Efficient Hand Detection on Client-server Recognition System

Victor Chernyshov
Department of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Moscow, Russia

Keywords: Hand Biometrics, Client-server System, Continuous Skeletons, Hand Detection, Hand Shape, Finger Knuckle Print.

Abstract: In this paper, an efficient method for hand detection based on continuous skeletons approach is presented. It showcased real-time working speed and high detection accuracy (3-5% both FAR and FRR) on a large dataset (50 persons, 80 videos, 2322 frames). This makes the method suitable for use as a part of modern hand biometrics systems including mobile ones. Next, the study shows that continuous skeletons approach can be used as prior for object and background color models in segmentation methods with supervised learning (e.g. interactive segmentation with seeds or abounding box). This fact was successfully adopted to the developed client-server hand recognition system — both thumbnailed colored frame and extracted seeds are sent from Android application to server where Grabcut segmentation is performed. As a result, more qualitative hand shape features are extracted which is confirmed by several identification experiments. Finally, it is demonstrated that hand detection results can be used as a region of interest localization routine in the subsequent analysis of finger knuckle print. The future research will be devoted to extracting features from dorsal fingers surface and developing multi-modal classifier (hand shape and knuckle print features) for identification problem.

1 INTRODUCTION

Rapid progress in mobile technologies naturally causes the development of personal biometrics systems based on tablets and smartphones. Together with iris and fingerprints, hand is one of the most promising biometrics modalities. Characteristics of modern devices (performance, camera quality, wireless communication capabilities) make it possible to realize various types of architecture of hand authentification/identification application (Franzgrote et al., 2011):

1. mobile device based (the full cycle of processing is on board of a mobile device),
2. “truly” client-server architecture (a mobile device only captures images/videos and sends to a server),
3. hybrid (the processing stages are shared by a mobile device and a server).

Apparently that schemes 1. and 3. suppose the client-side of the application to be as fast as possible.

The main objective of this paper is to introduce fast and reliable method for hand detection, which can be effectively used in real-time client-server identification system. The proposed method (Section 3) is based on continuous skeletons approach comprehensively described in the papers (Mestetskiy and Semenov, 2008), (Mestetskiy et al., 2011).

Another motive of the study is to show the outlook of applying skeleton representation for setting appearance models in segmentation methods with supervised learning (Section 4.1). Accurate segmentation is an integral part of majority shape features extraction methods.

It is also demonstrated that skeleton hand shape representation can be used for extraction regions containing finger knuckle prints (Section 4.2). This is the initial step to start the local texture analysis of dorsal fingers surface.

Algorithms described in this work are a part of ongoing project “Mobile Palm Identification System” (MoPIS), earlier introduced in (Chernyshov and Mestetskiy, 2013).

We remind of the fact that MoPIS system has a client-server architecture where the client is an application for Android-based mobile devices, and the server is implemented with a help of Debian GNU/Linux distributive, Nginx web-server and programming language C++. Android application uses frames from a high-resolution video camera as input.

Let us highlight the main ideas which have led to the development of the proposed method.

While processing a frame we usually conduct two consequent subroutines: hand detection in the frame (at client-side) and, in the case of a positive outcome, detailed analysis of this frame with a hand (at server-side). Due to the network restrictions we can send only several frames to the server during the identification session.

That's why hand detection procedure should approve for the further analysis as few images guaranteed to be unfit as possible (e.g. we need a low false positive rate). And at the same time, it should process a video stream from the camera at wide range of mobile devices in real-time. Thus, the required method has to meet very strict requirements both to the quality of recognition and performance.

