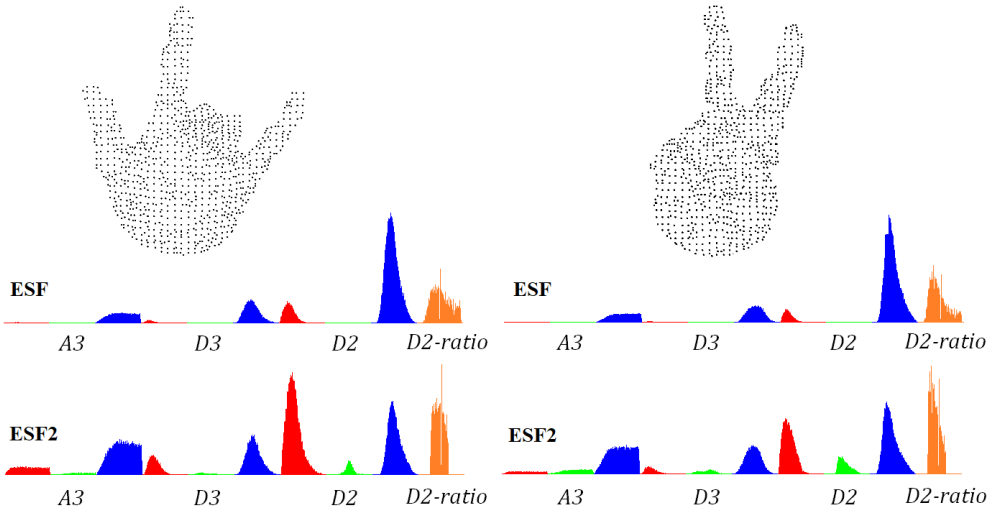

Figure 3. ESF histograms generated for all point clouds representing two hand postures: subhistograms IN and MOSTLY-IN marked in red; OUT and MOSTLY-OUT – green; MIXED – blue; $D2 - ratio$ histogram marked in orange

4.2. Global Radius-based Surface Descriptor

The Global Radius-based Surface Descriptor (GRSD) was described for the first time in [18]. It is a global version of the Radius-based Surface Descriptor (RSD) [16, 17] with some elements of the Global Fast Point Feature Histogram (GFPFH) [30] and the Speeded Up Robust Feature (SURF) method [2]. The GRSD describes the radial relationships of the points with their neighborhood. To explain the algorithm of the descriptor, the local RSD must be described.

RSD is created in the following way: for each pair of point and its neighbor, the distance between them and the difference between their normal vectors are calculated. Then, a sphere is defined with these two points located on its surface. The normal vectors calculated for the points have to be perpendicular to the sphere's surface. Note that, if the perpendicularity requirement is omitted, there is an infinite number of spheres for each pair of points. Finally, the largest and smallest spheres are chosen, and their radii are used to make a descriptor of the point. It can be observed that, if both points lie on a curved surface of a cylinder, the radius of the generated sphere is approximately equal to the cylinder's radius. If, in turn, the points lie on a plane, the radius is infinitely long.

The algorithm of the GRSD calculation has two parameters: $GRSD_{neighbors}$ – the number of points belonging to the local neighborhood used to estimate the normal vectors, and $GRSD_{rnp}$ – the radius within which the neighboring points are chosen. We performed the experiments of verifying the recognition rates depending on the chosen parameters, which allowed us set the value of $VFH_{neighbors}$ to 45. We observed that this value does not affect the results significantly if it is not too small (below 20). The value of $GRSD_{nrp}$ was set to 4.5 mm according to the experiments.

The first step in calculating the GRSD is the surface categorization. The RSD descriptors of the points determined in the previous step are used as an input of the Conditional Random Field (CRF) algorithm [19], which assigns one of the following labels to the point: plane, cylinder, sharp edge, or noise, rim, sphere. At this stage, the point cloud has categorized each point depending on the type of object or certain region to which the point belongs (see Fig. 4).

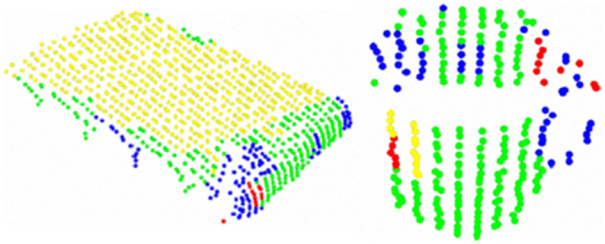

Figure 4. Surface categorization performed for each point in the two objects; different colors represent the following categories: plane – yellow; cylinder – green; sharp edge or noise – red; rim (boundary, transition between surfaces) – blue; sphere (not classified in the presented point clouds); based on [16]

The next step is the generation of an octree – a hierarchic structure for storing voxels, facilitating their compression and searching, and point cloud downsampling. For each created voxel, the algorithm assigns the surface category that most likely represents it. This is the category of the most commonly occurring points within the voxel. Then, for each pair of voxels, a line connecting their center points is generated (see Fig. 5). The next step is finding the voxels through which the line passes and determining whether at least one point lies within them. Thus, intersection histogram H_i is created (see Fig. 6), where i is the number of voxel pairs. The histogram assigns its category number to each voxel (or 0 if the voxel is empty; i.e., contains no points [5]).

