INVARIANT, INTRA-CLASS RETRIEVAL IN HOMOGENEOUS IMAGE DATABASES

Ronald Alferez and Yuan-Fang Wang
{ronald,yfwang}@cs.ucsb.edu

Department of Computer Science
University of California at Santa Barbara

ABSTRACT

In this paper, we present a new method for content-based image retrieval and indexing. The context is for searching an image database containing objects of similar appearance for a query object in a way that is insensitive to incidental environmental factors such as change in viewpoints and slight, nonessential shape deformation. The scheme is well suited for discriminating among objects within the same class, with members that can appear very similar to each other (what we call a "homogeneous" database, e.g., the FishBase containing thousands of fish images). The scheme comprises a global alignment and a local matching process. Affine transform is used to model the different viewpoints associated with positioning the camera. Furthermore, multi-dimensional indexing techniques are used to make the global alignment scheme efficient. A local matching process based on dynamic programming allows optimal matching of local structures using cost metrics that may ignore nonessential local shape deformation. Results show the method's ability to cancel out visual distortions caused by a changing viewpoint, tolerance to noise, occlusion, and slight deformations of the object, as well as its ability to distinguish between similar, but different objects of the same class.

1. INTRODUCTION

Content-based image search engines can be very useful in today's explosion of available multimedia archives. Yet, one cannot deny that these tools have been largely underutilized. For example, scientific digital image libraries such as the FishBase [3] featuring thousands of images of fish and the Perseus Digital Library [2] which features images of historical objects like vases and sculptures, still provide only text-based queries. There is a clear lack of enthusiasm in embracing current image-based search technology, indicating that further research is needed to gain wider acceptance.

In this paper, we present a new method for content-based image retrieval and indexing which address the problems involved in searching homogenous image databases such as [2][3]. These types of databases are composed of images of the same class of objects, thus appearing very similar to each other. While majority of image-retrieval systems are able to distinguish between different classes of objects (e.g., fish vs. airplanes, or what we call the "*inter-class*" retrieval problem), only a few are capable of discriminating among objects within the same class (e.g., trout vs. bass, or what we call the "*intra-class*" retrieval problem;). Clearly, it is desirable to have a system that succeeds at the more specific *intra-class* queries, such as retrieving pictures of rainbow trout (characterized by the shape of the body and fins) from an ensemble of fish images. However, current systems will likely fail with this query, generating lists of images containing various species of fish. The *aggregate* features adopted by many current systems (such as histograms and low-ordered moments surveyed in [6][7]) capture only the general shape of a class and are not descriptive enough to distinguish objects within a particular class.

Moreover, only a handful of the current systems are able to accommodate the visual distortions caused by incidental environment changes, and slight and nonessential deformations of the query object. Allowing global viewpoint change and local deformation increases the complexity of the search. Query images, though belonging to the class of interest, can appear different due to several factors, such as a change in the camera's viewpoint, or local shape deformations attributed to noise, occlusion, or simply peculiarities of the individual objects. It is this more challenging scenario (*that of intra-class retrieval, with invariance to viewpoint change and nonessential shape deformation*) that is the focus of this paper.

Invariant, intra-class retrieval is important in many real-world applications. For example, a useful botanical image database application might be to help identify a leaf as belonging to a particular species. Children visiting an aquarium might produce an image of a fish (taken from an unknown viewpoint) and an automated search is performed to provide information about its particular species, along with other interesting facts such as migration patterns. Other applications include searching on-line catalogs of tools, trademarks, etc., for similar designs and patterns. Many real-world applications require intra-class retrieval, and in a manner that is immune to incidental environmental changes and slight deformations of the object.

To achieve viewpoint invariance, the affine transform is used to model the different viewpoints associated with positioning the camera. In general, we extract representative feature points from the image (e.g., corner and inflection points from contours). The recovered affine parameters, derived from an efficient multi-dimensional indexing process, will allow a best global alignment between the query object and a few good candidate objects from the database.

The discrepancy between two object contours produced by such a global alignment process, in general, can be attributed to a number of factors: such as noise, and small local deformation (e.g., resulted from the flapping of tail and dorsal fins when a fish is swimming). To further refine the matching, we use a local matching process (based on dynamic programming) to determine the shape deformation. The cost in the match is used to indicate how much local deformation there is between the two objects. The cost metric can be designed in such a way (with domain specific knowledge) to discount nonessential or incidental local deformation.

2. TECHNICAL DETAILS

There are two major issues we must address: canceling out the effect of a changing viewpoint through global alignment, and obtaining accurate similarity measure through local matching.