There have been some research to detect hand using AdaBoost-based methods with promising results in accuracy and speed (Kölsch and Turk, 2004), (Fang et al., 2007), (Xiao et al., 2010) suitable for use in unconstrained environments (various kinds of lighting, diverse background, etc.). But these methods generally need exhaustive classifier training and a large dataset including samples with different rotations and scaling. Also such methods don't explicitly utilize any hand geometrics like mutual disposition of fingers or their proportions, making difficult to separate "bad" hands (e.g. with partially "glued" fingers — Fig. 3; such sample is ineligible for the shape analysis procedure) from "good" ones. Another approach is to use skin color based detection (Elgammal et al., 2009), (Vezhnevets et al., 2003) but it is unreliable because of sensitivity to lighting conditions and messing with skin-colored objects. Optical flow methods (Sobral, 2013) demonstrate good results for stationary cameras and permanently moving objects, so, cann't be directly used in our case without improvements.

The rest of the paper is organized as follows. Equipment and collected dataset are described in Section 2. The hand detection procedure is fully presented in Section 3. Further server-side image processing is given in Section 4. Next, the experiment results are introduced and discussed in Section 5. Finally, Section 6 shortly concludes the paper.

2 DATA AND EQUIPMENT

During the research we collected 80 short videos of hands of 50 different people (1-3 video for each person, the back side of the right hand was captured). All videos were taken using cameras of mobile devices. After that they were decomposed into frames (each 5th frame was used), which were saved as graphic files (*.jpg or *.bmp). As a result, we got 2322 images.

Important notice: in our work we consider the assistance of participants — the reasonable person's intention is to be correctly and quickly recognized by the identification system. Thus, all cases of cheating (and corresponding videos as well) are excluded from consideration.

To form a qualitative dataset (as to get adequate results from using MoPIS) one should follow the recommendations given below while capturing videos:

1. The videos should be recorded using a mobile device with a camera matrix resolution at least 1.3 Megapixels (Mp), which produces video files with a resolution of 640*480 and more and frequency of 15 frames/second or higher. Choosing a low or middle resolution camera gives a little chance to extract any promising texture features, though shape analysis stays rather effective. Preference should be given to high-end devices with hardware autofocus support. The optimal duration is 3-6 seconds. For video recording one can utilize an Android application which is a part of MoPIS, and also similar applications. During the experiments we mainly used the smartphone LG G2 with 13Mp front camera (supports autofocus and optical stabilization), and recorded HD video (1920x1080 or 1280x720 resolution and 30 frames/sec frequency) with a help of MoPIS Android application. Also, some data was captured using Samsung Galaxy Note 10.1 tablet (Fig. 1).
2. The background should be black or dark (homogeneity is not necessary) otherwise there may be problems with binarization by Otsu (Otsu, 1979) (and therefore with a hand detection in the frame). This in turn will affect the quality of the further analysis of the hand. So, we used black homogeneous cloth as a background in our experiments.
3. The camera is recommended to be stable, only the tested hand should move. Modern mobile devices have good optical stabilization system, so, there is no need in a tripod or a holder to fix the position. Nonetheless, some of the videos of our dataset were made with a help of a tripod (Fig. 1).
4. To increase the variability of frames obtained from the video the probationer should slowly move his fingers (bring together and separate them) in the horizontal plane. One should avoid sudden movements. To improve the representativeness of the dataset it's highly advisable to record a few videos from each hand in different lighting conditions.

5. Experiments should be carried out in good diffused light (artificial or natural), and in its absence we recommend to turn on the built-in mobile flash.