Calculated histograms H_i have variable lengths of horizontal axes. Based on these, single global histogram Hg of a fixed length is created. Each bin of H_g represents a different combination of transitions between the voxels of various categories (including Category 0), whereby pairs $\langle category_A, category_B \rangle$ and $\langle category_B, category_A \rangle$ are equivalent. The bins of H_g are filled in by the values of the particular transitions present in each H_i. The size of H_g is equal to the number of two-element combinations with repetitions from the $GRSD_{nc}$-element set, where $GRSD_{nc}$ is the number of

surface categories including Category 0 (of empty voxels): $(nc+2)\cdot(nc+1)/2$. For $GRSD_{nc} = 6$, this number is equal to 21; thus, GRSD is represented by histogram H_g consisting of 21 bins.

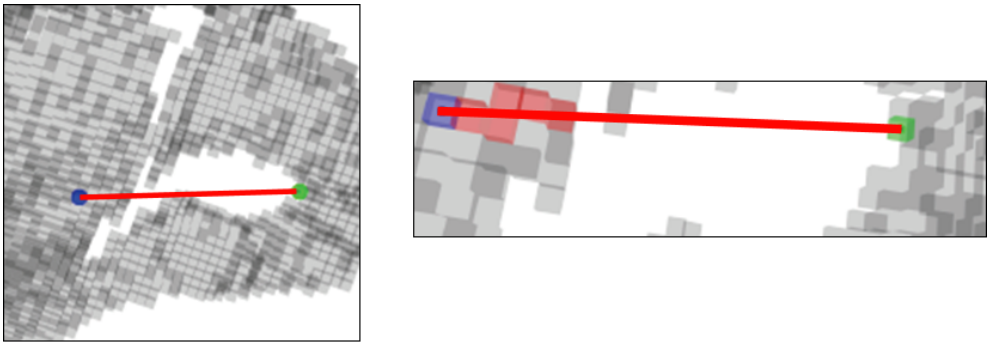

Figure 5. Calculation of GRSD descriptor shown on voxel grid; figure presents example line connecting 2 voxels created based on point cloud; based on [30]

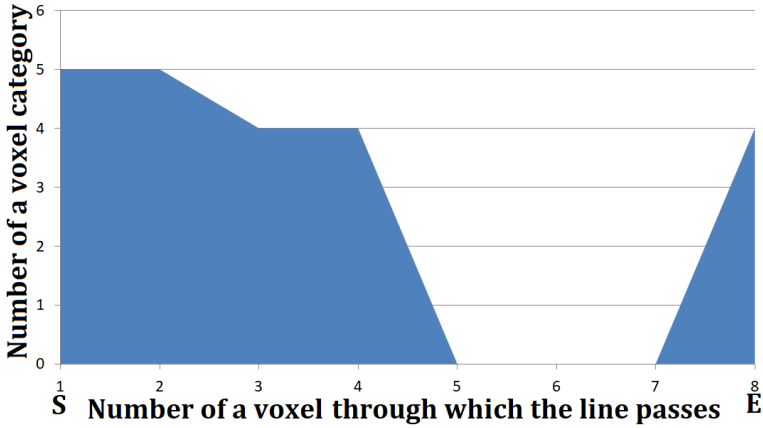

Figure 6. Example intersection histogram H_i: S is starting point; E – ending point

To illustrate the effects of the GRSD descriptor, we show the histograms calculated for two different hand postures in Figure 7.

We propose a modification of the descriptor in which there are no 21-element histograms H_g based on transition combinations. Instead, we use histograms H_gc of a size equal to $GRSD_{nc}$ that are based only on the surface categories; i.e., each bin represents the number of voxels of a particular category (omitting the category of empty voxels).

Figure 7. GRSD histograms generated for all point clouds representing two hand postures

If the set of categories is properly chosen, the modified descriptor (called GRSD-C – the "C" stands for categorization) is more effective than the standard GRSD in terms of recognition rates. Such a modification also decreases the number of features (histograms have smaller sizes) as well as the number of operations that the algorithm needs to calculate the features, and the method is simpler to implement. In Section 5, we present the results of the experiments verifying the recognition rates using the original and modified version of the GRSD for various sets of categories.

4.3. Feature vectors

The single ESF features consist of three histograms of a size equal to 64 bins. To avoid problems with high dimensionality, the histograms are represented by their mean and standard deviation. Then, these values are normalized by scaling them to a range of [0–1] using the minimum and maximum value of each feature from a training set. Finally, the feature values are saved to the feature vector. The GRSD descriptor consists of only one histogram of a size equal to 21 bins, while the histogram of its modified version (GRSD-C) has 2 to 5 bins. In this case, the experiments were performed for both the whole histograms (where the single bin values were the feature vector values) and its representation by mean and standard deviation.

The size of a feature vector, called FV_{size}, is calculated as follows:

$$FV_{size} = nc \cdot \sum_{i=i}^{nf} nh_i \cdot nr_i$$

where:

nc – number of cells to which bounding box is divided;
nf – number of descriptor features (e.g., two features if GRSD-C and $D2$ of ESF2 are used);
nh_i – number of histograms of i-th feature;
nr_i – number of values representing histograms of i-th feature (value is equal to 2 in case of histograms represented as mean and standard deviation).

For example, if the feature vector consists of one ESF and one GRSD feature (for the case of representing histograms by each of their bins; i.e., without calculation of the mean and standard deviation), the feature vector size is equal to $3 \cdot ((3 \cdot 2) + (1 \cdot 21)) = 81$. It is worth noting that having a feature vector that is too large is disadvantageous in terms of classifier learning speed, the curse of dimensionality problem, or in the case of some classifiers, the classification speed. Figure 8 presents the example feature vector of ESF2 feature $D2$ and its sub-histograms generated for all three horizontal cells of the point cloud bounding box.

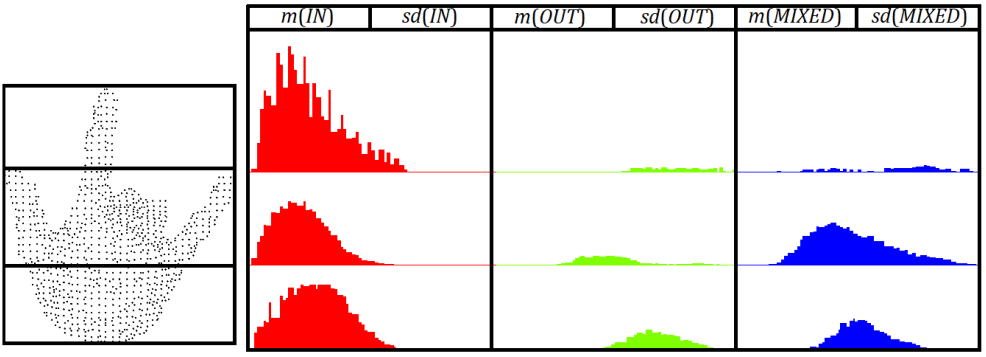

Figure 8. Feature vector with three sub-histograms of ESF2 $D2$ feature calculated for three horizontal cells of example hand posture point cloud; sub-histograms are represented by mean m and standard deviation sd

5. Experiments

The experiments were carried out using four datasets. The first dataset was recorded by the author with the help of two people. This is described in [9] and publicly available at `http://vision.kia.prz.edu.pl/statictof.php`. Denoted as DS_W, the dataset includes 16 hand postures performed 20 times by 3 people, which is 960 executions in total. The hand postures are letters of the Polish finger alphabet that are represented with static gestures (A, B, C, E, I, L, M, N, O, P, R, S, T, U, W, and Y). Distinguishing between some letters of this alphabet is challenging. For example, the postures for the letters O, S, and T differ only in slight shifts of the index finger in relation to the thumb. The gestures were recorded using a MESA SR4000 camera and were performed with the hands facing upwards.

The second dataset (denoted as DS_R) was first described in [27] and is available at `http://eeeweba.ntu.edu.sg/computervision/people/home/renzhou/HandGesture.htm`. The dataset includes 10 hand postures performed 10 times by 10 people, which is 1000 executions in total. The gestures were recorded using a Kinect camera and were performed with a quite variable orientation around the z axis (coinciding with the camera's optical axis) differing up to about 135 degrees.

The third dataset (denoted as DS_D) was introduced in [14, 15] and is available at `http://lttm.dei.unipd.it/downloads/gesture`. It includes 10 hand postures performed 10 times by 14 people, which is 1400 executions in total. The hand postures are letters of the American finger alphabet (A, D, I, L, V, W, and Y), digits (5 and 7), and one additional posture – the inner side of the hand with extended little finger, index finger, and thumb. The gestures were performed with a variable orientation around all axes (x, y, z). Therefore, some postures related to the same gesture class differ significantly, even for a human. The gestures were recorded by Kinect and Leap Motion (skeletal data) cameras; however, in this work, only the depth data acquired from Kinect was used.

The last dataset (denoted as DS_P) was first described in [24] and is available at `http://empslocal.ex.ac.uk/people/staff/np331/index.php?section=FingerSpellingDataset`. It includes 24 hand postures performed a variable number of times (around 550) by 5 people, which amounted to 64,749 executions in total. The subsequent gesture executions are the video file frames that differ from neighboring frames only a little; therefore, there are many similar executions in the dataset. The experiments using such a large dataset are very time-consuming; therefore, we had to reduce it. We used the fact of the similarity of neighboring gesture executions to remove redundant data by choosing only 100 files that were divisible by 4 for each person and gesture (starting at file number 100). The offset from the beginning was caused by accidentally captured head images (without a visible hand) or incorrect hand postures present in the first files in some cases. Thus, the modified dataset includes 12,000 files. The hand postures are all of the letters of the American finger alphabet except J and Z (because they involve motion). The gestures were recorded by the Kinect camera. The depth maps are preprocessed in a way that only those fragments containing a hand and, in some cases, a part of a forearm are visible. Thus, the depth maps have a variable resolution. Hand segmentation was necessary, however, because the depth data in small fragments around a hand contained background objects, and the visible forearm parts are relatively long in some cases. DS_P is the most challenging of the four considered datasets because of its large number of gesture classes and variable hand orientations, which are similar to the DS_D dataset.

