FINGER DETECTION AND HAND POSTURE RECOGNITION BASED ON DEPTH INFORMATION

Stergios Poularakis and Ioannis Katsavounidis

Department of Electrical and Computer Engineering, University of Thessaly, Greece

ABSTRACT

In this work, we propose a novel framework for automatic finger detection and hand posture recognition, based mainly on depth information. Our method locates apex–shaped structures in a hand contour and deals efficiently with the challenging problem of partially merged fingers. Hand posture recognition is achieved using Fourier Descriptors of the contour, while global information about the fingers helps reducing the size of the search space. Our experiments on a dataset obtained from a Kinect device confirm the high recognition accuracy of our approach.

Index Terms— hand detection, finger detection, depth camera.

1. INTRODUCTION

Hand gesture recognition is gradually becoming popular as a means of Human Computer Interaction (HCI). The release of Kinect $^{\text{TM}}$, along with an easily adaptable Software Development Kit (SDK), made the possible applications grow tremendously and become a basic HCI part of many products. Additionally to standard 2D video, Kinect offers depth information, which provides more complete information about the 3D world and helps solving some hard problems, such as varying lighting or cluttered background. However, gesture recognition is still an interesting and challenging problem, since some assumptions are still necessary, even with the use of depth information.

In this work, we focus on recognition of *hand postures*, i.e. gestures where information lies only on the shape of the still hand, regardless of its position. We show that *local* features, such as Fourier Descriptors of a palm's contour can provide very high recognition results for a limited number of simple postures. However, *global* information, based on the number, shape and relative position of fingers, is probably the most critical for posture recognition, while it is also necessary for more advanced tasks. Specifically, we evaluate some purely global features of the fingers, that provide high recognition results, too. Motivated by the above, we also propose an automated method for reliable finger detection and show that we can significantly improve the recognition accuracy of local feature methods, through search space reduction based on the number of fingers.

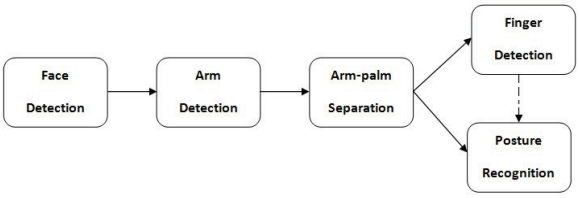

Fig. 1. General structure of our proposed system. Finger detection is optional, to improve recognition results through search space reduction (as described in Sec. 3.3.2).

(a) (b) (c)

Fig. 2. Three of the 10 different hand postures used in [5] and in our experiments. One can observe the necessity of using depth under highly correlated background (e.g. in (b,c)).

The remaining of this paper is organized as follows. In Sec. 2 we provide a short overview of some recent approaches which also use depth information for hand posture recognition. In Sec. 3 we present in detail our approach for finger detection and hand posture recognition, followed by our experimental results in Sec. 4. Finally, Sec. 5 concludes this work and addresses our plans for the future.

2. RELATED WORK

In a very recent work, Kulshreshth *et al.* [1] used Fourier Descriptors to process data from a Kinect system and recognized the number of fingers in the palm with 90% accuracy. Bagdanov *et al.* [2] classified a palm as either open or closed, using SURF features [3] and a non-linear Support Vector Machine, reporting 87.8% accuracy. Kurakin *et al.* [4] performed recognition of 12 dynamic American Sign Language (ASL) signs, using both shape and motion features.

Recently, some other approaches based on *global* features and hand models appeared as well. Keskin *et al.* [6, 7] fit a 3D skeleton to hand points, achieving almost perfect recognition (99.9%) for the 10 ASL digits on synthetic and real datasets. Billiet *et al.* [8] learned a hand model with 9 Degrees of Free-

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

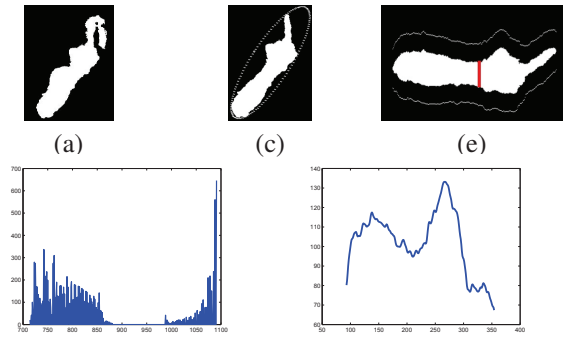

Fig. 3. (a) Binary mask after depth thresholding. (b) Corresponding histogram of depth values. Note arm pixels (small depth values) and noisy part (big depth values). (c) Resulting hand and Minimum Enclosing Ellipsoid after applying Otsu's segmentation. (d) The signal of hand widths, $widths(n)$. (e) Final cutting point, (vertical red line). The upper and the lower signals are shown slightly shifted for clarity.