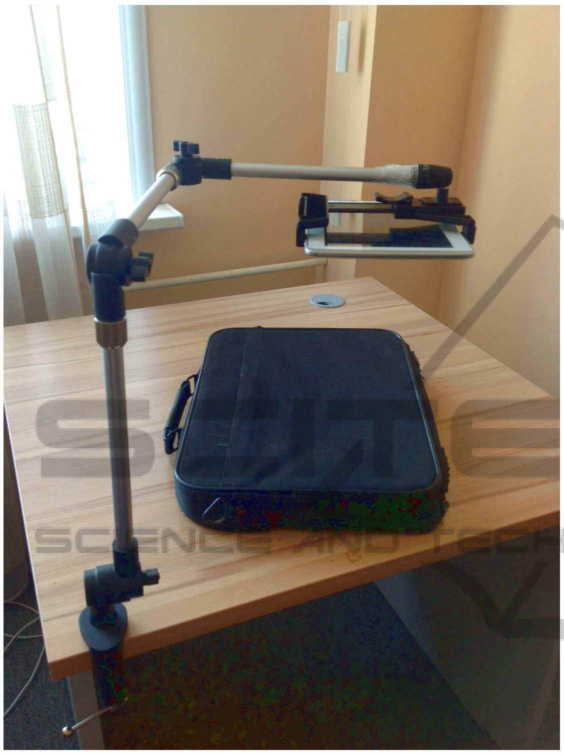
Figure 1: MoPIS experimental setup. Based on Samsung Galaxy Note 10.1 (5Mp front camera).

3 CLIENT-SIDE HAND DETECTION

The first thing we need to do after getting a frame is perform a rescaling (factor of $1/2$, $1/3$ or even less; the majority of experiment was done with 640x360 images) — hand detection procedure is a real-time routine even at mobile devives.

Next, a binarization is performed and the largest contours are marked out for further analysis. We have chosen Otsu binarization (Otsu, 1979) as a primary method (Fig. 5). The good balance of speed, accuracy and versatility was proved by the numerous experiments. Of course, one should follow the recommendations described in the section 2 to get acceptable quality but really it doesn't significantly limit the range of applicability of application in case of "indoor" usage. Background and lighting restrictions can be weakened by applying advanced segmentation routines or utilizing some specific hardware (e.g. depth cameras like Kinect or Intel RealSense which are likely to be integrated in future desktop and mobile devices).

In the Fig. 6 you can find the visualized output of the detection procedure. Though the boundary of hand (i. e. the result of countour traversing after binarization, green line) is a little bit saw-edged, the whole detection procedure worked out correctly.

3.1 Skeleton Construction

As it was mentioned in Introduction, the detection procedure heavily uses continuous skeleton of a binarized image. Skeleton representation of an object supposed to be a hand is built and afterwards regularized using the same logic as described in (Mestetskiy et al., 2011). Both internal and external skeletons (see Fig. 6 — they are drawn via pink and yellow, blue and turquoise segments respectively) of hand shape are used for the further analysis. The hand detection problem in our case consider low false positive rate, so, we have to develop an extended check "valid hand/not valid" that is introduced in the next subsection.

3.2 Hand Validation

All the checks provided in this section are applied sequentially. If we get the false detection result then the shape is excluded from the following analysis. If all tests are passed the shape is supposed to be valid hand.

First of all the internal skeleton is examined to the depth from the vertices of degree 1. It searches the branches starting at vertices of degree 1 and ending at the *root* (the center of maximal circle with radius R_{max} inscribed in the shape; we also call it *center of hand*). Thus, we obtained *finger branches candidates*. A branch failed to pass some check is eliminated from the following processing. If at a certain point we get less than 5 branches we consider the false detection result. Similarly, if in the end of validation we have more than 5 branches.

Radius of maximum inscribed circle is associated with any point of skeleton. Function $R(x)$ that maps points of skeleton to radiuses of maximum inscribed circles is called the *radial function*. We calculate the values of the radial function (or its interpolation) in 30 cue points evenly located from the tip to the end of the finger branch candidate. After it, we apply the linear mapping $X \times R(X) \to [0,1] \times [0,1]$. The obtained function is named *normalized branch radial function*. Using train dataset we can easily calculate the lower and upper bounds of this function in the cue points and than slightly expand this "tube" called *finger boundary corridor*. We consider the general boundary corridor for all fingers. To perform the *boundary coridor check* one just need to sure that normalized branch radial function is located inside the boundary corridor (Fig. 2).

