The following publication K. -P. Leung and N. F. Law, "An efficient automatic palm reading algorithm and its mobile applications development," 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), Hong Kong, China, 2016 is available at https://doi.org/10.1109/ICSPCC.2016.7753706.

An Efficient Automatic Palm Reading Algorithm and its Mobile Applications Development

Kwan-Pui Leung and N.F. Law
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University
Hong Kong

Abstract— **Palm reading is a traditional practice in China for a few thousand years to tell ones' fortune. Currently, there is a lack of mobile applications that allow palm reading to be done automatically and efficiently. This study aimed at developing an effective palm reading algorithm which can run in an Android platform efficiently. OpenCV and Java were used for the implementation. Our palm reading algorithm uses an adaptive thresholding approach to segment the palm image from the background, extract the fingers and calculate their length, extract the three principal palm lines in which regression is applied to produce connected and continuous palm lines. The algorithm was implemented as an Android application. Results showed that the algorithm can be run within 2 to 4 seconds, and the automatic palm reading can be done on mobile platforms accurately. The study enriched existing market of mobile applications that aim at palm reading. With successful implementation of such platform, and by collecting more personal information of the users, such as personality and health status, this application can be applied for future research on the prediction of personality and health.**

Index Terms—**Palm reading, personality, health, mobile, automatic, Android, OpenCV, Java**

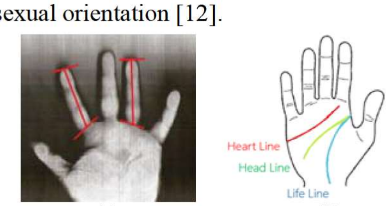
I. Introduction

Palm reading has long been believed to be a good means for fortune telling in the Chinese culture [1]. Information such as the length of fingers, positions and slopes of palm lines are said to be related to ones' personality and health according to the traditional Chinese "Feng Shui" (fortune telling) theories [2]. However, the concepts and theories for fortune telling based on palm reading are indeed complicated. For instance, Leung [3] suggested 100 distinct case studies of palm lines for beginners to learn, which demonstrates that palm reading is difficult to be learnt or to quantify. Therefore, this study aimed at developing computational algorithms that allow palm reading to be performed automatically and conveniently in an Android application. There are three reasons to support the importance of developing such an application. First, according to Liu [4], technological advance has led to increasing demands and markets for online fortune telling. Users are willing to pay for getting immediate results of their palm reading. Second, palm reading has the potential to predict personality and health. Although palm reading may be viewed as "pseudoscience", a number of recent studies in the Western settings suggested that information from palms predicts personality or health. Nevertheless, most studies focus on the length of fingers or the index finger to ring finger ratio (2D:4D ratio), but not the information about the complex palm lines. With the development of an application, these data could be quantified and would have strong potential to accurately predict personality and health. Third, while there were previous studies which developed algorithms to compute the length of fingers and extract information of palm lines from a palm image manually, implementing such algorithms in mobile platforms are scarce. If palm reading algorithms are implemented as application on mobile devices, the portability of mobile devices would greatly enhance the ease of future development on palm reading. As the computational power of mobile devices is lesser than that of computer, this study also aimed at developing palm reading algorithms that can run at a reasonable speed on mobile applications. Since a lot of image processing operations have to be carried out under the Android platform, the feasibility of doing this in terms of complexity, battery and memory requirement, were also studied.

II. Background

A. Theories of Palm Reading

The length of index finger to the length of ring finger is known as 2D:4D ratio. Fig. 1(a) shows the 2D:4D ratio on a palm. Past studies suggested that the length of fingers and the 2D:4D ratio could predict a series of psychological and health constructs such as the amount of androgens [6], the risks of getting diseases, aggressiveness [7], assertiveness [8], agreeableness [9], likelihood of getting psychological disorder [10, 11] and sexual orientation [12].