2.1 Global alignment

To achieve viewpoint invariance, the affine transform is used to model the different viewpoints associated with positioning the camera. The affine transform model has the ability to account for an imaged object's rigid motion, as well as the camera's change in viewpoint and/or zoom. Using a combination of translation, rotation, scale, and shear, this model is often sufficient to explain the many different ways that an object can be posed (or conversely, the many different ways the camera can be positioned). It is also a generally accepted approximation to the more general perspective projection, as long as the perspective distortion is not too severe [4]. Mathematically, an affine transformation of a point $[x,y]^T$ is defined by

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \qquad \text{(Eq. 1)}$$

for some nonsingular matrix m, and m_{ij}, t_x, $t_y \in \Re$. Such a transformation preserves a number of characteristics in an image. For instance, straight lines will remain straight and parallel lines remain parallel. On the other hand, many significant measurements are lost in the transformation. For example, distances are not preserved, nor are angles between line segments. This visual distortion in the image is primarily what makes it difficult to compare images with one another.

At the heart of our global alignment algorithm is a well-known geometric property on affine transform, which states that the ratio of the area of two triangles is constant under an affine transformation [7]. That is, given any two triangles Δ_1 and Δ_2,

$$\frac{\alpha(\Delta_1)}{\alpha(\Delta_2)} = \frac{\alpha(\Delta'_1)}{\alpha(\Delta'_2)} \qquad \text{(Eq. 2)}$$

where α denotes the area and Δ' is some affine transformation of Δ. Assuming that we can effectively decompose an image into a small set of representative feature points, we can use the fact that a correspondence of only three ordered points uniquely determines an affine transformation between images, thus allowing us to recover the pose of the object. However, the number of possible triplet correspondences between two sets of n points is in the order of n^6. Since only one correspondence is needed, the search space is reduced to n^3. Therefore, assuming v is the time to verify a similarity match between two images, and there are m models in the database, the total search time for a given query will be in the order of n^3*m*v, which can be prohibitively high, especially with very large databases.

Therefore, we need to make the process efficient, achieved by employing ideas from multi-dimensional indexing [1] and geometric hashing [4] techniques that have gained popularity over the past decade. The main idea is to build a hash table that can be processed off-line without prior knowledge of the query image. A set of affine-invariant features (Eq.2) is extracted from each image in the database, and is used as indexes into the hash table to enter the same image code in the different bins. During the search phase, the same affine-invariant features are extracted from the query image, and used as indexes into the hash table, with the expectation that similar objects will index to the same set of bins in the hash table. Hence, by inspecting only relevant bins in the hash table during the search phase, and ranking the images based on the number of times they are retrieved from the hash table, huge numbers of unlikely candidates from the database are immediately filtered out, allowing for a closer inspection of only a small number of good candidates.

Since the hash table is built only once and well in advance of any search task, the time needed to construct it becomes immaterial. As will be shown in Sec. 2.1.3, however, the actual search time can be reduced to the order of n*v, if we assume a negligible number of collisions in the bins.

2.1.1 Pre-processing stage

A multi-dimensional hash table is constructed, and is filled in with encoded information from each image in the database. Each image is reduced to a set of representative feature points, which are carefully chosen such that the same set of points should also be present in the query image (e.g., corner and inflection points).

For each image M in the database represented by m feature points, and for each ordered triplet (P_1,P_2,P_3) of M,

1. Compute a set of 3-tuple index keys, described in Sec.2.1.2.
2. At the bins designated by each 3-dimensional index key, the entry $(M,(P_1,P_2,P_3))$ is added.

2.1.2 Computing the hash index key

Given a triangle $\Delta P_1P_2P_3$ (using three feature points as vertices), we use the remaining m-3 feature points Q_i in M to compute a set of ratios that are invariant to an affine transformation (see Eq.2). We present two variations.

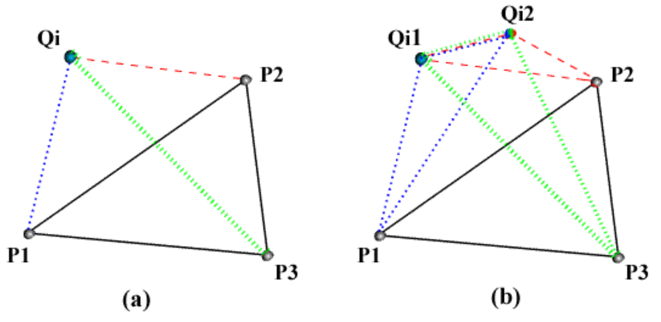

Figure 1. Two ways to compute the hash index key.