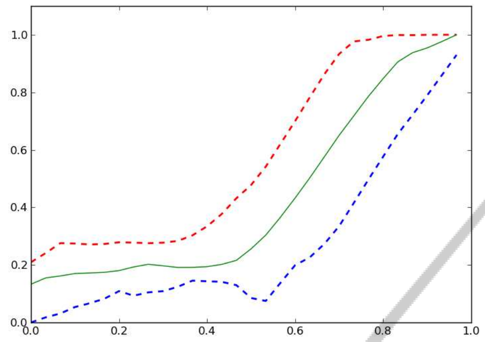

Figure 2: Frame 1562.0015. Doted lines are bounds of finger boundary corridor. Green solid line — normalized branch radial function of the pointing finger ($id = 1$).

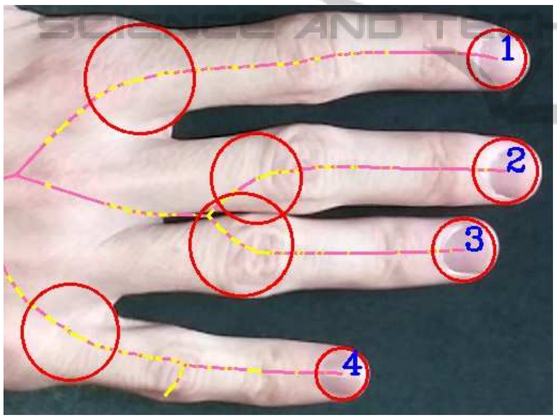

Figure 3: Frame 1237.0010, cropped. Fingers sticked together.

The next step is to determine the branch top and bottom nodes corresponding to the tip and base of a potential finger. This is done similarly to the method given in (Mestetskiy et al., 2011). The line connecting top and bottom nodes is called the *axis* of the branch. The total length of branch edges between top and bottom nodes is called *length* of a potential finger. After it, we apply several threshold checks (hereinafter, all the values are obtained from train dataset and all the distances are normalized to R_{max}). We check whether the bottom node of a finger is at distance from the root which is greater than the threshold value ε_1. The same checking is applied for the top node (threshold ε_2).

Further, for a given potential finger we delete all the fingers which bottom nodes are located inside the bottom node circle of this finger on condition that they are shorter than the given one.

Next, we come to *triples* check. The fingers are arranged in the order of the contour traversal. Triples of adjacent fingers are examined. For each triple the middles of the first and the third fingers are connected. This segment is supposed to intersect the second finger (the point of intersection should lie within both segments). Such a heuristics successfully works because of blob structure of a hand and small axes between neighboring fingers.

To the present moment we discarded the vast majority of non-fingers branches. We consider, that the most distant to the others bottom node belongs to the thumb. So, we arrange the fingers in the order of the contour traversal starting from the thumb ($id = 0$ is assigned).

Though the validation process above is strict and reliable, it doesn't cope with fingers sticked together (Fig. 3). That's why a *median* check was implemented. We consider 4 fingers (without a thumb). The euclidian distances $\rho_i, i = 1\ldots4$ from the bottom node of the fingers to the root are ordered by ascending, as a reference value we select the second value d_2 from the beginning. After that, for each finger we count normalized deviations from the reference value: $\eta_i = |\rho_2 - \rho_i|/R_{max}, i \in \{1,3,4\}$. It is verified that these deviations are less than the threshold ε_3.

4 SERVER-SIDE HAND ANALYSIS

4.1 Hand Shape Processing

One of the main purposes of a MoPIS client-server architecture is to remove computationally heavy procedures (like “clever” Markov Random Fields based segmentation methods that are usefull for more accurate shape features extraction) from Android application to server. At the same time, when the detection procedure is completed we already know the internal and external skeletons representation of a hand. And it can be easily adopted for using as prior for object and background color models in advanced segmentation methods with supervised learning (e.g. interactive segmentation with seeds or abounding box).