(a) (b)

Fig. 1. (a) 2D:4D ratio [5] and (b) Principal palm lines [13]

Besides the 2D:4D ratio, it is believed that the three principal lines (heart line, head line and life line) of the palm shown in Fig. 1(b) serve different functions [3,13-15]. In particular, the heart line predicts ones' emotion and love, the head line predicts ones' intelligence while the life line predicts ones' health status. It is also believed that the lengths, slopes and positions of the three lines relate to different personalities and health.

B. Past Studies on the Computational Algorithms of Palm Reading

Nigam, Yadav and Thakur [16] proposed to use Canny edge detector to detect the edges of the palm and then use Hough transform to derive the outline of the palm. Nevertheless, it is difficult to locate the fingers accurately, as the palm images have to be matched with those in the database. Sigurdsson and Wong [17] proposed to use a boundary tracking algorithm to outline the palm. Nevertheless, this algorithm only allows one finger to be tested at a time. When all the fingers are to be detected, the running speed is slow due to the number of iterations and mathematical operations required. Sandnes [5] explored the feasibility of using smart phones to measure 2D:4D ratio by first segmenting the palm from the background using thresholding and binarization, and then calculating the change of slopes of the lines around the fingertips to locate the fingers. However, the accuracy might be affected by the orientation of the palm in the photo.

Chung [18] proposed to use pre-processing operations including thresholding, median filtering, histogram equalization, dilation and erosion to segment the palm from its background. The canny edge detector is then applied in which the edge detection should be stopped when a certain number of edge points has been detected. The limitation of this method was that the detected edge points might be disconnected and no continuous palm line could be formed. The starting point for each principal palm line is arbitrarily pre-defined, yet, the clustering of palm line depends on the starting point.

Lee [19] suggested to use linear regression to improve the efficiency of automatic palm reading. A model that best fits all the given coordinate points with a minimized error could be formulated using linear regression, and this equation could be used to connect the palm lines. The findings suggested that such algorithm was efficient while the error rate was small.

C. Aims of the Study

The main objective of this study is to develop an effective and efficient computational algorithm that can automatically perform palm reading. First, an algorithm that is able to compute the length of fingers, extract palm lines and compute information about the three principal palm lines automatically is proposed. Then, a mobile application (in Android) with a user-friendly Graphical User Interface (GUI) that integrates these algorithms is developed to illustrate the proposed algorithm can be run efficiently in practical applications.

III. PROPOSED PALM READING ALGORITHM

An effective and efficient algorithm is developed for palm reading. First, the palm has to be segmented from the background. Second, the length of fingers is computed, and finally, the three principal palm lines are extracted. The following subsections describe details about these three steps.

A. Segmenting the Palm from the Background

To reliably segment the palm from the background, adaptive segmentation using Otsu's thresholding technique is combined with flood filling. First, the image is converted to the YUV format. Since the mobile development platform is considered, quality of palm image may be affected by illuminations and shadows. We have found that the YUV format gives accurate segmentation results under different levels of illuminations and shadows. The palm color range was proposed as follows [20],

$$Y > 80 \quad 85 < U < 135 \quad 135 < V < 180 \tag{1}$$

Using the fixed thresholds in eq. (1), good segmentation might be obtained for simple background as shown in Fig. 2. However, the performance may not be good for complicated background. Hence, adaptive segmentation using Otsu's thresholding was proposed. In particular, the palm color range was redefined adaptively as,

$$U_{lower} < U < U_{upper} \quad V_{lower} < V < V_{upper} \tag{2}$$

Fig. 2. Segmentation in the YUV model

In the U channel, the lower limit U_{lower} was set to 85 while the upper limit U_{upper} was determined using Otsu's thresholding. In the V channel, the lower boundary V_{lower} was set to the mean intensity value while the upper limit V_{upper} was determined using Otsu's thresholding.