In Figure 1a, the hash index keys are computed as follows. For each point Q_i, compute the areas of $\Delta P_1P_2Q_i$, $\Delta P_1P_3Q_i$ and $\Delta P_2P_3Q_i$, which become part of the first, second, and third dimensions of a 3-dimensional key, respectively. The hash index key corresponding to point Q_i, with α denoting area, will be

$$\left(\frac{\alpha(\Delta P_1P_2Q_i),\alpha(\Delta P_1P_3Q_i),\alpha(\Delta P_2P_3Q_i)}{\alpha(\Delta P_1P_2P_3)}\right), \qquad \text{(Eq. 3)}$$

which is quantized according to a fixed bin size. This results in m-3, 3-tuples that are affine-invariant.

An alternate version is shown in Figure 1b. For each unordered pair of points Q_{i1} and Q_{i2}, we compute the areas of $\Delta P_1Q_{i1}Q_{i2}$, $\Delta P_2Q_{i1}Q_{i2}$ and $\Delta P_3Q_{i1}Q_{i2}$. As before, the hash index key corresponding to the unordered pair $\{Q_{i1},Q_{i2}\}$ will be the quantized components of

$$\left(\frac{\alpha(\Delta P_1Q_{i1}Q_{i2}),\alpha(\Delta P_2Q_{i1}Q_{i2}),\alpha(\Delta P_3Q_{i1}Q_{i2})}{\alpha(\Delta P_1P_2P_3)}\right), \qquad \text{(Eq. 4)}$$

resulting in ${}^{m-3}\mathbf{C}_2$ (order of m^2) 3-tuples that are affine-invariant.

The choice of which variation to use is completely domain-specific, the most noticeable difference being in the number of entries produced by each method.

2.1.3 Retrieval of candidates

After the off-line construction of the hash table, the system can search for a given query image, represented by n feature points,

1. Select an arbitrary ordered triplet (P_1,P_2,P_3) from the set of points of the query image, and compute a set of 3-tuple index keys for this triplet, as described in Sec.2.1.2.
2. Check the entries in each bin designated by the computed index keys, and accumulate the image/triplet information encoded within. To account for noise and deformations of the object, neighboring bins within a certain radius should also be included.
3. Compute a histogram for the occurrence of each accumulated (image, triplet) pair, and select the peak values as suitable candidates for the correct image and triplet values in the database corresponding to the query image and triplet (P_1,P_2,P_3), respectively.
4. Since three ordered points uniquely determines an affine transformation between two images, recover the affine parameters $\boldsymbol{A}$ which transforms the candidate triplet to the triplet (P_1,P_2,P_3). Immediately exclude affine transforms that unrealistically distort the image, by using the condition number of the 2x2 matrix of $\boldsymbol{A}$ (Eq.1), $\|m\|*\|m^{-1}\|$, as an estimate. The condition number measures how nearly singular a matrix is.
5. Apply $\boldsymbol{A}$ to all feature points (and possibly the whole image itself) in the database image M to obtain M', canceling out the effect of any affine transformation.
6. Verify that M' is a suitable match for the query image, using a suitably designed similarity metric, as described in Sec. 2.2. If no (image, triplet) candidate pair is a suitable match, or if a refinement of the search results is desired, return to step 1 for another triplet.

2.2 Local matching

Since M' (from Sec. 2.1.3, step 5) and the query image are now properly aligned in the same affine coordinate frame, any visual distortion caused by a change in viewpoint, including translation, rotation, scale, and shear, has been canceled out. The two images can thus be conveniently compared, point-by-point, using a similarity metric suitably designed for the particular domain.

Conventionally, the measure of similarity is usually defined to be the minimum summation of some cost function (such as squared Euclidean distances) between matched points (with corresponding penalties for unmatched points), among all possible pairings of the points between the two contours. However, this does not have to be the only choice. If we are interested in qualitative similarity, e.g., the two contours should have similar twists-and-turns characteristics, (for example, maple leaves are not going to be identical in shape and size, but will have very similar visual characteristics that are described by a pattern grammar or a production rule for that species), then a metric measuring the number and ordering of corner and inflection points will suffice, without insisting that the corner and inflection points be at exactly the same locations. Domain specific knowledge, (e.g., downplaying the dissimilarity of fin and tail positions when a fish is swimming) can also be brought in to emphasize/de-emphasize certain characteristics. By carefully designing the cost function, nonessential local deformation can be downplayed.

The underlying assumption from Sec. 2.1.3, step 3, is that the correspondence of three points, (P'_1,P'_2,P'_3) to (P_1,P_2,P_3), is already known. Hence, we can break the problem down into matching the three contour segments, $P'_1P'_2$, $P'_2P'_3$, and $P'_3P'_1$ with the corresponding segments from the other contour P_1P_2, P_2P_3, and P_3P_1, respectively.