As seeds we use circles with centers at the vertices of the skeleton graph (Fig. 7) located inside the bounding box (Fig. 6). For an internal skeleton only fingers branches and the root circle are used. For an external skeleton only 4 branches lying between the fingers axes are considered. Seeds radiuses are the values of the radial functions in the skeleton vertice multiplied by a coefficient (we use 0.9 for internal skeletons, 0.7 for external). Also we limit the minimal external seed radius. Finally, we utilized the region inside bounding box and the seeds as input for graph cut driven method (Tang et al., 2013) and got really good segmentation (Fig. 8). The boundary is correct and smooth — shape is ready to use for features extraction. It should be noted that there is a quantity of supervised segmentation methods that can be used jointly with seeds, Grabcut segmentation was selected because of its accuracy and speed. Experimental results are discussed in Section 5.2.

4.2 Finger Region Processing

Although hand shape features can be rather informative in itself (Yörük et al., 2006), some authors propose effective multi-modal hand recognition methods for example using palmprint and shape (Kumar and Zhang, 2006), (Kozik and Choras, 2010). Our hand recognition system MoPIS is designed to work with two modalities — finger knuckle prints and shape. Choosing back side of a hand is determined by our experience while collecting dataset — people find more comfortable putting palmprint side of a hand down on a table.

Features extraction from finger knuckle patterns assumes the presence of high-resolution source images (Kumar, 2012) or even special capturing devices (Zhang et al., 2010). Moreover, some texture analysis methods require massive calculations and cannot be directly used in per-frame processing at client-side device. On the other hand, sending heavy full-hand images to server certanly is not the best option due bandwidth and time limits. A logic alternative is to perform a fast localization of regions with finger knuckle patterns at client and then transmit to server only these regions.

Hand detection procedure 3 gives us all necessary details to produce accurate and fast fingers localization. So, consider the part of a finger branch locating between the top and bottom finger nodes. The futher costruction of corresponding finger polygon and its bounding box is rather trivial (Fig. 9). Since at client-side hand detection algorithm usually operates with resized frames we apply a simple linear transform to polygon and bounding box coordinates to get the finger region in the source frame (Fig. 10). Further, extracted region in high-resolution is asynchronously sent to server. The common size of such regions in jpeg format is 15-30kB for 1920x1080 source frame while the frame itself is 250-500kB. Thus, hand detection results can be used as a region of interest localization routine in the subsequent analysis of finger knuckle print.

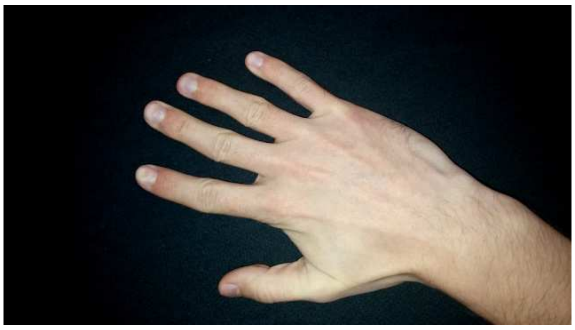
Figure 4: Frame 1562.0015. Source image.

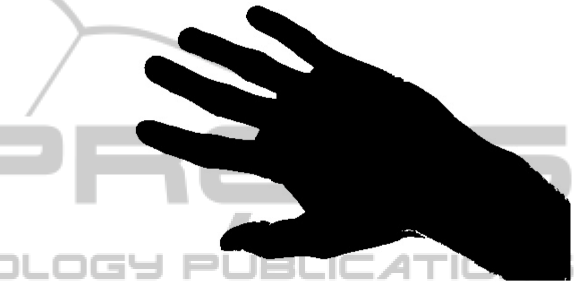
Figure 5: Frame 1562.0015. Otsu binarization.

