2D IMAGE DATABASE INDEXING: A COEFFICIENT-BASED APPROACH

Xiaoqian Jiang, Wanhong Xu, Yiheng Li, Latanya Sweeney, Ralph Gross, Daniel Yurovsky[*]

The Laboratory for International Data Privacy
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

The problem investigated in this paper is the quick identification of free-form 2D objects using a coefficient based indexing technique. We demonstrate that combining implicit polynomial shape descriptors with calibration techniques has the potential to quickly identify images of similar shapes. As a pilot experiment, our approach is evaluated against a small database consisting of 15 images, with three different images from five categories. The index system achieves an overall Top-3 matching accuracy of 88.9% comparing to 91.1% of the traditional brutal force method on the same database with a computational cutoff of 67.7%.

1. INTRODUCTION

With the rapid proliferation of the internet and the worldwide-web, the amount of digital image data accessible to users has grown enormously. Image databases are becoming larger and more widespread, and there is a growing need for effective and efficient image retrieval systems [1]. For instance, Google's Image Search contains billions of images available for viewing [2]. Google indexes these images based on text appearing around images in publicly available web pages. The associated text enables text search for images. While this is useful, it is limited. Often the text inappropriately describes the image and similar images have radically different text. There is increasing demand for automated indexing of images such that given an image, the search engine would retrieve similar images based on the image itself.

Many proposed techniques to associate images with free-form shapes based on their similarity measurements (e.g., [3]). A detailed survey of shape correspondence techniques can be found in [4]. Among the popular methods, there are shape moments, Fourier descriptors, Hausdorff distance and algebraic invariants [3], and shock-graph based shape matching [6]. In general, the basic idea underlying these methods is either retrieving global shape descriptor like invariants statistics, shape measures, deformable models, on the one hand, measuring local edge similarities, on the other [10]. However, current models have various limitations or drawbacks and potential of shape based image indexing in real world systems has not been fully appreciated.

- Global shape descriptor like implicit polynomial fitting [7] is robust in presence of noise and perturbations. But the result coefficients can not be used directly for recognition as they are subject to rotation, translation and affine transformation of the shape. The calculation of Euclidian and affine invariant descriptors is expensive yet the accuracy rate is not sufficient for practical systems.
- Local edge similarity method like shock graph descriptors [6] does a good job in classifying images into semantically meaningful categories based on their underlying structure. However, they are not designed to identify the difference between images of its own group. For example, brush rabbit and arctic hare are indistinguishable because the both share the same limb and body structure.

The aim of this writing is to demonstrate a novel approach to compute Euclidean and affine invariant polynomial coefficients via a conjunction of calibration techniques and a robust implicit polynomial fitting, and illustrate their use in recognizing objects through experiments.

2. SELF CALIBRATE IMPLICIT POLYNOMIAL FITTING

In model based vision, implicit polynomials have been proven to be a powerful object recognition technique in representing non-star complex shapes [7]. The strengths of this representation includes their interpolation properly for handling missing data and smoothing property against noise and pertubations. Thus, implicit polynomials provide a computational inexpensive solution to recognize simple two-dimensional objects. However, when objects become complex, low degree implicit polynomials fail to fit due to insufficient degree of

[*] Many thank to the members of the Data Privacy Lab at Carnegie Mellon. This work was funded by U.S. Department of Justice award 2005-IJ-CX-K046.

freedom. On the other hand, high degree implicit polynomials, though capable of describing complex shapes, result in unstable results because of their high sensitivity to noise, rotation, translation and other affine transformations. Most methods of computing algebraic invariants are thus ad-hoc and attentions have been restricted to arbitrary classes of invariants, or brutal force searches [5]. Note that algebraic invariants of a polynomial curve or surface is a function of polynomial coefficients that is either Euclidean or affine invariant.

Comparing to symbolic methods that generates algebraic invariants of high degree implicit polynomials of pages long [5], our method has a simple representation and can be extended up to arbitrary high degree.

2.0.1. *Implicit Polynomial Fitting*

An implicit polynomial of degree N is a polynomial function $f(x, y) = 0$ where $f(x, y) = m^T a$, 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 the distance of the polynomial to data points of interest. The goal is to approximate η_0, the set of data points (x, y) representing the boundary η_0 of a two dimension object D of interest, by the zero level set of a implicit polynomial function f. This is to minimize the following error function:

$$E = \sum_{(x,y)\in\eta_0} f(x, y)^2. \tag{1}$$

here, $(x, y) \in \eta_0$, η_0 is the boundary of D, which is equivalent to the following opimization problem:

$$a^* = argmin_a\{a' m'_{\eta 0} m_{\eta_0} a\} \tag{2}$$

Blane shows 3L implicit polynomial fitting algorithm is significantly faster and more repeatable than existing methods in [7]. To fit an object of interest, the 3L approach first computes two ribbon belts (level sets) η_{-c}, η_{+c}, (red and blue belts in Figure 1) of the object boundary (green belt), using *D-Euclidian* distance transform function $\phi(x, y)$. Figure 1 depicts three level sets and the surface of the corresponding *D-Euclidian* distance transform function of a simple object.