dom, describing each finger as *stretched* or *closed*, and recognized 8 postures. Oikonomidis *et al.* [9] tracked the 3D position, orientation and full articulation of a human hand, based on Particle Swarm Optimization and managed to control computational complexity by utilizing GPUs.

The works of Ren *et al.* [5] and Doliotis *et al.* [10] are the most relevant to our work. Doliotis *et al.* [10] performed hand–palm separation from Kinect 3D data and then used the Chamfer distance [11] to match real postures to $82,560$ synthetic hand shapes. In our approach, we use a slightly modified version of their method for hand–palm separation. Ren *et al.* [5] required the user to wear a black bracelet on the gesturing hand's wrist, for easier hand–palm separation, and performed finger detection using a distance-based profile of the palm and shape decomposition. The same authors proposed Finger-Earth Mover's Distance (FEMD) to perform posture recognition.

In this work we use the same dataset as in [5], but we avoid the use of a black bracelet, since it might be restrictive in certain cases (e.g. *What if the user loses the black bracelet?*). For that reason, we restrict the use of pixel luminance values to detect face location during initialization, and we only use depth information for all the other tasks, i.e. hand detection, segmentation and representation. Moreover, we propose a method for automated finger detection, which is much faster than the method presented in [5] and provides very accurate results. Although our experimental results are significantly better than those reported in [5], we still believe that full (Y, C_b, C_r) information can further improve and make even more robust hand detection, and we plan to address this issue in future work, using a more appropriate dataset.

Fig. 4. (a) Hand M_0 and the maximum inscribed circle. (b) The estimated palm M_{palm}, after morphological opening. (c) Subtraction of the two masks $M_0 - M_{palm}$. (d) Resulting mask $M_{fingers}$ with candidate finger components.

3. OUR APPROACH

The general structure of our method is shown in Fig. 1. The initialization step involves periodic applications of the Viola–Jones face detector [12] on the luminance component of an image; we assume this is always possible when the user first appears in the scene. When a user is eventually detected, we compute the average depth of the face region, T_f. Kinect's depth values typically have a range of $[0, 2047]$, with lower values denoting distances closer to the camera. As a result, T_f is the maximum depth that a hand may appear, assuming it is always closer to the camera, as in Fig. 2.

3.1. Hand detection

In order to locate the gesturing arm, we apply depth thresholding, keeping pixels with depth $d < T_f - T_0$, where T_0 is a small value that typically represents the minimum distance from the face plane to the waving hand - we found out that a value of $T_0 = 100$ is suitable for our dataset. Subsequently, we perform Connected Component Analysis (CCA) and keep the biggest component as the candidate arm. Since depth thresholding may actually not perform a perfect segmentation (Fig. 3), we further apply Otsu's segmentation algorithm [13] on the final component to clean it from any background noise. Finally, similar to [10], we compute the Minimum Enclosing Ellipsoid (MEE) [14, 15] to find the elongation axis and rotate the arm in a horizontal position, such that the palm is always at the right side, as shown on Fig. 3-e.

In the following, let's assume Cartesian coordinate system, with the $(0, 0)$ point in the lower-left corner of each image. For hand–palm separation, restricting ourselves in the bounding box of the arm, we scan all x-positions of pixels that belong to the largest component and track the pixels that have the minimum ($low(x)$) and maximum ($upp(x)$) y-coordinates. These two sequences form two 1D-signals, as shown in Fig. 3-e. Subtracting these two signals produces the signal of widths $widths(x)$ (Fig. 3-d). We observe that the arm–palm separating point, (x), can be located at the local minimum of $widths(x)$ that is closest to the global maximum. The final separation result is a mask M_0 containing the palm, as shown in Fig. 4-a.