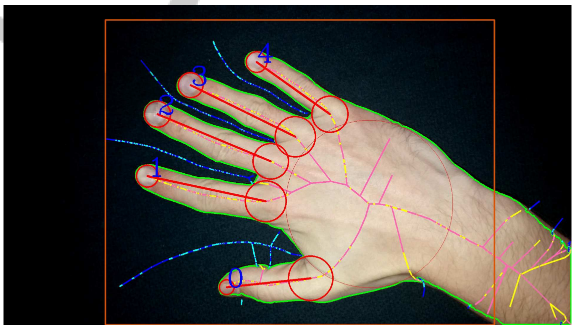

Figure 6: Frame 1562.0015. After applying detection procedure. Nonlinear Voronoi sites are yellow and turquoise “segments”, linear — red and blue segments. Bounding box of the hand is orange.

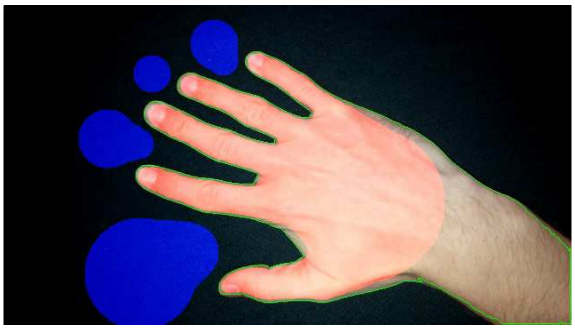
Figure 7: Frame 1562.0015. Hand (red) and background (blue) seeds.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

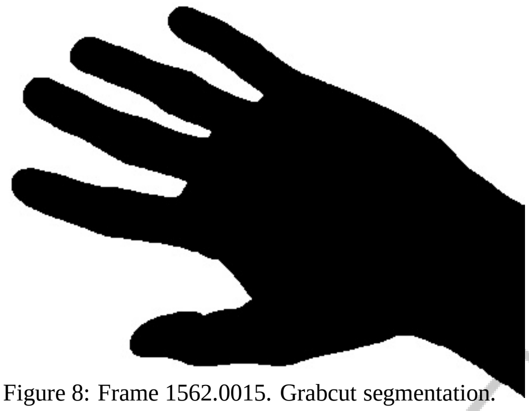
Figure 8: Frame 1562.0015. Grabcut segmentation.

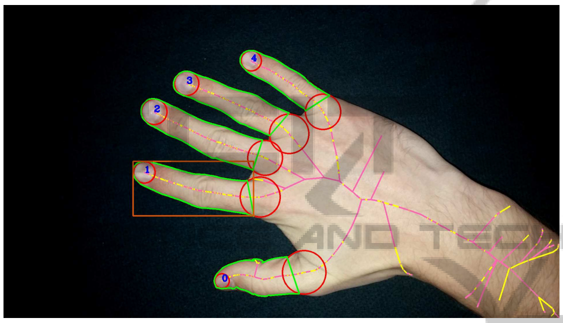
Figure 9: Frame 1562.0015. Finger polygons are green. Bounding box of the index finger is orange.

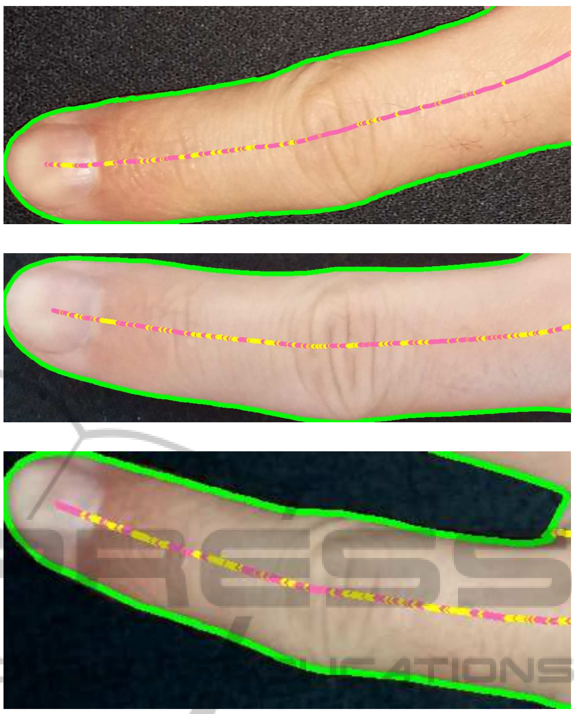
Figure 10: Regions with index fingers of the same person (id=18) extracted from different hand images. In the top figure region is obtained from standard photo with resolution 4160x3120, in the middle — from 4160x3120 HDR photo, in the bottom — from 1920x1080 video frame 1562.0015.