The least-squares solution for a is obtained by,

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

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 $(N_{-c} * |C|)$, $(N_0 * |C|)$, $(N_{+c} * |C|)$ matrices of monomials for the corresponding sets of points in the ribbon belts computed using $\varphi(x, y)$, and $-c$, 0, and $+c$ are the $(N_{-c} * 1)$, $(N_0 * 1)$ and $(N_{+c} * 1)$ column vectors having values $-c$, 0, and $+c$.

Although 3L fitting is Euclidean invariant, the raw coefficients themselves can not be directly used for recognition. The reason is coefficients a of 3L fitting are subject to change under rotations and translations. For example, $2X^2 + Y^2 = 1$ and $X^2 + 2Y^2 = 1$ represent the same oval object but they have different coefficients $[0, 0, 2, 0, 1]$ and $[0, 0, 1, 0, 2]$ with respect to monomials $[x, y, x^2, xy, y^2]$. Because coefficients are global descriptors, even if two shapes looks the same locally, their fitted coefficients could differ glocally under rotation or translation. Besides these limitations, 3L has an inherent inconsistency problem under geometric contraction and expansion. Refer to Equation 3, levels sets η_{-c}, η_{+c} are computed using *D-Euclidian* distance transform function from the two dimension object boundary η_0 point by point. Level sets generated on scaled objects are not preserved and thus polynomial coefficients a calculate before and after affine transformation are unstable.

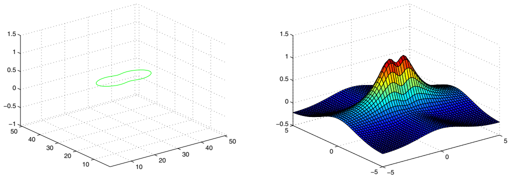


Fig. 1. Zero level set of the implicitit polynomial function and its associated polynomial surface

2.1. Self calibrate 3L fitting

To address the challanges in traditional 3L fitting, we propose a novel method to produce Euclidean and affine invariant polynomial coefficients. Intuitively, data points representing the same object have invariant distribution in their feature space regardless of translation and rotation. To ensure robust mathematical representation of implicit polynomials, it is good idea to convert the data of interest to a standard position and coordination with respect to its distribution. This problem is naturally translated into a question: what is the best way to "re-express" the original data set X invariant to translation and rotation. To normalize the calibration, we assume directions with largest variance in measurement vector space contain the dynamics of interest. For instance, a toy example from [8] demonstrate that the dynamics of a spring is along its extension, which is parallel to the X axis, refer to Figure 2.

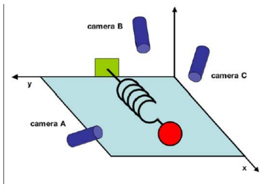


Fig. 2. Priciple Component Analysis Toy Example

This assumption suggests that the basis for which we are searching are the directions containing most variance, in other words, directions that minimize the squared reconstruction error. Our problem is thus naturally bridged to Principle Component Analysis (PCA), which provides a practical solution consisting of orthogonal basis vectors [8]. Assume the data is a 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 orthogonal 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 the mean equals $\bar{x} = 1/N \sum_{n=1}^{N} x^n$. Instead of mapping the data into a low dimension, we project the data using all orthogonal basis vectors to avoid information loss when data dimension is low. In case data dimension is huge, we replace PCA with SVD and select eigenvectors representing 95% of total data variances. The result coefficients are now insensitive to both rotation and translation. Note that the results are also affine invariant due to our normalization step 5.

```
Algorithm 1 Self Calibrate 3L fitting
Input: $p$ (Images of 2D free-form objects)
Output: Affine and Euclidean invariant polynomial
coefficients.
1: Uniformly sample $1/5$ of the original data $X$, collect
   $X^s$.
2: Perform PCA or SVD on the sampled 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: Normalize the images with respect to x axis, preserve
   their original ratios.
6: Fit 3L implicit polynomial to the transformed data
   points, compute the coefficients $a = M_{3L}^{-1} b$.
```

2.2. Index

Using polynomial coefficients obtained in the previous section, we propose a heuristic indexing approach. We now briefly describe our algorithm. The idea underlying coefficient-based matching is to associate images based on boundary similarity represented by their polynomial coefficients global descriptors. However, computing shape distance function over the entire database can be formidably expensive. To avoid unnecessary computation, we introduce a heuristic method to calculate distance function E_D^Q only when multiple candidates remained after pruning the query. The pruning step is based on congruent signs of polynomial coefficients in a query shape against previous recored coefficients of a shape database.