Otsu's thresholding determined a threshold that maximizes the between-class variance of the background and foreground. At each intensity value, the background and the foreground was separated, and the between-class variance was calculated. After iterating through all intensity values, it returned the intensity value that gave the maximum between-class variance as the threshold. The between-class variance was calculated as [21],

$$\sigma^2 = w_b(t)[1 - w_b(t)][\mu_b(t) - \mu_f(t)]^2 \tag{3}$$

where $w_b(t)$ is the weighted sum of background, $\mu_b(t)$ is the mean intensity value of background and $\mu_f(t)$ is the mean intensity value of foreground. After maximizing the between-class variance, the separation between the background and foreground could be optimized, which then enhanced the accuracy of the segmentation. It was found that 80% - 90% of the segmentations work well under plain background while 60% - 70% of the segmentations work well under complex background. Fig. 3(a) and Fig. 3(b) show the segmentation of palm under complex background using fixed and adaptive thresholding respectively. We can see that the adaptive thresholding in eq. (2) provides better segmentation result.

(a) (b)

Fig. 3. (a) Segmentation of palm under complex background using fixed thresholding and (b) adaptive thresholding.

To further filter out the background pixels and produce connected regions, flood filling was performed after thresholding. The starting point was set as the middle of the palm image, and then its 8 directions were scanned as in Fig. 4. If a white pixel was found, it would be joined to the original starting point and became a new starting point itself. The process ended when no more white pixels were connected with the 8 directions of previous starting points.

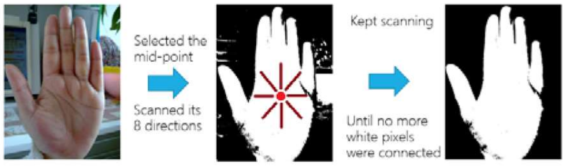


Fig. 4. The flow of flood filling

B. Extracting Fingers and Computing Their Length

After segmenting the palm from the background, fingers were extracted with their lengths computed. This is done by first tracing the contour of the palm and enclosing it by a convex hull. Then convexity defects of the palm were calculated to locate the fingertips and finger gaps.

Contour of the palm means the boundary adjoining the segmented palm. Border Tracing algorithm was used to trace the contour of the palm. Using the function Imgproc.findContours in OpenCV, the contour of the palm was traced accurately as in Fig. 5. The contour could be enclosed by a rectangular box so that the width and height of the palm can be obtained.

Fig. 5. Contour of the palm

Next, the palm was enclosed by a convex hull. Convex hull means the smallest convex polygon that could enclose all the points of the contour. If it is a convex polygon, a straight line connecting any two points cannot exceed the polygon's boundary. Gift Wrapping algorithm proposed by Javis [22] was applied to find the convex hull. First, the contour's vertex was identified. Next, starting from the vertex, the contour was scanned in clockwise or anti-clockwise direction. The furthest and outermost point was chosen in each row of the contour. The convex hull was traced. Fig. 6 shows an example of the convex hull of a palm.

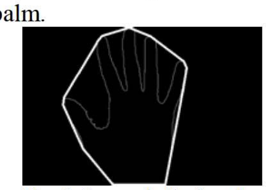
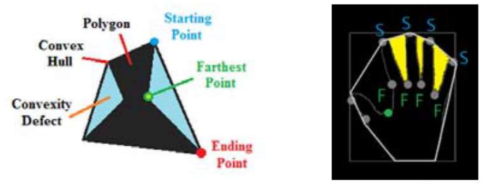


Fig. 6. Convex hull of a palm

Finally, convexity defects of the palm were determined so as to locate the fingertips and finger gaps. As shown in Fig. 7 (a), convexity defects are the area which does not belong to the polygon but being enclosed by the convex hull.



(a) (b)

Fig. 7. (a) Definition of convexity defects and (b) convexity defects of a palm

Fig. 7(b) shows the convexity defects of a palm. The convexity defects correspond to the area enclosed by two fingers. The starting point (S) is the fingertip, the ending point is the next fingertip while the farthest point (F) is the finger gap.

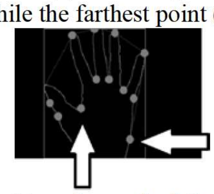


Fig. 8. False positives of convexity defects of a palm

False positives as shown in Fig. 8 were always a problem. In light of this, the following rules were added to filter the false positives:

- All the points must be above the middle of the palm;
- The length between the starting point and the farthest point must be greater than 10% of the palm's height; and
- The inner angle of finger gap as shown in Fig. 9 should be smaller than 120 degrees.

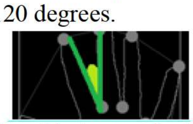


Fig. 9. Definition of the inner angle of finger gap

Mathematically, the inner angle of finger gap is defined as,

$$\theta = \arccos\frac{\mathbf{v}\cdot\mathbf{w}}{\|\mathbf{v}\|\|\mathbf{w}\|} \tag{4}$$

where v is the vector connecting the starting point and the farthest point; and w is the vector connecting the ending point and the farthest point. With the accurate extraction of the starting points and farthest points of the palm, the length of fingers could be obtained as the difference between the starting point and the farthest point.

C. Extracting the Three Principal Palm Lines

First, the region containing the palm lines was cropped. Next, several pre-processing operations were performed to remove noises and sharpen the image. Finally, Canny edge detection was carried out to find the palm lines followed by quadratic regression to connect the palm lines.

To produce reliable results, an elliptical region was first extracted as in Fig. 10.

Fig. 10. Elliptical region of interest

Pre-processing was then carried out. First, noises on the palm image could affect the extraction of palm lines. For example, the wrinkles surrounding the palm lines were not of interest and may be regarded as noises. In light of this, a 3x3 median filter was applied to the image to remove these noises. After applying the median filter, unnecessary details and noises were blurred while the palm lines were preserved. Besides, since mobile devices were used to take the palm image, the image contrast might be low. To enhance the image contrast, histogram equalization was performed on the image. It enhanced the image contrast by allocating the pixel intensities evenly in the range of 0 to 255. Fig. 11 shows a palm image after histogram equalization.

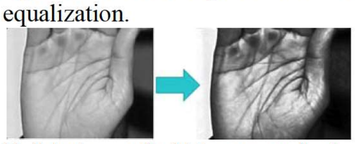


Fig. 11. Palm image after histogram equalization

Unsharp masking was then carried out on the image to sharpen the image. In unsharp masking, the output image g(x,y) is the difference between the input image f(x,y) and the blurred image b(x,y). The blurred image b(x,y) was created by applying a 3x3 Gaussian filter on f(x,y). After subtracting the blurred image from the original image, a sharper version of image was obtained. Afterwards, a 3x3 line sharpening filter defined in eq. (5) was applied to g(x,y).

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{5}$$

Fig. 12 shows a palm image after unsharp masking and line sharpening.

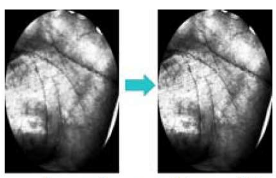


Fig. 12. Palm image after unsharp masking and line sharpening.

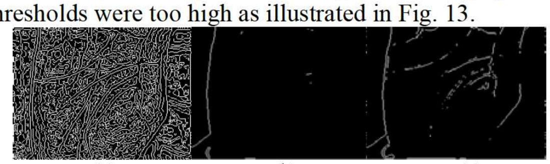
After filtering out the noises, contrast was enhanced and the image became sharper and clearer. Canny edge detection could now be performed to extract the three principal palm lines. In Canny edge detection, edges are the discontinuities in an image representing sharp intensity changes. Palm lines could be regarded as edges because they appear as strong discontinuities in the palm. A pixel would be regarded as edge if its gradient magnitude exceeds a particular threshold. To compute the gradient magnitudes of a pixel, two 3x3 kernels defined as

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{6}$$

were applied on the image. Note that G_x and G_y computes the gradient magnitudes of a pixel in x and y directions respectively. The gradient magnitude (G) of a pixel could be calculated by

$$G = |G_x| + |G_y| \tag{7}$$

If G is greater than an upper threshold, this pixel would be an edge point. If G is smaller than a lower threshold, this pixel would not be an edge point. In between the lower threshold and upper threshold, this pixel would be an edge point only if this pixel is connected to a pixel that is an edge point. Using this scheme, the threshold setting is important. Too many pixels were regarded as edge points if the thresholds were too low. However, too few pixels were regarded as edge points if the thresholds were too high as illustrated in Fig. 13.