5 EXPERIMENTS

All frames from the dataset (see Section 2) were manually marked: either with valid hand or without.

5.1 Detection Testing

The testing scheme for detection was organized as follows. We made 3 random partitions of our frame dataset into train and test datasets, containing frames of 20 and 30 different people respectively. Train dataset was used to calculate threshold statistics, test was utilized for quality and speed estimation. All frames with resolution 1280x720 and 1920x1080 were resized to 640x360. The experiments were run on a laptop with Intel Core i7 2.4 GHz processor without using multiprocessing routines. According to the time profiler the most time consuming procedure was creation of Voronoi diagram — about 60% of computational time. Hand detection results can be found in Table 1. Each row corresponds to a dataset partition. The last column contains time per frame (TPS) values for the whole detection method (summing execution time of binarization, skeleton construction and hand validation).

Table 1: Detection results.

#	FAR, %	FRR, %	errors, %	TPF, ms
1	3.2	4.1	3.4	40.5
2	2.8	4.6	3.3	40.2
3	3.7	5.2	4.1	40.0

Since proposed detection procedure demonstrates low error rates (both false acceptance and false rejection) and real-time processing speed it can be used in hand biometrics systems, e. g. as a part of client-side application. The algorithm is robust to image quality — it works well even with low-resolution images (e. g. with 320x180). The most false acceptance cases are related to hands partially placed in the frame. Also, there are some difficulties with blurred frames though we tried to eliminate quick hand and fingers motions in data acquisition process.

5.2 Segmentation Testing

In Section 4.1 is showed how hand detection results can be adopted for setting appearance models in supervised segmentation methods. Grabcut segmentation (Fig. 8) gives an enhancement over Otsu method (Fig. 5) and it is proven by identification results. We used the same hand dataset and testing scheme as in section above, extracted only shape-based features and run simple 1NN classifier — utilizing segmentation produced by Grabcut's method gave significantly lower identification error rate then the same system powered with Otsu segmentation (Table 2). The second and the third colums contain identification error rates for Otsu and Grabcut based systems correspondively, the last two columns — execution time of the aforementioned segmentation routines.

Table 2: Segmentation results.

#	Otsu, %	Grabcut, %	Otsu, ms	Grabcut, ms
1	12.3	7.7	0.4	2002
2	13.6	8.1	0.4	1922
3	13.2	8.2	0.4	2107

An average size of colored 640x360 hand images used in segmentation experiments is 30-50kB that meets client-server bandwidth limits. This image quality is sufficient for accurate shape features extraction — using instead source images with 1280x720 or 1920x1080 resolution didn't cause any notable identification results improvements. So, server-side Grabcut segmentation combined with client-side seeds produces solid base for the further hand shape features extraction.

6 CONCLUSION

In this paper, a fast and reliable method for hand detection based on continuous skeletons approach is presented. It showcased real-time working speed and high detection accuracy (3-5% both FAR and FRR) on a large dataset (50 persons, 80 videos, 2322 frames). This makes the detection method suitable for use as a part of modern hand biometrics systems including mobile ones.

Next, the study shows that continuous skeletons approach can be used as prior for object and background color models in segmentation methods with supervised learning. This fact was successfully adopted to the developed client-server hand recognition system — both thumbnailed colored frame and extracted seeds are sent from Android application to server where Grabcut segmentation is performed. As a result, more qualitative hand shape features are extracted which is confirmed by several identification experiments.