Next, we introduce the shape distance function. Given a query image Q containing Z_Q points and an image database D_{image} containing $D_k, k \in 1...n$ images with each Z_i points in each image. The fitting models for these are

$$(x, y) : f_q = \sum_{0<i,j;i+j\leq de} a_q x^i y^j = 0 \tag{4}$$

$$(x, y) : f_k = \sum_{0<i,j;i+j\leq de} a_{D_k} x^i y^j = 0. \tag{5}$$

Here de refers to the degree used to fit polynomials and $N_{q,k}$ represents data dimension of the union of the query image Q and a image D_k in the database. Following reference [9] , we define distance between Q and D_k as:

$$\begin{aligned} E_{D_k}^Q &= ||f_q - f_k||_{Z_Q \cup Z_k} \\ &= \frac{1}{N_{q,k}} \sum_{(x,y)\in(Z_Q \cup Z_k)} [\sum_{0<i,j;i+j\leq de} (a_q - a_{D_k}) x^i y^j]^2 \end{aligned} \tag{6}$$

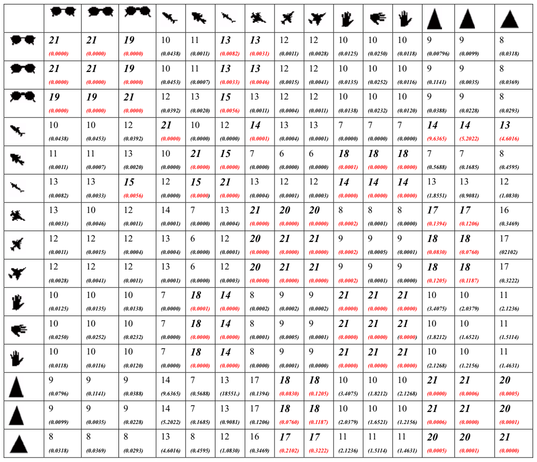
```
Algorithm 2 Coefficient-based indexing
Input: A query image $Q$ and an image database $D_{image}$
with implicit polynomial coefficients
Output: Top-K similar images $P_1, ..., P_K$ in the database

Initalize similarity threshold $K$ and $K^*, K < K^*$,
compute total number of coeffients
$N = [(de + 1)(de + 2)]/2$, Given $de$ is the degree of
polynomials.
Fit implicit polynomials to the query image $Q$ using
Algorithm 1, obtain polynomial coefficients
$A_q = a_1, ..., a_N$.
Binary codes $A_q = a_1, ..., a_N$ to a sequence of 0s and 1s
$S_q = s_1, ..., s_N$ based on their signs.
for Each binary coded coeffients $S_j, j \in [1...n]$ in the
database do
    $T_j = \sum XOR(S_q, S_j)$
end for
Push Top-$K^*$ candidates in terms of $T_j$ into a stack
$Stack_{candidates}$.
for Each candidate $d, d \in D_{image}$ in $Stack_{candidate}$ do
    Calculate the distance (Equation 6).
end for
Return K images $P_1, ..., P_K$ with smallest $abs(E_D^Q)$
value.
```

3. EXPERIMENTS AND RESULTS

We demonstrate our shape matching algorithm with examples. To evaluate the performance, we used the Brown Shape Indexing of Image Databases (SIID) to test our algorithm [6].

We choose a subset consisting of three images in each of the following five categories: fish, plane, triangle, hand and glasses. We collect five candidate images for each query when the threshold $K^* = 5$. Note that the pruning step should always be considered in conjunction with distance based matching in the second step. Following Algorithm 2 in Section 2, we fit five degree implicit polynomials to PCA-calibrated images. Only the Top-K^* candidates of the pruning step are considered for distance matching. Refer to Figure 3, the figures on the first line of each cell refer to the number of congruent signs in polynomial coefficients. The figures in the parentheses refer to the distance between two different shapes. In the pruning step, we select best matching candidates and highlight their figures in bold italic format. Distance (red figures) is only computed for these candidates.