3.2. Finger detection

For finger detection, we first estimate the palm's radius, R_p, by solving for the Maximum Inscribed Circle [16] (Fig. 4-

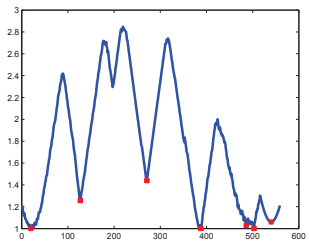

Fig. 5. The resulting signal $r(n)$ and the FAIs (*shown as red squares*) found by the method of [17]. Note that low-height apexes can be easily rejected.

a). We then apply morphological image opening on the mask M_0 with a disk of radius $R_{disk} = 0.5 \cdot R_p$, in order to keep only the palm M_{palm} (Fig. 4-b). Subtracting M_{palm} from M_0 keeps mainly the fingers (Fig. 4-c), while we remove small, noise-like components in the mask by additional morphological opening with a smaller structuring element which finally results in the mask $M_{fingers}$ (Fig. 4-d).

While this processing is simple and fast, two problems may appear:

1. *False Fingers*. Some components may survive the preprocessing step because they are large enough and not because they have a valid finger shape.
2. *Merged Fingers*. Some fingers may appear as one component due to viewpoint or to device artifacts.

To alleviate these problems, we propose a novel method for finger detection, which first detects candidate fingers and then uses the mask $M_{fingers}$ to confirm the validity of the results. Similar to [5], we find the contour coordinates $\{x(n), y(n)\}$ of M_0 and then form the signal of radial distances, $r(n)$:

$$r(n) = ((x(n) - c_x)^2 + (y(n) - y_c)^2)/R_p \qquad (1)$$

where (c_x, c_y) denote the centroid coordinates of M_{palm}.

As we see in Fig. 5, the signal $r(n)$ presents apex–shaped lobes around finger areas. This property was also observed by Ren *et al.* [5], who used near-convex hand decomposition [18], obtaining accurate finger detection results, but at a high computational cost. In our work, we propose a much simpler method, based on the method presented in [17] for motion analysis. This algorithm forms teams of two minima and one maximum and tries to combine neighbouring teams, in order to detect apex-shaped parts, called Full Action Instances (FAI), in 1D signals; each FAI is roughly a mountain peak between two valleys. In our approach, we use this algorithm for finger detection, since fingers resemble FAIs. After the initial FAI detection (Fig. 5), we reject small FAIs based on a minimum apex height, T_h, to obtain the final candidate fingers. This process removes false fingers from the mask $M_{fingers}$, since it keeps only apex-shaped components of the hand. In our experiments, we used $T_h = 0.3 \cdot R_p$, i.e. a finger is at least a third the radius of the palm.

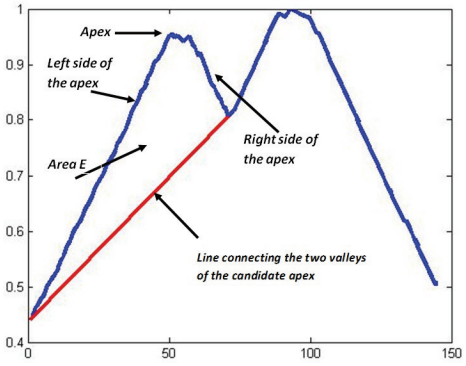

Fig. 6. An example of our method for apex detection and FAI splitting, dealing with two merged fingers. When area E is larger than a threshold T_a, an apex is detected in signal $r(n)$.

In order to deal with the problem of merged fingers, we propose a novel algorithm that detects apexes in 1D signals, which we then apply on each FAI. Our main observation is that an apex can be approximated by a triangle, with area E, as shown in Fig 6. Our algorithm begins with the left valley point of the FAI and scans local minima, computing E, the area between the signal (blue curve) and the line connecting the left valley point to the local minimum under test (red line). Depending on the relative position of the two curves, some areas have positive sign (signal lies above the line) while others have a negative sign (signal lies below the line). When $E > T_a$, a significant apex is detected at the point of maximum height. When $E < 0$, we understand that the signal forms a *valley*, which signals our algorithm to backtrack to the last local minimum and restart the whole process from there. Additionally, false apexes can be rejected based on a minimum length of the two sides around the apex, as well as a maximum ratio of the longest over the shortest side. In this way, although originally we assumed one apex per FAI, we can now detect additional apexes and thus disambiguate merged fingers. In our experiments we normalized $r(n)$ to 1 and then used $T_a = 0.1$.

At the end of the above process, almost all of the false positives have been eliminated. Accuracy can be further improved, by using the connected components in the mask $M_{fingers}$ for final verification. Since each FAI, with its corresponding apex, left and right sides and valleys, should be restricted to a *single* mask component, we can choose the midpoints at each side of the apex and connect them with a line. If the apex corresponds to a real finger, then most of the points on the line (ideally all of them) will belong to the same component, without intersecting background pixels or other components.

3.3. Hand posture recognition

3.3.1. Recognition based on Shape Descriptors

Our first method for posture recognition uses directly the palm's mask, M_0 (Sec. 3.1). More specifically, we use the

palm contour coordinates, $\{x(n), y(n)\}$, form the complex 1D-signal $z(n) = x(n) + jy(n)$, take its Discrete Fourier Transform (DFT) $F_k = \sum_{n=0}^{N-1} z(n)\mathrm{e}^{-j2\pi kn/N}$ and keep only the $2P$ coefficients corresponding to $k \in [-P, P] - \{0\}$. Finally, we keep only the magnitudes $|F_k|$, normalized by $|F_0|$. This representation, known as Fourier Descriptors (FD), is invariant to translation, rotation, scaling and choice of initial boundary point [19]. Instead of $z(n)$, Kulshreshth *et al.* [1] used the signal $r(n)$ (Eq. 1). Its main advantage is that we now need only half the number of coefficients, since DFT is symmetric for real signals. For completeness, we considered and evaluated both approaches, using $P = 8$, since we didn't notice any significant improvement with larger values. Finally, posture recognition is done using a standard Nearest Neighbour (NN) classifier.

3.3.2. Recognition based on Finger Information

Information about the fingers is also valuable for the recognition process and can be used in two different ways. First, to reduce the search space size, since we can consider only those training postures with the same number of fingers. We named these approaches $FD^*(z)$ and $FD^*(r)$, corresponding to the previously mentioned $FD(z)$ and $FD(r)$. Second, we propose some *global* features to describe a posture, based only on the fingers (as derived from the FAI analysis). More specifically, we describe a finger using its size characteristics (height, width) and its relative distance from the leftmost finger. Finally, we concatenate all this information for multiple fingers into one feature vector and perform Nearest Neighbour classification. We refer to this finger-characteristics-only method as "Fingers" in Table 1.

4. EXPERIMENTAL RESULTS

4.1. Experimental setup

For our experiments we used the dataset of [5], which contains an alphabet of 10 different postures, with 10 examples from 10 different persons, i.e. 1000 postures in total. As we stated above, we used luminance only for face detection and depth information for hand and finger detection. For our recognition experiments, we used 10-fold cross-validation in a user–independent mode, i.e. at each round we used 900 postures (9 people $\times$ 10 categories $\times$ 10 examples) for training and the rest 100 examples for testing. Our method was implemented in a Matlab environment, executing on a standard PC system.

4.2. Hand–palm separation and finger segmentation

Hand–palm separation was visually confirmed, with results of similar quality to the example shown in Fig. 3-e. The only exception was one case (out of 1000) where the resulting mask missed the thumb and placed the hand–palm separation line incorrectly, in the middle of the palm. Regarding finger detection, we considered as correct cases those where the expected number of fingers - known a priori by the posture class - was returned, with 996 correct results out of 1000 postures. A more detailed evaluation would consider additional finger information, but we didn't have any reference for that.

Method	Rec. Accuracy (%)
Ren *et al.* [5]	93.9%
$FD(z)$	98.5%
$FD(r)$	97.1%
$FD^*(z)$	99.5%
$FD^*(r)$	99.1%
Fingers	96.3%

Table 1. Recognition accuracy for various methods. The FD^* methods use Fourier Descriptors and achieve search space reduction based on the number of fingers.

4.3. Hand posture recognition

As we see in Table 1, $z(n)$ is a better choice than $r(n)$ as the basic signal in the Fourier Descriptor representation. Zhang and Lu [20] actually reported the opposite result, but they worked on highly complicated artificial shapes, while we currently consider only hand shapes.

While the results of $FD(z)$ and $FD(r)$ are very good, we also experimented by considering only training postures with the same number of fingers as those detected in the query posture. This reduction gave higher posture recognition accuracies, mostly due to the very high accuracy on the number of fingers detected ($996/1000$), as shown by lines $FD^*(z)$, $FD^*(r)$ in Table 1. Finally, we also evaluated our global features based on the fingers characteristics ("Fingers"), which proved better than [5] but not as good as the four FD-based methods mentioned previously. Regarding computational complexity, 1 search with $FD(z)$ required 136.1 μs and 80.5 μs with $FD^*(z)$. Please note that, these figures include significant computational overhead by the environment, and we normally expect optimized implementations to run much faster on a commercial device.