It should be mentioned that the gestures from all of the used datasets were performed on a non-uniform and often cluttered background. In some cases in the DS_P dataset, the head of the person performing the gestures was visible on the background of the hand.

The experiments with the ZD_W and DS_P datasets was performed using the segmentation method of "gesticulation area," while in the case of DS_R and DS_D, "precise segmentation" was used. We used the method of "gesticulation area" because the DS_W and DS_P datasets do not meet the following requirements of the "precise segmentation" method: the DS_W dataset was recorded by the MESA SR4000 camera, which provides data with insufficiently accurate depth; in some executions of DS_P, the forearm parts are not visible.

In the experiments, we used 5-fold cross-validation tests for which a training subset consisted of 80% of the executions and a testing subset accounted for 20% of the executions in each test. The results of all five tests are presented as a single mean value. In each experiment in which the ESF or ESF2 features were used, the cross-validation tests were repeated 30 times; their mean and standard deviation values are shown (separated by slashes in the tables). The cause of the test repetition is the fact that the ESF descriptor is nondeterministic – this is also true for ESF2; therefore, the repeated experiments using the same data, methods, and parameters can be different.

Table 1 compares the recognition rates of the single features of ESF and ESF2. The results show that every feature of ESF2 save $D2 - ratio$ are more effective than their counterparts from ESF in posture recognition. The worse result for $D2 - ratio$ (that is, the least effective feature in both ESF versions) can be explained by the fact that it is related to the ratios of the lengths of the line fragments present in the $MIXED$ sub-histogram of the $D2$ feature, and this histogram takes fewer values than ESF to increase the number of values in the other two sub-histograms.

Table 1
Comparison of recognition rates for single features of original ESF descriptor and its modified version – ESF2

Feature	Recognition rate/standard deviation [%]					Feature vector size
	DS_W	DS_R	DS_D	DS_P	mean	
ESF($A3$)	61.1/1.65	82.7/0.96	51.3/1.23	49/0.52	61	18
ESF($D3$)	42.8/1.78	74.2/1.27	50.3/0.9	43.3/0.6	52.7	18
ESF($D2$)	69.9/0.77	83.3/0.58	47.6/0.55	51.1/0.24	63	18
ESF($D2 - ratio$)	68.1/0.44	65.7/0.34	43/0.47	32.1/0.13	52.2	6
ESF2($A3$)	80.9/0.68	90.1/0.61	65.9/0.52	61.8/0.31	74.7	18
ESF2($D3$)	63.4/1.2	84/0.65	54.5/0.44	52/0.58	63.5	18
ESF2($D2$)	81.4/0.87	86.3/0.51	57.3/0.45	57.8/0.17	70.7	18
ESF2($D2-ratio$)	57.7/0.79	63.8/0.58	35.2/0.71	32.3/0.21	47.3	6

Table 2 compares the recognition rates of the original GRSD descriptor and its modified version – GRSD-C. For the original GRSD, two results are shown. The first is the case of those feature vectors consisting of all 21 bins of the GRSD histogram. In the second case, the feature vectors containing the mean and standard deviations were used. The first variant turned out to be more accurate, with a difference of 11.6%. For the GRSD-C descriptor, the results are presented for three cases differing in the number of categories. The experiments started from all five categories proposed in the original GRSD method. Then, we tried to connect particular categories that, when used independently, do not improve the results (and even make them worse in certain cases).

Table 2
Comparison of recognition rates of original GRSD descriptor and its modified version – GRSD-C

Descriptor	Recognition rate [%]					Feature vector size
	DS_W	DS_R	DS_D	DS_P	mean	
GRSD (entire histogram)	61.5	69.5	52.6	46.7	57.6	63
GRSD (mean and st. dev. of histogram)	58.4	56.1	38.7	30.7	46	6
GRSD-C (5 cat.)	66.8	57.2	48.4	48.4	55.2	15
GRSD-C (3 cat.)	73.5	65.8	58.5	47.9	61.4	9
GRSD-C (2 cat.)	78.9	70.8	54.1	45.8	62.4	6

As a result, we used the following sets of categories.
Five categories – GRSD-C(5 cat.):

1. plane;
2. cylinder;
3. sharp edge or noise;
4. rim;
5. sphere.

Three categories – GRSD-C(3 cat.):

1. plane, cylinder, or rim;
2. sharp edge or noise;
3. sphere.

Two categories – GRSD-C(2 cat.):

1. plane, cylinder, rim, or sphere;
2. sharp edge or noise.

Such a composed set of categories result in improved classification rates as related to the first variant of the original GRSD averaged for all sets with a difference of 4.8%. Note that the feature vectors have significantly smaller sizes for GRSD-C(2 cat.). In the case of the second GRSD variant with the same feature vector size (6 values), the result for the proposed GRSD-C is higher by 16.4%. Analyzing the results for the particular datasets, it can be seen that GRSD-C(2 cat.) is not the most effective in all cases. For the DS_D dataset, the best is variant GRSD-C(3 cat.), while for DS_P, the best is variant GRSD-C(5 cat.). However, for each dataset, at least one of the GRSD-C variants is better than both variants from the original GRSD. These results

led to the conclusion that the GRSD-C modified descriptor is more accurate in the problem of posture recognition than its original version.

In Table 3, we present the results obtained for feature vectors composed from both ESF and GRSD as well as their modified versions. For ESF and ESF2, only the two most effective features (in terms of the recognition rate presented in Table 1) were used, because we observed that adding other features does not improve the results (or improves them only slightly). In the case of the original GRSD, there are two results related to both feature representations. For GRSD-C, only the best set of categories were used. The average results for the modified descriptors are greater by about 14% than those for their original versions.

Table 3

Comparison of recognition rates for feature vectors, including original ESF and GRSD descriptors and their modified versions

Features	Recognition rate/standard deviation [%]					Feature vector size
	DS_W	DS_R	DS_D	DS_P	mean	
ESF2 ($A3$, $D2$), GRSD-C (2-cat.)	89.8/0.48	92.1/0.52	72.7/0.33	68.9/0.25	80.9	42
ESF ($A3$, $D2$), GRSD-C (entire histogram)	69.8/0.14	84.9/0.73	56.9/0.1	55.6/0.1	66.8	99
ESF ($A3$, $D2$), GRSD (mean and standard deviation of histogram)	71.3/1.51	88/1.17	54.5/0.85	55.8/0.18	67.4	42

In all of the previously described experiments in which either ESF or ESF2 was used, the ESF_n parameter (algorithm iteration number) was set to 20,000. The value of this parameter significantly affects the processing time of the algorithm (it is more or less proportional); therefore, we decided to determine its impact on the recognition rate. An experiment was conducted in which the ESF_n value was decreased down to 1000 with steps equal to 1000; for each value, the validation tests on the DS_R dataset were performed using all of the ESF2 features. The experiment results are shown in Figure 9. A downward trend can be observed starting from value 5000; however, a deterioration of the results greater than 0.5% (in relation to the default ESF_N value) can be seen for values less than or equal to 3000. In the DS_R dataset, the hands were shown at a distance of about 80 cm from the camera. The average number of cloud points for this dataset is equal to 790 after noise removal and downsampling. Therefore, based on the results of the experiments, we postulate the following: in the problem of hand posture recognition, if the hands are shown at a distance of about

80 cm from the camera and the point clouds are downsampled, the ESF_n value can be decreased from 20,000 to 4000 without a significant reduction in the recognition rate. In this case, the calculation of the descriptor values is approximately five times shorter: 42.33 ms for $ESF_n = 20\,000$ and 8.76 ms for $ESF_n = 4000$ (on a laptop with an Intel Core i7-4710HQ and a 2.5 GHz CPU).

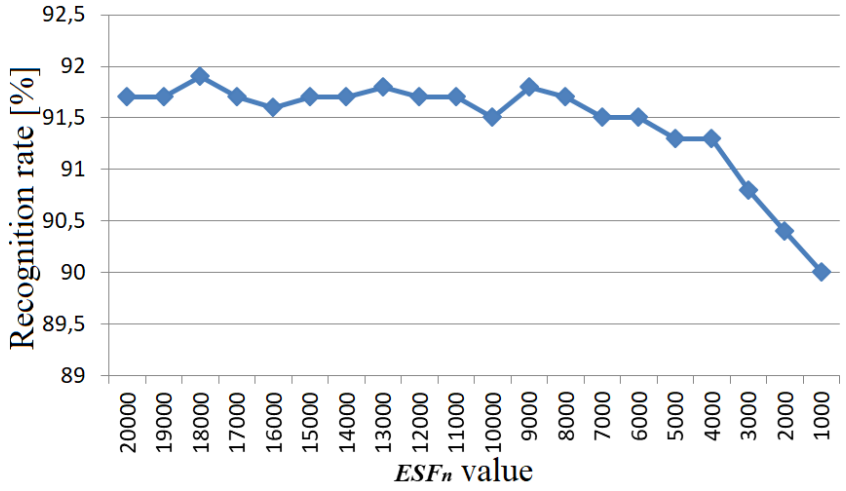

Figure 9. Recognition rates (in [%] on vertical axis) obtained using DS_R dataset depending on ESF_n parameter value (horizontal axis)

To show what kind of postures are easy and difficult to recognize by our method, we present example color images representing the hand shapes from each class recorded during the depth-data acquisition of the DS_R dataset in Figure 10. We also show the confusion matrix for the cross-validation tests performed using feature sets ESF2($A3$, $D2$) and GRSD-C(2-cat.) on the DS_R dataset in Figure 11. The most misclassifications were related to Gesture G7, which was mistaken for G8 12 times. There is a similarity between these postures, as there is an extended thumb and another extended finger in both. Also, Gesture G10 was mistaken for G9 nine times, which is interesting because a thumb and two fingers are extended in G10 while only a thumb is extended in G9. Misclassifications of G10 with G9 may result from the fact that these two postures are the only ones featuring a visible external hand side.

Figure 10. Example DS_R postures from each class

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	100	0	0	0	0	0	0	0	0	0
G2	0	87	9	2	1	0	0	1	0	0
G3	0	1	95	4	0	0	0	0	0	0
G4	0	2	5	93	0	0	0	0	0	0
G5	0	0	0	5	94	0	1	0	0	0
G6	0	0	0	0	0	100	0	0	0	0
G7	1	4	0	0	1	4	76	12	1	1
G8	0	9	0	0	0	0	3	87	0	1
G9	0	0	0	0	0	0	0	0	100	0
G10	0	0	0	0	0	0	0	2	9	89

Figure 11. Confusion matrix obtained for cross-validation tests on DS_R dataset using feature sets ESF2($A3$, $D2$) and GRSD-C(2-cat.)

The proposed recognition method using the GRSD-C and ESF2 descriptors alone with no other features yields good results. However, such an algorithm does not outperform the most accurate methods found in the literature. Therefore, we propose to combine GRSD-C's and ESF2's most effective features and configurations with the Viewpoint Feature Histogram (VFH) descriptor used in our previous works [9,36] for hand-gesture recognition.

Table 4

Comparison between recognition rates of our method using VFH(all features), ESF2($A3$), GRSD-C(2-cat.), and algorithms found in literature; experiments were performed using 5-fold cross-validation tests and DS_p dataset

Method	Recognition rate [%]
Keskin et al. [10]	43
Pugeault et al. [24]	49
Pedersoli et al. [22]	56
Kuznetsova et al. [11]	57
Wang et al. [35]	58.3
Dalal et al. [3,6]	65.4
Dong et al. [5]	70
Zhang et al. [40]	73.3
our method – k-nearest neighbor classifier	74.8
Wang et al. [34]	75.8
Feng et al. [6]	78.7
our method – SVM classifier	79.6

In Table 4, we present a comparison of the developed method and feature sets VFH(all features), ESF2($A3$), and GRSD-C(2-cat.) with the results found in the literature. The methods of every cited work were tested using 5-fold cross-validation performed on the DS_p dataset. These tests are person-independent, which means that no person from the testing set is present in the training set. In this experiment, we did not merely choose those files divisible by 4 to reduce the dataset because we wanted a reliable comparison with the other methods. Two results of our method are presented: the first was obtained using the k-nearest neighbor classifier as in the previous experiments, and in the second case, we used the SVM classifier with all-versus-all multi-class classification. The version of our method with the KNN classifier yielded relatively good results, and the second version using the SVM classifier outperformed each of the other compared methods.

6. Conclusions

In this paper, we presented an approach for the recognition of static hand gestures based on 3D data and point cloud descriptors: Ensemble of Shape Functions and Global Radius-based Surface Descriptor. We proposed modifications of these descriptors, which include the following: (i) replacing transition-based histograms with category-based histograms of the GRSD; (ii) replacing IN and OUT sub-histograms with MOSTLY-IN and MOSTLY-OUT sub-histograms of the ESF descriptor; (iii) replacing a constant value determining the iteration number with the ESF_n parameter that can be adjusted to shorten the computational time of the descriptor calculation method. These modifications increased hand posture recognition rates and decreased the quantity of the used features as well as the computational cost of the algorithms, which was proven by the experiments performed using four challenging publicly available datasets and cross-validation tests.

Future work related to this subject may be adapting the proposed recognition method and modified descriptors in the problem of static hand posture sequence recognition. Such a method may be helpful in developing a system for the recognition of finger-spelled words.

References

[1] Aldoma A., Marton Z.C., Tombari F., Wohlkinger W., Potthast C., Zeisl B., Rusu R.B., Gedikli S., Vincze M.: Tutorial: Point Cloud Library: Three-Dimensional Object Recognition and 6 DOF Pose Estimation, *IEEE Robotics and Automation Magazine*, vol. 9(3), pp. 80–91, 2012. `http://dx.doi.org/10.1109/MRA.2012.2206675`.

[2] Bay H., Ess A., Tuytelaars T., Gool L.V.: Speeded-Up Robust Features (SURF), *Computer Vision and Image Understanding*, vol. 110(3), pp. 346–359, 2008. `http://dx.doi.org/10.1016/j.cviu.2007.09.014`.

[3] Dalal N., Triggs B.: Histograms of oriented gradients for human detection. In: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'05, IEEE Computer Society, Washington DC, pp. 886–893, 2005. http://dx.doi.org/10.1109/CVPR.2005.177.

[4] Dominio F., Donadeo M., Zanuttigh P.: Combining multiple depth-based descriptors for hand gesture recognition, In: *Pattern Recognition Letters*, vol. 50(C), pp. 101–111, 2014. http://dx.doi.org/10.1016/j.patrec.2013.10.010.

[5] Dong C., Leu M.C., Yin Z.: American Sign Language Alphabet Recognition Using Microsoft Kinect. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 44–52, 2015. http://dx.doi.org/10.1109/CVPRW.2015.7301347.

[6] Feng B., He F., Wang X.: Depth-projection-map-based bag of contour fragments for robust hand gesture recognition, *IEEE Transactions on Human-Machine Systems*, vol. 47(4), pp. 511–523, 2016. http://dx.doi.org/10.1109/THMS.2016.2616278.

[7] Flasiński M., Myśliński S.: On the use of graph parsing for recognition of isolated hand postures of Polish Sign Language, In: *Pattern Recognition*, vol. 43(6), pp. 2249–2264, 2010. http://dx.doi.org/10.1016/j.patcog.2010.01.004.

[8] Jeong E., Lee J., Kim D.: Finger-gesture Recognition Glove using Velostat. In: *2011 11th International Conference on Control, Automation and Systems (ICCAS)*, IEEE, 2011.

[9] Kapuściński T., Oszust M., Wysocki M., Warchoł D.: Recognition of Hand Gestures Observed by Depth Cameras, *International Journal of Advanced Robotic Systems*, vol. 12(4), pp. 48–57, 2015. http://dx.doi.org/10.5772/60091.

[10] Keskin C., Kiraç F., Kara Y.E., Akarun L.: Real time hand pose estimation using depth sensors. In: *ICCV Workshops*, IEEE, pp. 1228–1234, 2011. http://dx.doi.org/10.1109/ICCVW.2011.6130391.

[11] Kuznetsova A., Leal-Taixé L., Rosenhahn B.: Real-time sign language recognition using a consumer depth camera. In: *IEEE International Conference on Computer Vision Workshops (ICCVW), 3rd Workshop on Consumer Depth Cameras for Computer Vision (CDC4CV)*, 2013. http://dx.doi.org/10.1109/ICCVW.2013.18.

[12] Li Y.: Multi-scenario gesture recognition using kinect. In: *Proceedings of the 2012 17th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES)*, CGAMES'12, IEEE, Washington, DC, USA, pp. 126–130, 2012. http://dx.doi.org/10.1109/CGames.2012.6314563.

[13] Luis-Pérez F.E., Trujillo-Romero F., Martínez-Velazco W.: Control of a Service Robot Using the Mexican Sign Language. In: Batyrshin I.Z., Sidorov G. (eds.): *MICAI (2)*, *Lecture Notes in Computer Science*, vol. 7095, pp. 419–430. Springer, 2011.

[14] Marin G., Dominio F., Zanuttigh P.: Hand gesture recognition with leap motion and kinect devices. In: *2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2014. http://dx.doi.org/10.1109/ICIP.2014.7025313.

[15] Marin G., Dominio F., Zanuttigh P.: Hand gesture recognition with jointly calibrated Leap Motion and depth sensor, *Multimedia Tools and Applications*, vol. 75(22), pp. 14991–15015, 2016. http://dx.doi.org/10.1007/s11042-015-2451-6.

[16] Marton Z.C., Pangercic D., Blodow N., Beetz M.: Combined 2D–3D categorization and classification for multimodal perception systems, *International Journal of Robotics Research*, vol. 30(11), pp. 1378–1402, 2011. http://dx.doi.org/10.1177/0278364911415897.

[17] Marton Z.C., Pangercic D., Blodow N., Kleinehellefort J., Beetz M.: General 3D modelling of novel objects from a single view. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei, Taiwan, 2010. http://dx.doi.org/10.1109/IROS.2010.5650434.

[18] Marton Z.C., Pangercic D., Rusu R.B., Holzbach A., Beetz M.: Hierarchical object geometric categorization and appearance classification for mobile manipulation. In: *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*. Nashville, TN, USA, 2010.

[19] McCallum A.: Efficiently inducing features of conditional random fields. In: *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, UAI'03, pp. 403–410. Morgan Kaufmann Publishers Inc., San Francisco, 2003.

[20] Molina J., Escudero-Viñolo M., Signoriello A., Pardàs M., Ferrán C., Bescós J., Marqués F., Martínez J.M.: Real-time user independent hand gesture recognition from time-of-flight camera video using static and dynamic models, *Machine Vision and Applications*, vol. 24(1), 2013. http://dx.doi.org/10.1007/s00138-011-0364-6.

[21] Oprisescu S., Su C.R.B.: Automatic static hand gesture recognition using tof cameras. *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pp. 2748–2751. Zenodo, 2012. http://dx.doi.org/10.5281/zenodo.42821.

[22] Pedersoli F., Benini S., Adami N., Leonardi R.: XKin: An open source framework for hand pose and gesture recognition using kinect. In: *The Visual Computer*, vol. 30(10), pp. 1107–1122, 2014. http://dx.doi.org/10.1007/s00371-014-0921-x.

[23] Plouffe G., Cretu A.M.: Static and dynamic hand gesture recognition in depth data using dynamic time warping, *IEEE Transactions on Instrumentation and Measurement*, vol. 65(2), pp. 305–316, 2016. http://dx.doi.org/10.1109/TIM.2015.2498560.

[24] Pugeault N., Bowden R.: Spelling it out: Real-time ASL fingerspelling recognition. In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1114–1119. IEEE, 2011. http://dx.doi.org/10.1109/ICCVW.2011.6130290.

[25] 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*, vol. 74(1), pp. 47–58, 2014. http://dx.doi.org/10.1007/s11265-013-0778-7.

[26] Ren Z., Yuan J., Meng J., Zhang Z.: Robust part-based hand gesture recognition using kinect sensor, In: *IEEE Transactions on Multimedia*, vol. 15(5), pp. 1110–1120, 2013. http://dx.doi.org/10.1109/TMM.2013.2246148.

[27] Ren Z., Yuan J., Zhang Z.: Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In: *Proceedings of the 19th ACM International Conference on Multimedia*, MM'11, pp. 1093–1096. ACM, New York, 2011. http://dx.doi.org/10.1145/2072298.2071946.

[28] Ribó A., Warchoł D., Oszust M.: An approach to gesture recognition with skeletal data using dynamic time warping and nearest neighbour classifier, *International Journal of Intelligent Systems and Applications*, vol. 8(6), pp. 1–8, 2016. http://dx.doi.org/10.5815/ijisa.2016.06.01.

[29] Rusu R.B., Cousins S.: 3D is here: Point Cloud Library (PCL). In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, 2011. Software available at http://pointclouds.org.

[30] Rusu R.B., Holzbach A., Beetz M., Bradski G.: Detecting and segmenting objects for mobile manipulation. In: *Proceedings of IEEE Workshop on Search in 3D and Video (S3DV), held in conjunction with the 12th IEEE International Conference on Computer Vision (ICCV)*. Kyoto, Japan, 2009.

[31] Sridevi K., Sundarambal M., Muralidharan K., Josephine R.L.: FPGA implementation of hand gesture recognition system using neural networks. In: *2017 11th International Conference on Intelligent Systems and Control (ISCO)*. IEEE, 2017. http://dx.doi.org/10.1109/ISCO.2017.7856017.

[32] Tewari D., Srivastava S.K.: A visual recognition of static hand gesture in Indian Sign Language based on Kohonen Self Organizing Map Algorithm, *International Journal of Engineering and Advanced Technology*, vol. 2(2), pp. 165–170, 2012.

[33] Uebersax D., Gall J., den Van Bergh M., Gool L.V.: Real-time sign language letter and word recognition from depth data. In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, 2011. http://dx.doi.org/10.1109/ICCVW.2011.6130267.

[34] Wang C., Liu Z., Chan S.C.: Superpixel-based hand gesture recognition with kinect depth camera, *IEEE Transactions on Multimedia*, vol. 17(1), pp. 29–39, 2015. http://dx.doi.org/10.1109/TMM.2014.2374357.

[35] Wang Y., Yang R.: Real-time hand posture recognition based on hand dominant line using kinect. In: *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2013. http://dx.doi.org/10.1109/ICMEW.2013.6618237.

[36] Warchoł D., Wysocki M.: Recognition of hand postures based on a point cloud descriptor and a feature of extended fingers, *Journal of Automation, Mobile Robotics and Intelligent Systems*, vol. 10(01), pp. 48–57, 2016. http://dx.doi.org/10.14313/JAMRIS_1-2016/7.

[37] Wen Y., Hu C., Yu G., Wang C.: A robust method of detecting hand gestures using depth sensors. In: *Proceedings of the 2012 IEEE International Workshop on Haptic Audio Visual Environments and Games*, pp. 72–77. 2012. http://dx.doi.org/10.1109/HAVE.2012.6374441.

[38] Wohlkinger W., Vincze M.: Ensemble of shape functions for 3D object classification. In: *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2987–2992. IEEE, 2011. http://dx.doi.org/10.1109/ROBIO.2011.6181760.

[39] Xu B., Zhou Z., Huang J., Huang Y.: Static hand gesture recognition based on rgb-d image and arm removal. In: *Advances in Neural Networks – ISNN 2017. 14th International Symposium, ISNN 2017, Sapporo, Hakodate, and Muroran, Hokkaido, Japan, June 21–26, 2017, Proceedings, Part I*, pp. 180–187. Springer International Publishing, 2017. http://dx.doi.org/10.1007/978-3-319-59072-1_22.

[40] Zhang C., Yang X., Tian Y.: Histogram of 3D facets: A characteristic descriptor for hand gesture recognition. In: *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, pp. 1–8. IEEE, 2013. http://dx.doi.org/10.1109/FG.2013.6553754.

Affiliations

Dawid Warchoł
Rzeszów University of Technology, Faculty of Electrical and Computer Engineering, Department of Computer and Control Engineering, W. Pola 2, 35-959 Rzeszów, Poland, dawwar@kia.prz.edu.pl

Received: 20.11.2017
Revised: 05.03.2018
Accepted: 05.03.2018