													▲	▲	▲
	21 *(0.0000)*	***21*** *(0.0000)*	***19*** *(0.0000)*	10 *(0.0438)*	11 *(0.0011)*	***13*** *(0.0082)*	***13*** *(0.0031)*	12 *(0.0011)*	12 *(0.0028)*	10 *(0.0125)*	10 *(0.0250)*	10 *(0.0118)*	9 *(0.00796)*	9 *(0.0099)*	8 *(0.0318)*
	21 *(0.0000)*	***21*** *(0.0000)*	***19*** *(0.0000)*	10 *(0.0453)*	11 *(0.0007)*	***13*** *(0.0033)*	***13*** *(0.0046)*	12 *(0.0015)*	12 *(0.0041)*	10 *(0.0135)*	10 *(0.0252)*	10 *(0.0116)*	9 *(0.1141)*	9 *(0.0035)*	8 *(0.0369)*
	19 *(0.0000)*	***19*** *(0.0000)*	***21*** *(0.0000)*	12 *(0.0392)*	13 *(0.0020)*	***15*** *(0.0056)*	13 *(0.0011)*	12 *(0.0004)*	12 *(0.0011)*	10 *(0.0138)*	10 *(0.0232)*	10 *(0.0120)*	9 *(0.0388)*	9 *(0.0228)*	8 *(0.0293)*
	10 *(0.0438)*	10 *(0.0453)*	12 *(0.0392)*	***21*** *(0.0000)*	10 *(0.0000)*	12 *(0.0000)*	***14*** *(0.0001)*	13 *(0.0004)*	13 *(0.0001)*	7 *(0.0000)*	7 *(0.0000)*	7 *(0.0000)*	***14*** *(9.6365)*	***14*** *(5.2022)*	***13*** *(4.6016)*
	11 *(0.0011)*	11 *(0.0007)*	13 *(0.0020)*	10 *(0.0000)*	***21*** *(0.0000)*	***15*** *(0.0000)*	7 *(0.0000)*	6 *(0.0000)*	6 *(0.0000)*	***18*** *(0.0001)*	***18*** *(0.0000)*	***18*** *(0.0000)*	7 *(0.5688)*	7 *(0.1685)*	8 *(0.4595)*
	13 *(0.0082)*	13 *(0.0033)*	***15*** *(0.0056)*	12 *(0.0000)*	***15*** *(0.0000)*	***21*** *(0.0000)*	13 *(0.0004)*	12 *(0.0001)*	12 *(0.0003)*	***14*** *(0.0000)*	***14*** *(0.0000)*	***14*** *(0.0000)*	13 *(1.8551)*	13 *(0.9081)*	12 *(1.0830)*
	13 *(0.0031)*	10 *(0.0046)*	12 *(0.0011)*	14 *(0.0001)*	7 *(0.0000)*	13 *(0.0004)*	***21*** *(0.0000)*	***20*** *(0.0000)*	***20*** *(0.0000)*	8 *(0.0002)*	8 *(0.0001)*	8 *(0.0000)*	***17*** *(0.1394)*	***17*** *(0.1206)*	16 *(0.3469)*
	12 *(0.0011)*	12 *(0.0015)*	12 *(0.0004)*	13 *(0.0004)*	6 *(0.0000)*	12 *(0.0001)*	***20*** *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	9 *(0.0002)*	9 *(0.0005)*	9 *(0.0001)*	***18*** *(0.0830)*	***18*** *(0.0760)*	17 *(02102)*
	12 *(0.0028)*	12 *(0.0041)*	12 *(0.0011)*	13 *(0.0001)*	6 *(0.0000)*	12 *(0.0003)*	***20*** *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	9 *(0.0002)*	9 *(0.0001)*	9 *(0.0000)*	***18*** *(0.1205)*	***18*** *(0.1187)*	17 *(0.3222)*
	10 *(0.0125)*	10 *(0.0135)*	10 *(0.0138)*	7 *(0.0000)*	***18*** *(0.0001)*	***14*** *(0.0000)*	8 *(0.0002)*	9 *(0.0002)*	9 *(0.0002)*	***21*** *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	10 *(3.4075)*	10 *(2.0379)*	11 *(2.1236)*
	10 *(0.0250)*	10 *(0.0252)*	10 *(0.0232)*	7 *(0.0000)*	***18*** *(0.0000)*	***14*** *(0.0000)*	8 *(0.0001)*	9 *(0.0005)*	9 *(0.0001)*	***21*** *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	10 *(1.8212)*	10 *(1.6521)*	11 *(1.5114)*
	10 *(0.0118)*	10 *(0.0116)*	10 *(0.0120)*	7 *(0.0000)*	***18*** *(0.0000)*	***14*** *(0.0000)*	8 *(0.0000)*	9 *(0.0001)*	9 *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	***21*** *(0.0000)*	10 *(2.1268)*	10 *(1.2156)*	11 *(1.4631)*
▲	9 *(0.0796)*	9 *(0.1141)*	9 *(0.0388)*	14 *(9.6365)*	7 *(0.5688)*	13 *(18551.)*	17 *(0.1394)*	***18*** *(0.0830)*	***18*** *(0.1205)*	10 *(3.4075)*	10 *(1.8212)*	10 *(2.1268)*	***21*** *(0.0000)*	***21*** *(0.0006)*	***20*** *(0.0005)*
▲	9 *(0.0099)*	9 *(0.0035)*	9 *(0.0228)*	14 *(5.2022)*	7 *(0.1685)*	13 *(0.9081)*	17 *(0.1206)*	***18*** *(0.0760)*	***18*** *(0.1187)*	10 *(2.0379)*	10 *(1.6521)*	10 *(1.2156)*	***21*** *(0.0006)*	***21*** *(0.0000)*	***20*** *(0.0001)*
▲	8 *(0.0318)*	8 *(0.0369)*	8 *(0.0293)*	13 *(4.6016)*	8 *(0.4595)*	12 *(1.0830)*	16 *(0.3469)*	***17*** *(0.2102)*	***17*** *(0.3222)*	11 *(2.1236)*	11 *(1.5114)*	11 *(1.4631)*	***20*** *(0.0005)*	***20*** *(0.0001)*	***21*** *(0.0000)*

Fig. 3. Coefficient-based matching procedures: 1. Prune the candidates 2. Compute similarities. When $K^* = 3$, apply pruning step in conjunction with distance based matching, the first choice is always correct and the second, third choices are $13/15$ and $12/15$ times correct, respectively.

The threshold K^* controls the tradeoff between accuracy and computational cost. Larger threshold K^* includes more candidates for distance matching and better accuracy rate while smaller threshold K^* has less computational cost. For example, when $K^* = 15$, distance is calculated for every pair of images and the best averaged Top-3 matching rate is 91.1%. This result corresponse to applying the brutal force approach mentioned in [9]. When $K^* = 3$, the averaged Top-3 matching rate is 88.9% (Accuracy is only sligtly degraded. First, second and third choices are $15/15$, $13/15$ and $12/15$ times correct, respectively) with a computational cutoff 67.7% of the above brutal force method.

4. CONCLUSION

This paper suggests an alternative approach to index 2D shape in large shape databases. The algorithm combines 3L fitting with PCA calibration. The result coefficients are both Euclidean and affine invariant, providing a simple shape representation for recognition purpose. In addition, the algorithm can be easily extended to higher dimension using a voting scheme. However, even though the pilot experiment conducted on a small image database indicates a high matching accuracy with reduced computational cost, more thorough experiements are needed to be run, including a large scale test of different kinds of affine transformations over various objects and performance evaluation in the presence of occlusion.

5. REFERENCES

[1] Michael S. Lew, et al, “Content-based multimedia information retrieval: State of the art and challenges”, ACM Transactions on Multimedia, Computing, Communications, and Applications, vol. 2, pp. 1-19, 2006.

[2] “Google Image Search Freuquently Asked Questions,”*http://www.google.com/help/faq_images.html*, 1/20/2007.

[3] V. Jain and H. Zhang, “Robust 2D Shape Correspondence using Geodesic Shape Context,”*in Proceeding of Pacific Graphics 2005, (short paper)*, pp. 121-124, 2005.

[4] R. C. Veltkamp, M. Hagedoorn. “State of the art in shape matching,” *In Technical Report UU-CS-1999-27*, Utrecht, 1999.

[5] Hakan Civi, “The Classical Theory of Invariants and Object Recognition Using Algebraic Curve and Surfaces,” *Journal of Mathematical Imaging and Vision*, vol. 19, pp. 237-253, 2003.

[6] T. Sebastian, P.N. Klein, and B.B. Kimia, “Shock-Based Indexing into Large Shape Databases,” *Proc. European Conf. Computer Vision*, vol. 3, pp. 731-746, 2002.

[7] 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.

[8] Shlens, J., “A Tutorial on Principal Component Analysis: Derivation, Discussion and Singular Value Decomposition,” *Technical report*, 2003.

[9] Zhibin Lei, Tolga Tasdizen and David B. Cooper, “PIMs and Invariant Parts for Shape Recognition,” *ICCV*, pp. 827-832, 1998.

[10] D. Sharvit, J. Chan, H. Tek, BB. Kimia, “Symmetry-based indexing of image databases,” *Journal of Visual Communication and Image Representation*, 1998.