Overall, our results are better than those in [5] and suggest that local features, such as Fourier Descriptors, can be very informative and sufficient for hand posture recognition, at least for certain vocabularies. On the other hand, global features, such as the number of fingers, can be used to improve the results through search space reduction.

5. CONCLUSIONS

In this work we proposed a novel approach for finger detection and hand posture recognition. Our experiments achieved state-of-the-art results on a challenging dataset, using simple shape features. Our goals for future work include integration of our method in a real-time system, experiments on other datasets and fusion of depth and color information.

6. REFERENCES

[1] A. Kulshreshth, C. Zorn, and J. LaViola, "Real-time markerless kinect based finger tracking and hand gesture recognition for hci," in *Proc. IEEE Symposium on 3D User Interfaces*, 2013, vol. 1, pp. 187–188.

[2] A.D. Bagdanov, A. Del Bimbo, L. Seidenari, and L. Usai, "Real-time hand status recognition from rgb-d imagery," in *21st Int'l Conf. on Pattern Recognition (ICPR), 2012*, 2012, pp. 2456–2459.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[4] A. Kurakin, Z. Zhang, and Z. Liu, "A real time system for dynamic hand gesture recognition with a depth sensor," in *20th European Signal Processing Conf. (EUSIPCO)*, 2012, pp. 1975–1979.

[5] Z. Ren, J. Yuan, and Z. Zhang, "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in *ACM Multimedia (ACM MM 11)*. 2011, pp. 1093–1096, ACM Press.

[6] C. Keskin, F. Kirac, Y.E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *IEEE Intn'l Conf. on Computer Vision Workshops (ICCV Workshops)*, 2011, pp. 1228–1234.

[7] C. Keskin, F. Kirac, Y.E. Kara, and L. Akarun, "Randomized decision forests for static and dynamic hand shape classification," in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012, pp. 31–36.

[8] L. Billiet, J. M. Oramas, M. Hoffmann, W. Meert, and L. Antanas, "Rule-based hand posture recognition using qualitative finger configurations acquired with the Kinect," in *2nd Intn'l Conf. on Pattern Recognition Applications and Methods*, 2013, pp. 1–4.

[9] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect," in *British Machine Vision Conf. (BMVC)*, 2011, pp. 101.1–101.11.

[10] P. Doliotis, V. Athitsos, D. Kosmopoulos, and S. Perantonis, "Hand shape and 3d pose estimation using depth data from a single cluttered frame," in *Int'l Symposium on Visual Computing (ISVC)*. 2012, vol. 1, pp. 148–158, IEEE.

[11] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, "Parametric correspondence and chamfer matching: two new techniques for image matching," in *5th Int'l Joint Conf. on Artificial Intelligence (IJCAI)*, 1977, vol. 2, pp. 659–663.

[12] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *5th Conf. on Computer Vision and Pattern Recognition (CVPR)*. vol. 1, pp. 511–518, IEEE.

[13] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[14] L.G. Khachiyan, "Rounding of polytopes in the real number model of computation," *Journal Mathematics of Operations Research*, vol. 21, no. 2, pp. 307–320, 1996.

[15] Anye Li, "Approximate lowner ellipsoid," 2008, link: http://www.mathworks.com/matlabcentral/fileexchange/21930.

[16] T. Birdal, "Maximum inscribed circle using voronoi diagram," 2011, link: http://www.mathworks.com/matlabcentral/fileexchange/32543-maximum-inscribed-circle-using-voronoi-diagram.

[17] S. Poularakis, A. Briassouli, and I. Kompatsiaris, "Full action instances for motion analysis," in *10th Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2009, pp. 37–40.

[18] Z. Ren, J. Yuan, C. Li, and W. Liu, "Minimum near-convex decomposition for robust shape representation," in *IEEE Int'l Conf. on Computer Vision (ICCV)*, 2011, pp. 303–310.

[19] R.C. Gonzalez and R.E. Woods, *Digital Image Processing (3rd Edition)*, Prentice-Hall, Inc., 2006.

[20] D. Zhang and G. Lu, "A comparative study of fourier descriptors for shape representation and retrieval," in *Proc. of 5th Asian Conf. on Computer Vision (ACCV)*, 2002, pp. 646–651.