(a) (b) (c)

Fig. 13. Results for (a) a small threhold, (b) a large threshold and (c) an appropriate threshold.

In light of this, an algorithm is proposed to determine an appropriate threshold for Canny edge detection.

1. The gradient magnitudes of all edge points were recorded.
2. The maximum gradient was adopted as the threshold to carry out Canny edge detection.
3. If the number of edge points detected is greater than x% of the image, the process stops where x is
   a. 5% for upper threshold, and
   b. 20% for lower threshold
4. Else, the maximum gradient = maximum gradient – 1, and return to step 2.

Fig. 16 shows an example of the detected edge points. 5% to 20% was believed to be a reasonable proportion of palm lines in a palm image.

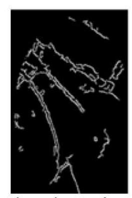


Fig. 14. The edge points detected

After the palm lines were detected, quadratic regression was carried out to connect the palm lines. Since palm lines are in form of parabolas generally, a quadratic parabola that joins a given set of points was used to connect the palm lines. Given a set of points, a parabola in the form of $y = ax^2 + bx + c$ could be formulated. The values of a, b, and c can be computed through the matrix operation as:

$$\begin{bmatrix} \sum x_i^4 & \sum x_i^3 & \sum x_i^2 \\ \sum x_i^3 & \sum x_i^2 & \sum x_i \\ \sum x_i^2 & \sum x_i & n \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \sum x_i^2 y_i \\ \sum x_i y_i \\ \sum y_i \end{bmatrix} \tag{8}$$

Using quadratic regression, the edge points of a palm line can be connected in the form of a quadratic parabola. Users are required to click the initial point of the palm line to start the regression. A window size of 7.5% of the palm's width and 7.5% of the palm's height was put at this initial point. All edge points in the window were stored in an ArrayList. After that, the window would slide to the mean position of the edge points, and the window continued to slide along the palm line until the number of edge points was less than 10, as illustrated in Fig. 15.

Fig. 15. The process of quadratic regression

Since all the edge points have been stored in the ArrayList, quadratic regression could be used to connect the points into the form of parabola, as shown in Fig. 16. Users could also indicate the end points of the palm lines to improve the accuracy of the connection of palm lines.

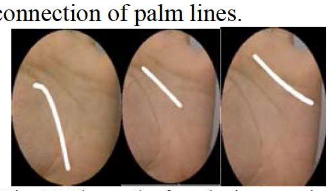


Fig. 16. The result of quadratic regression

Finally, the length of the palm lines was obtained using the Euclidean distance defined as,

$$L = \sqrt{(E_y - S_y)^2 + (E_x - S_x)^2} \tag{9}$$

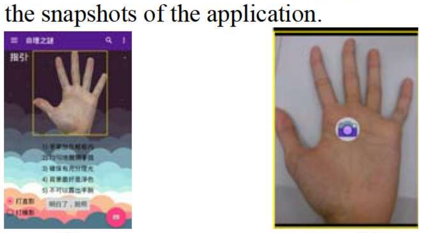
where E_x and E_y are the x-coordinate and y-coordinate of the ending point, and S_x and S_y are the x-coordinate and y-coordinate of the starting point respectively. The slope of the palm lines was obtained using the slope formula as,

$$M = \left| \frac{E_y - S_y}{E_x - S_x} \right| \tag{10}$$

It should be noted that the length of the palm lines was further normalized using the diagonal length of the palm; and absolute value of the slope was taken because the slope of palm lines in the left palm and right palm has an opposite different sign. Note that the starting position and ending position of the palm lines could also be found by comparing with the coordinates of the four fingers calculated earlier.

IV. Experimental results

The palm reading algorithm was implemented as an Android application. A user-friendly Graphical User Interface was developed to capture the palm image, compute the length of the fingers and extract the three principal palm lines. Figure 17 shows the snapshots of the application.



(a) (b)

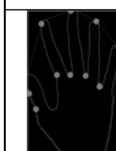
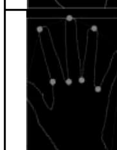
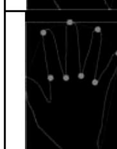
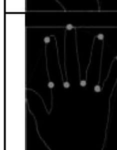
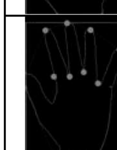
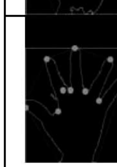
Fig. 17. (a) Tutorial page and (b) the capture screen of the application

A. Study of Accuracy

The application was tested on Sony Xperia. Table 1 summarizes the results of the 2D:4D ratio obtained from the proposed algorithm. The ratio was also compared with the ground truth. We can see that the errors were smaller than 2% of the ground truth. Hence, the ratio can always be obtained accurately. Table 2 shows the results of extracting the three principal palm lines. We can see that the three lines can be fit by using the regression model.

B. Computational Complexity: Speed, Memory and Battery Usage

In terms of computational complexity, we consider the computational speed, the memory required and the battery usage. Under normal conditions, the algorithm takes 2 – 4 seconds. Only 1% of the battery was required for 8 - 10 complete operations. However, the memory usage was quite high. Only 10% of the memory was free. The high memory usage might also be caused by the User Interface (UI) and the background activities.

Results	**2D:4D ratio**		
	Ground truth	**Proposed algorithm**	**% difference**
	0.9375	0.9416	0.44%
	1.0000	1.0154	1.54%
	0.9701	0.9744	0.44%
	0.9589	0.9652	0.66%
	1.1143	1.0943	-1.79%
	1.0000	0.9808	-1.92%

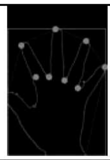
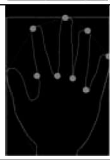
	0.9333	0.9262	-0.76%
	0.9863	0.9912	0.50%

Table 1: The 2D:4D ratio obtained from the proposed algorithm and the ground truth

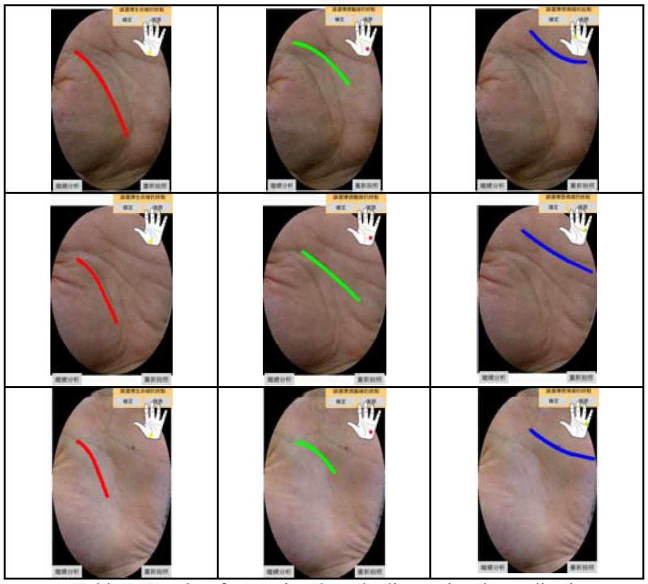


Table 2: Results of extracting the palm lines using the application

V. CONCLUSIONS

This study is among the first to develop a palm reading algorithm that could segment the palm from the background, compute the length of fingers, and extract the three principal lines of the palm automatically. The proposed algorithm was integrated into an Android application with a user-friendly Graphical User Interface (GUI). Experimental results showed that the proposed algorithm produces accurate detection results while the computational requirement is not demanding.

There are a few important implications of this study. First, although there are a lot of image processing operations in our proposed algorithm, the computational speed under the mobile application development platform is acceptable. In contrast to the other available applications in the mobile application market, users do not have to draw their palm lines on screen but to select the starting points only. The errors of detecting palm lines are greatly reduced. Second, this project has the potential to trace the changes of the palm lines over time for further research. Our future works include a large scale experimental study and development of a data mining algorithm that associates the metrics from palm reading and the users' data.

REFERENCES

[1] C. Kuo, "Searching for the Factors and Motives That Influence Online Fortune-Telling Behavior," *Mass Communication Research,* vol. 85, pp. 141-182, 2005.

[2] J. J. Zhang, "Zhangwen zhu wenli xiantiao chouqu [Extraction of Principal Palm Lines]," Department of Computer Science and Information Engineering, Chung Hua University, 2014.

[3] C. S. Leung, *Luoji shou xiang xue [Logical Study of Palm Reading]*. Di chuang zhizuo she, 1993.

[4] Y. F. Liu, "The predicting factors of consumers' perceived risk and purchase intention in online fortune-telling services," Graduate Institute of Technology Management, 2013.

[5] F. E. Sandnes, "Measuring 2D:4D finger length ratios with Smartphone Cameras," in IEEE International Conference on Systems, Man, and Cybernetics, 2014.

[6] S. M. van Anders, P. A. Vernon and C. J. Wilbur, "Finger-length ratios show evidence of prenatal hormone-transfer between opposite-sex twins," *Hormones and Behavior,* vol. 49, pp. 315-319, 2006.

[7] A.A. Bailey and P.L. Hurd, "Finger length ratio (2D:4D) correlates with physical aggression in men but not in women," *Biological Psychology*, vol. 68, no. 3, March 2005.

[8] G. D. Wilson, "Finger-length as an index of assertiveness in women," *Personality and Individual Differences,* vol. 4, no. 1, pp. 111 – 112, 1983.

[9] M. F. Luxen and B. P. Buunk, "Second-to-fourth digit ratio related to Verbal and Numerical Intelligence and the Big Five," *Personality and Individual Differences,* vol. 39, pp. 959-966, 2005.

[10] J. Kornhuber, G. Erhard, B. Lenz, T. Kraus, W. Sperling, K. Bayerlein , T. Biermann, C. Stoessel, "Low Digit Ratio 2D:4D in Alcohol, Dependent Patients", *PLoS ONE,* vol. 6, 2011.

[11] M. Evardone and G. M. Alexander, "Anxiety, Sex-linked Behavior, and Digit Ratios," *Arch Sex Behav,* vol. 38, pp. 442–455, 2009.

[12] T. Grimbos, K. Dawood, R. P. Burris, K. J. Zucker and D. A. Puts, "Sexual orientation and the second to fourth finger length ratio: a meta-analysis in men and women," *Behav Neurosci,* vol. 124, pp. 278–287, 2011.

[13] S. Cheung, "Smart palm reading app with auto pattern recognition (iOS platform)," City University of Hong Kong, 2014.

[14] S. M. Fung, *Guan zhang zhixin−−zhang qiu zhangwen pian [Palm Reading—Palm Mount and Palm Lines]*. Yuan Fang Chuban She, 2013.

[15] Lucky Top, "*Shouxiang tujie [Palm Reading Graphical Illustrations],"* 2016. [Online] Available: http://tiger168.com/hanfac/168han00.htm

[16] V. Nigam, D. Yadav and M. K. Thakur, "A Novel Approach for Hand Analysis Using Image Processing Techniques," *International Journal of Computer Science and Information Security,* vol. 8, no. 2, 2010.

[17] G. Sigurdsson and A. Wong, "Finger Tracking for Rapid Cropping Applications," Stanford University, 2008.

[18] T. Y. Chung, "Smart Palm Reading App with Auto Pattern Recognition (Android Platform)," City University of Hong Kong, 2014.

[19] C. H. Lee, "Palmprint Recognition using Simple Linear Regression," Tatung University, 2007.

[20] G. Kukharev and A. Nowosielski, "Visitor identification: elaborating real time face recognition system," in Proceedings of the 12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen-Bory, Czech Republic, 2004, pp. 157-164.

[21] O. Nobuyuki, "A threshold selection method from gray-level histograms," in IEEE Trans. Sys., Man., Cyber, vol. 9, 1979, pp. 62-66.

[22] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane", *Information Processing Letters,* pp. 18–21, 1973.