Finally, it is demonstrated that hand detection results can be used as a region of interest localization routine in the subsequent analysis of finger knuckle print. Applying finger knuckle images for personal identification in context of client-server hand recognition system is a challenging task for the further research. Also, significant work should be done on constructing reliable multi-modal (hand shape and knuckle print features) classifier to improve identification results.

ACKNOWLEDGEMENT

The author thanks the Russian Foundation for Basic Research for the support on this study (grant 14-01-00716).

REFERENCES

Chernyshov, V. and Mestetskiy, L. (2013). Mobile machine vision system for palm-based identification. In *Proceedings of the 11th International Conference "Pattern Recognition and Image Analysis: New Information Technologies" (PRIA-11-2013)*, volume 2, pages 398–401.

Elgammal, A. M., Muang, C., and Hu, D. (2009). Skin detection. In *Encyclopedia of Biometrics*, pages 1218–1224.

Fang, Y., Wang, K., Cheng, J., and Lu, H. (2007). A real-time hand gesture recognition method. In *Proceedings of the 2007 International Conference on Multimedia and Expo (ICME 2007), Beijing, China*, pages 995–998. IEEE.

Franzgrote, M., Borg, C., Tobias Ries, B., Büssemake, S., Jiang, X., Fieleser, M., and Zhang, L. (2011). Palmprint verification on mobile phones using accelerated competitive code. In *2011 International Conference on Hand-Based Biometrics (ICHB)*, pages 124–129. IEEE.

Kölsch, M. and Turk, M. (2004). Robust hand detection. In *In International Conference on Automatic Face and Gesture Recognition (to appear), Seoul, Korea*, pages 614–619.

Kozik, R. and Choras, M. (2010). Combined shape and texture information for palmprint biometrics. *Journal of Information Assurance and Security*, 5:60–66.

Kumar, A. (2012). Can we use minor finger knuckle images to identify humans? In *Proceedings of the IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS 2012)*, pages 55–60. IEEE.

Kumar, A. and Zhang, D. (2006). Personal recognition using hand shape and texture. *Trans. Img. Proc.*, 15(8):2454–2461.

Mestetskiy, L., Bakina, I., and Kurakin, A. (2011). Hand geometry analysis by continuous skeletons. In *Proceedings of the 8th international conference on Image analysis and recognition - Volume Part II*, ICIAR'11, pages 130–139, Berlin, Heidelberg. Springer-Verlag.

Mestetskiy, L. and Semenov, A. (2008). Binary image skeleton-continuous approach. In *Proceedings of 3rd International Conference on Computer Vision Theory and Applications (VISAPP)*, volume 1, pages 251–258. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66.

Sobral, A. (2013). BGSLibrary: An opencv c++ background subtraction library. In *IX Workshop de Vis?o Computacional (WVC'2013)*, Rio de Janeiro, Brazil.

Tang, M., Gorelick, L., Veksler, O., and Boykov, Y. (2013). Grabcut in one cut. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 1769–1776, Washington, DC, USA. IEEE Computer Society.

Vezhnevets, V., Sazonov, V., and Andreeva, A. (2003). A survey on pixel-based skin color detection techniques. In *Proceedings of the GraphiCon 2003*, pages 85–92.

Xiao, B., Xu, X.-m., and Mai, Q.-p. (2010). Real-time hand detection and tracking using lbp features. In Cao, L., Zhong, J., and Feng, Y., editors, *Advanced Data Mining and Applications*, volume 6441 of *Lecture Notes in Computer Science*, pages 282–289. Springer Berlin Heidelberg.

Yörük, E., Konukoglu, E., Sankur, B., and Darbon, J. (2006). Shape-based hand recognition. *IEEE Transactions on Image Processing*, 15(7):1803–1815.

Zhang, L., Zhang, L., Zhang, D., and Zhu, H. (2010). Online finger-knuckle-print verification for personal authentication. *Pattern Recognition*, 43(7):2560–2571.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS