Contactless hand recognition

Xiaoqian Jiang
xiaoqian@andrew.cmu.edu

Wanhong Xu
wanhong@cmu.edu

1 Introduction

Because of the increased hygiene concern in biometric systems and the difficulty in recognizing fingerprints of manual laborers and elderly people, hand geometry has been currently employed in many systems for personal verification mostly as a complement to finger-print authentication.

Traditional hand geometry-based systems use low-resolution cameras or scanners to capture users' hand images with the help of peggies or by forcing them to touch a screen. Those systems measure a hand shape to extract its features, like lengths and widths of fingers, and hand contour, for recognition. Unfortunately, traditional techniques face unsolved problems: low discriminability due to low-resolution hand images and bad user acceptability because users worry about hygienic issues when they have to touch screens.

To improve discriminability and user acceptability, our new hand recognition systems have to acquire high-resolution hand images without peggy constraints and also contacts. However, those new hand images rise new challenges, like hand texture, motion and shadow, to extract hand shapes, and measure hand features. Our project aims at those new challenges and gives a machine learning solution.

2 Related works

Most of the hand-based biometric schemes use geometric features of the hand. Oden used geometric features and implicit polynomial invariants of fingers. The classifiers are based on Mahanlanobis distance [1]. Duta-Jain designed a classifier based on mean average distance of contours [2]. Kumar designed correlation-like similarity measurement to distinguish different hands [3]. All of them require certain constraints. We face, in the context of contactless requirements, new challenges like image segmentation, feature extraction and recognition are introduced. We proposed a suit of algorithms to address these individual problems, highlighted the following algorithms: a robust color segmentation algorithm, a pose and illumination insensitive hand extremities localization algorithm, an affine and Euclidean invariant implicit polynomial fitting algorithm and a novel indexing method for quick and accurate identification.

3 Method

3.1 Hand Segmentation

The segmentation procedure aims to separate the hand region from a hand image and extract the hand contour, also called hand boundary. However, this traditional method like

edge detectors, do not work here because they might find noise boundaries caused by artifacts, like rings, and specially creases, which can not be ignored in our high- resolution images. The hand contour must be accurate since the difference between hand of different individuals could be minute. Small biases in hand contour can lead to significant errors in feature detection and further hand recognition. Hence, the problem we have to face is how to distinct hand boundaries from other non-hand boundaries. Obviously, no edge detection algorithm or image segmentation algorithm addresses this problem in the literature.

Motivated by using skin color in face detection [5], we consider using hand skin color to do hand segmentation. The intuition behind it is that the (R, G, B) colors of pixels in hand region should follow some pattern, called hand color pattern. By learning this pattern, we can classify whether any pixel has the hand color, and then we can extract the hand boundary without worrying about noise boundaries.

[5] gives a color pattern for face colors, which is a heuristic classify rule consisting of seven conditions. The rule is listed as follows: (R, G, and B represent color levels of a pixel in RGB space.)

$R > 95$ and $G > 40$ and $B > 20$ and $max\{R, G, B\} - min\{R, G, B\} > 15$ and $|R - G| > 15$ and $R > G$ and $R > B$.

However, after careful observation, the above rule actually can be simplified to the following equivalent rule.

$R > 95$ and $G > 40$ and $B > 20$ and $R - G > 15$ and $R > B$

We test this rule over our hand image databases. The rule does correctly classify most of pixels in hand regions, but does badly classify pixels in over-exposed hand regions and gray parts of boundaries. The reason is that the parameters (numbers) in the above rule are learned from face images specially under different illumination. Learning those parameters directly from our hand training image would improve the performance. Unfortunately, the rule proposed by [8] is found empirically, and no algorithm is given to find those parameters automatically. In this project, we propose an algorithm to find the parameters in rules automatically.

Another finding in our experiments is that $R > 95$ dominates $G > 40$ and $B > 20$, and thus the rule can be further simplified. In the rest of this section, we ignore $G > 40$ and $B > 20$ conditions.

Let us take a look at this rule again in another view. The condition $R > 95$ can be rewritten as $R - 95 > 0$. We can think $R - 95 > 0$ is a linear regression classifier over R subspace of RGB color space. Similarly, $R - G > 15$ can also be rewritten as $R - G - 15 > 0$, a linear regression classifier over RG subspaces. Hence, the rule can be viewed as a combination of three linear classifiers over different RGB subspaces. However, how can we learn parameters of three classifiers together. We propose the following method by extending logistic regression.

Let X be RGB data of training pixels with training label Y. Let $X^l = \{X^l_R, X^l_G, X^l_B\}$ be the $l - th$ pixel and Y^l be its class label, skin pixel or not. For each subspace R, R&G and R&B, we define a parametric logistic regression model for it. For example, we have logistic regression model R & G subspace. We let parameters $W_{RG} = \{W^0_{RG}, W^1_{RG}, W^2_{RG}\}$, and have

$$P_{RG}(Y^l = 0|X^l) = \frac{1}{1 + exp(W^0_{RG} + W^1_{RG}X^l_R + W^2_{RG}X^l_G)} \quad (1)$$

$$P_{RG}(Y^l = 1|X^l) = \frac{exp(W^0_{RG} + W^1_{RG}X^l_R + W^2_{RG}X^l_G)}{1 + exp(W^0_{RG} + W^1_{RG}X^l_R + W^2_{RG}X^l_G)} \quad (2)$$

Similarly, we have also logistic regression models for R subspace, denoted as P_R with

parameters W_R , for R&B, denoted as P_{RB} with parameters W_{RB}. Now, we learn those parameters $W = \{W_{RG}, W_R, W_{RB}\}$ that satisfy

$$W \leftarrow argmax_W \Pi_l P_R(Y^l|X^l) P_{RG}(Y^l|X^l) P_{RB}(Y^l|X^l) \tag{3}$$

Those parameters can be updated by just using the same way as updating parameters in the logistic regression classifier. The motivation using the above goal function is to maximize the probability of three classifiers can correctly classify a pixel together.

In our hand image database, each hand image is 3008 * 2000 pixels. It's no way to label the full image for training. Instead, we pick up two small regions and label them manually. One has 300673 pixels as our training sample, and the other has 63024 pixels as our test sample. We train our classifiers using logistic regression to get the following rule: $-1.28306 + 0.01530 * R > 0$ and $-0.14815 + 0.23601 * R - 0.25353 * G > 0$ and $-0.14594 + 0.18861 * R - 0.21689 * B > 0$. We get 99.51% accuracy over the test samples by using classifier, but the original rule has 98.54% accuracy. Although we only get one percent improvement, we have a machine learning way to learn those parameters, and have comparable accuracy with empirical results.

Finally, we use a boundary walk algorithm to find the hand boundary. It works as following: We define a pixel is in the boundary if it has skin color but one of its bottom, top, left, right neighborhood pixels has no skin color. We first find one boundary pixel, and then recursively find boundary pixels in its neighbors.

3.2 Localization of Hand Extremities

Detecting and localizing the hand extremities, the fingertips and the valley between fingers is the first and most important step in feature extraction. The precision and robustness requirement on this procedure are high otherwise recognition will suffer from augmented measurement errors. Naive gradient method suffered from artifacts and unsmoothed contour, reported in [4].

Inspired by both gradient gram algorithm and radial distance method [4], we proposed a new method that iteratively refining hand extremities. The algorithm makes no assumption on the fingers' pose like gradient gram algorithms and it outperforms the previous radial distance method as duplicate reference points are used to find unique global maxima in the refinement iterations. Along with the upper and lower wrist points, we extracted 13 out of following 15 measurements in Figure 1 for recognition excluding measurements 1 and 11.

Algorithm 1: Fingertips and valley localization

Input: p (a set of hand contour points in counter-clockwise order)
Output: Location of fingertips and valley between fingers

1: Plot averaged gradient gram of the contour $c_j = \sum(arctan((y_i - y_{i-s})/(x_i - x_{i-s})))$, s is the step size, centroid of clustered local minima are sought as initial fingertip locations;
2: Starting from thumb's fingertip, find the radial distance maxima using two nearby fingertips as reference points. Label this maximum as valley between fingers.
3: Refine fingertip of index finger, middle finger, ring finger using its two neighbor valleys found by the step 2. For thumb and little finger fingertips, use the only neighbor valley.
4: Iterate this process to the next finger until little finger fingertips is refined.

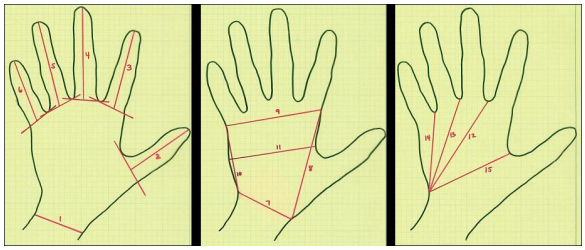


Figure 1: Measurements

3.3 Self Calibrate Implicit Polynomial Fitting

Implicit polynomials are known to be effective in representing non-star complex shapes in two dimensions. However, when objects are complex, low degree implicit polynomial fails to describe it and high degree implicit polynomial suffers from unstable results. These results are highly sensitive to noise, rotation and transformation. We propose a new method here to address this dilemma and our algorithm produce Euclidean and Affine invariant implicit polynomial fitting.

3.3.1 Implicit Polynomial Fitting

An implicit polynomial of degree N is a polynomial function $f(x, y) = 0$ where

$$f(x, y) = m^T a \tag{4}$$

, m is the $(N \times 1)$ column vector of monomials $x^i y^j, i+j <= N$, and a is the $(N \times 1)$ polynomial coefficients. "Implicit polynomial fitting" is the task to find t implicit polynomial coefficients that minimize its distance to the data points. The goal of implicit polynomial fitting is to approximate η_0, the set of data points in (x, y) represents a 2D object of interest by the zero level set of a implicit polynomial function $f(x, y)$. This is accomplished by minimizing the error function

$$E = \sum_{(x,y) \in \eta_0} f(x, y)^2 \tag{5}$$

3.3.2 3L Fitting Algorithm

Blane shows 3L implicit polynomial fitting algorithm is significant faster and more repeatable than existing methods [6]. The algorithm computes two ribbon belts (level sets) η_{-c}, η_{+c} of the interested image boundary using *D-Euclidian* distance transform function $\phi(x, y)$.

The least-squares solution for a is obtained by,

$$a = M_{3L}^{-1} b \tag{6}$$

here $M_{3L} = [M_{r_{-c}}\ M_{r_0}\ M_{r_c}]'$, $b = [-c\ 0\ +c]$, where $(M_{r_{-c}}\ M_{r_0}\ M_{r_c})$ are the $(N_{-c} * |C|)$, $(N_0 * |C|)$, $(N_{+c} * |C|)$ matrics of monomials for the corresponding sets of points in the ribbon surface $\varphi(x, y)$, and $-c$, 0, and $+c$ are the $(N_{-c} * 1)$, $(N_0 * 1)$ and $(N_{+c} * 1)$ column vectors having values $-c$, 0, and $+c$.

However, the result zero level set, as well as the algebraic and geometric invariants of 3L fitting computed from the coefficients of a are subject widely to rotations and translations of the data. This is because 3L fitting algorithm is inherently Euclidean but not affine invariant, refer to Equation 6. Levels sets η_{-c}, η_{+c} are computed by *D-Euclidian* distance transform function from the 2D object boundary η_0 point by point. As a result, distance of two points is not preserved under an affine transformation [7].

3.3.3 Self calibrate 3L fitting

3L implicit polynomial algorithm produces physically meaningful representations of complex objects but poses new challenges of affiant transformation unstableness. We proposed a new method that addresses the affine transformation problem while preserving 3L fitting Euclidean invariants.

To ensure a robust mathematical representation of implicit polynomial, it is good idea to convert the contour data to a standard position and coordination relative to its distribution. Intuitively, data points of the same shape have the same distribution in space. This is naturally translated into a question: what is the best way to "re-express" the original recorded data set X invariant to transformation and rotation. Observations lead us to make an assumption that directions with largest variance in measurement vector space contain the dynamics of interest.

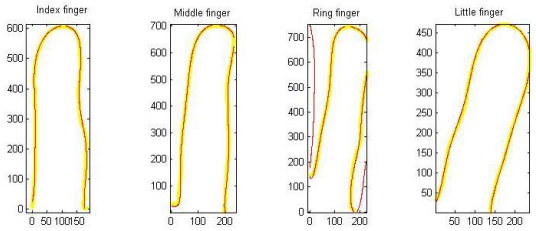


Figure 2: Self-calibrate fit for index, middle, ring and little fingers

This assumption suggests that the basis for which we are searching are the directions covering most variances in data, in other words, directions that minimizes the squared reconstruction errors. Principle Component Analysis (PCA) provides a practical solution to this problem with a orthogonal basis vectors [8]. Assume data is set of d-dimension vectors, where nth vector is $x^n = x_1^n, ..., x_d^n$. These vectors can be represented in terms of d orthogomial basis vectors $x^n = \sum_{i=1}^{d} z_i^n u_i, u_i' uj = 0$. PCA searches for $u_1, ..., u_M$ that minimizes $E_M = \sum_{n=1}^{N} ||x^n - \hat{x}^n||^2$, where $\hat{x}^n = \bar{x} + \sum_{i=1}^{d} z_i^n u_i$ and mean equals $\bar{x} = 1/N \sum_{n=1}^{N} x^n$. Instead of mapping the data into lower dimension, we projected the data using all orthogonal basis vectors to avoid information loss. In case of huge data dimension is involved, we replace PCA with SVD and select eigenvectors that covers 95% of total data variances. Algorithm 2 fuses the eigenvector decomposition with the 3L implicit polynomial fitting to produce Affine and Euclidean invariant results.

Algorithm 2: Self Calibrate 3L fitting

Input: p (a 2D object represented by its boundary)
Output: Affine and Euclidean invariant implicit polynomial efficients

1: Uniformly sample the original data X, set $ratio = 1/5$, collect X^s.
2: Perform PCA or SVD on the sample data X^s, compute eigenvalue $\lambda_1, ..., \lambda_d$ and corresponding eigenvectors $U = u_1, ..., u_d$.
3: Project original data to the orthogonal space $Y = XU$
4: Bring the centroid of the projected data to point (0,0), the center of the new coordinate system.
5: Fit 3L implicit polynomial to the transformed data points, compute the coefficients $a = M_{3L}^{-1} b$.

4 Experiments

In this section, we compare different feature selection and classification methods. Two different feature selection as well as six classifiers are chosen and results are compared and analyzed. In detail, we will evaluate the performance of feature selection methods: Relief-F and SVM-RFE. The classification algorithms to be compared include decision tree, Logistic Regression, Artificial Neural Network, Naive Bayes, 1-Nearest-Neighbor and linear SVM.

4.1 Data Acquisition

Our database consists of 48 frontal hand images. These are taken from 16 distinctive person and 3 image per person using Professor Sweeney's "Hand Capture Device". The database is available online at *http://austin.cs.uiowa.edu/xjia/handGeometry/rawdata/*. The result of hand segmentation in Section 3.1 produce smooth contours like Figure 3.

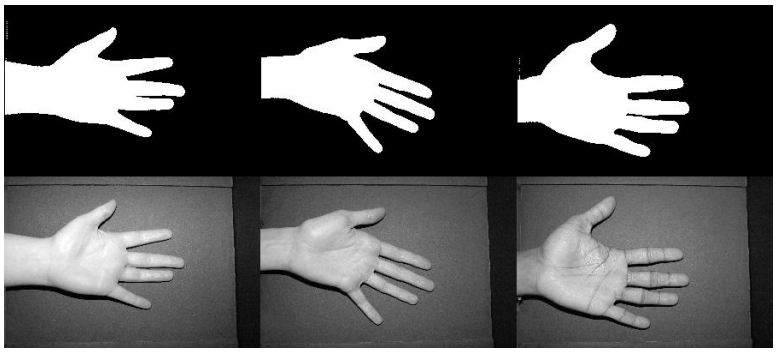


Figure 3: Color based segmentation

After feature extraction in section 3.2 and section 3.3, 13 measurement features and $15 * 4$ implicit polynomial coefficients for the index finger, the middle finger, the ring finger and the little finger are collected. Note that we fit three degree implicit polynomial to the segmented fingers. We used Matlab 7.0 and Weka [9] to conduct some of the experiments in this paper.

4.2 Experiment Result

4.2.1 Verification

We evaluated the verification performance of various algorithms in terms of accuracy. We first compared two different feature selection methods Relief-F and CfsSubsetEval(forward). The top 5 features selected are $[9, 7, 8, 6, 4]$ and $[2, 3, 4, 5, 7]$ respectively. Table 1 summarizes the comparison under 3-fold cross validation using these features. (Note that 3 is maximum number of fold since we have only 3 samples/class.)

It can be observed that the performance of using Relief-F features is slightly better than that of CfsSubsetEval(forward)'s. This coincides with our believe that greedy algorithm like CfsSubsetEval(forward) sometimes miss the correlation across potentially dependent features.

We further studied the performance of different classifiers as we gradually increase input features. Table 4(b) summarized the top 1 match accuracy rates of a few popular classification algorithms. The input features are obtained using Relief-F algorithm discussed

Table 1: Accuracy of the top 5 features selected by Relief-F and CfsSubsetEval(forward) under different algorithms using 3-cross validation

Methods	Relief accuracy	CfsSubsetEval(forward)
Decision Tree (DT)	0.56	0.56
Logistic Regression (LR)	0.85	0.79
ANN	0.87	0.833
NB	0.71	0.64
1 nearest neighbour (1NN)	0.92	0.895
Linear SVM	0.92	0.83

above. In specific, the features in this experiment are $[9, 7, 8, 6, 4, 3, 5, 12, 15, 14]$, ranked in descending order. We iteratively add new features and run the classifiers. Revealed by Figure 4(a), our best performer is 1-Nearest-Neighbor in this specific experiment setting. The highest accuracy is 94% with input feature $[9, 7, 8, 6, 4, 3]$.

Table 2: Top 1 match accuracy with various size of input features

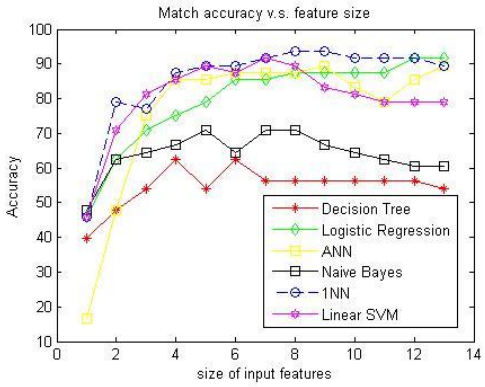
Methods	1	2	3	4	5	6	7	8	9	10
Decision Tree	0.40	0.48	0.54	0.63	0.56	0.56	0.63	0.56	0.56	0.56
LR	0.46	0.63	0.71	0.85	0.85	0.88	0.85	0.85	0.88	0.88
ANN	0.17	0.48	0.75	0.87	0.87	0.88	0.88	0.88	0.90	0.83
NB	0.48	0.63	0.65	0.64	0.71	0.71	0.65	0.71	0.67	0.65
1NN	0.90	0.79	0.77	0.90	0.92	0.94	0.90	0.92	0.94	0.92
Linear SVM	0.49	0.71	0.81	0.87	0.92	0.90	0.88	0.92	0.83	0.81

4.2.2 Identification

The cost for recognizing hand geometry in large databases could be very expensive. Here we provide a simple solution using implicit polynomial coefficients described in section 3.3 for fast indexing. We binary coded the signs of coefficients $A = a_1, ..., a_N$ of $X = x_1, ..., x_N$ into $S = s_1, ..., s_N$. Given D is the degree, $N = [(d + 1)(d + 2)]/2$. A search for similar hand shapes is pruned by counting the number of $1s$ after XOR two encoded coefficients. If there are multiple options after pruning, we used the best-performance algorithms in the verification section 4.2.1. Because we have only limited samples of hands, we tested our algorithm on the Brown Shape Indexing of Image Databases (SIID). We selected a tool achieve to evaluate the performance. A sample run is illustrated in Figure 4 and top 1 match rate on this sample achieve is 100% and 88.24% on average. We then felt confident to apply the same algorithm to our hand geometry database. The average searching was reduced by 85% while the best accuracy rate was preserved.

5 Conclusion

This paper introduced a machine learning solution for contactless hand recognition. We tried to address the new challenges such as hand texture, free motion, shadow and shape deformation. In this writing, we proposed a robust color segmentation algorithm, a pose



(a)

	21	4	0	15	9	9	11	8	8
	4	21	17	10	14	8	14	9	13
	0	17	21	6	12	12	10	13	13
	15	10	6	21	3	15	7	14	2
	9	14	12	3	21	3	17	4	20
	9	8	12	15	3	21	3	20	4
	11	14	10	7	17	3	21	2	16
	8	9	13	14	4	20	2	21	5
	8	13	13	2	20	4	16	5	21

(b)

Figure 4: (a) (b)Similarity between database shapes in terms of signs. In each row, large number corresponds to the similar shapes.

and illumination insensitive hand extremities localization algorithm, an Affine and Euclidean invariant implicit polynomial fitting algorithm and a novel indexing method for quick and accurate identification. Recognition accuracy is evaluated using two feature selection methods and a few popular classifiers. Among these results, the best matching accuracy is obtained by the nearest neighbor algorithm peaking 94%. This encourages our further effort along the same lines. Upon obtaining more subject hands and more sample for each hand, we expect our algorithm will have better performance.

References

[1] C. Oden, A. Ercil and B. Buke, "Combining implicit polynomials and geometric features for hand recognition", Pattern Recognit. Lett., vol 24, pp. 2145-2152, 2003

[2] A. K. Jain and N. Duta,"Deformable matching of hand shapes for verification," presented at the Int. Conf. Image Processing, Oct. 1999.

[3] Y. A. Kumar, D. C.M.Wong, H. C. Shen, and A. K. Jain, "Personal verification using palmprint and hand geometry biometric.", in Proc. 4th Int. Conf. Audio Video-Based Biometric Person Authentication, Guildford, U.K., Jun. 9-C11, 2003, pp. 668-C678

[4] Yoruk, E. Konukoglu, E. Sankur, B. Darbon, J., "Shape-based hand recognition", IEEE transactions on image processing, Vol. 15, Issue 7, Page 1803-1815, 2006

[5] Jure Kovac, Peter Peer, Franc Solina (2003), "Human Skin Colour Clustering for Face Detection.", International Conference on Computer as a Tool, pp.144- 148.

[6] Michael M. Blane, Zhibin Lei, "The 3L Algorithm for Fitting Implicit Polynomial Curves and Surfaces to Data.", IEEE transaction on PAMI, Vol. 22 No. 3, March 2000.

[7] Hakan Civi, "The Classical Theory of Invariants and Object Recognition Using Algebraic Curve and Surfaces." Journal of Mathematical Imaging and Vision 19: 237-253, 2003.

[8] Shlens, J., "A Tutorial on Principal Component Analysis: Derivation, Discussion and Singular Value Decomposition." Technical report, Available online from http://www.snl.salk.edu/ŝhlens/pub/notes/ pca.pdf, 2003.

[9] Ian H. Witten and Eibe Frank (2005) "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.