Consider matching m points s_i of segment S to n points s'_j of segment S'. The ordering of the points along the contour is assumed preserved. That is, if point s_i corresponds to s'_j, then a point s_k can only correspond to points s'_l, for $k < i$ and $l < j$, or for $k > i$ and $l > j$. To obtain an optimal solution, a dynamic programming algorithm is used, described as follows. A table T is constructed, and filled according to the following rules:

$$\begin{aligned}&\mathrm{T}[i,0] = i * \rho\\ &\mathrm{T}[0,j] = j * \rho \qquad \text{(Eq. 5)}\\ &\mathrm{T}[i,j] = \min(\mathrm{T}[i-1,j-1] + \delta(s_i,s_j), \mathrm{T}[i-1,j] + \rho, \mathrm{T}[i,j-1] + \rho)\end{aligned}$$

where ρ is the penalty imposed for failing to match a point, and δ is the cost function between two points, typically the squared Euclidean distance. An entry T[i,j] represents the optimal value in matching the first i points in S with the first j points in S'. The optimal solution for the whole contour segment is thus found by computing for T[m,n]. As the table is filled-in diagonally from T[0,0] to T[m,n], the operation can easily be visualized as going through each contour segment, point-by-point, starting with the first pair of points. At each turn, there is choice of whether to match the two points, declare the point of S as unmatched, or declare the point in S' as unmatched. As the optimal solution is found, point correspondence for all points is easily established by tracing the decision made at each turn.

3. EXPERIMENTAL RESULTS

We present results obtained from two different databases. In both cases, the feature points used are corner points and points of inflection along the object contour. Note that although these feature points are not affine invariant in a strict sense, they are likely to remain unchanged in an affine transformation with a distortion that is not too severe and the viewpoint is not incidental (e.g. any 2D pattern looked edge-on reduces to a line segment with no corner or inflection points, we use the condition number to make sure that this degeneracy does not happen). A 4-dimensional hash table was used, the additional fourth dimension being a combination of the type of each point in the triplet. In particular, we classified each feature point as being one of the following six types: a corner forming a convex/concave angle that is a member/non-member of the convex hull (a total of 4 types), and a point of inflection which goes in one direction or the another (a total of 2 types). The second variant (Eq.4) was used to compute the hash index keys. We stored entries only for the ten largest (in terms of triangular area) triplets in each model, and consequently, iterating among the ten largest triplets in the query image. Dynamic programming was used (Sec. 2.2) to optimally measure the amount of local deformation between the query image and the candidate images, using the squared distance between corresponding points as the cost function δ (in Eq.5).

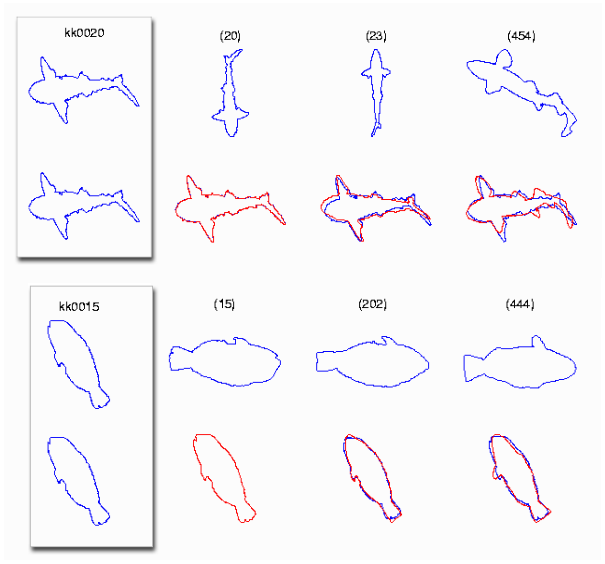

Figure 2. Results from querying the SQUID database. The leftmost columns are the query images. The top three search results (1st and 3rd rows) also show the best global alignment found (2nd and 4th rows).

Figure 2 shows results from the SQUID database used in [5]. The database consists of 1,100 images of marine animal contours, many of which appear very similar to each other. The boxed images on the left are the query images and its contour. (Unfortunately, the actual scanned images are not provided due to copyright restrictions, so for the SQUID database, the contour *is* the image itself). The top three search results are shown, arranged from left to right, with the leftmost being the most similar to the query image. The upper rows depict the retrieved images from the database, identified by the image number in parentheses. The lower rows show the best global alignment that was found between the query image and the corresponding database image. Such results are typical for this database, affirming the proposed method's ability to retrieve highly similar objects, while at the same time, disregarding varying viewpoints and nonessential shape deformation.

Similarly, Figure 3 shows search results from a small subset of the FishBase [3] database. Fifty images were used, the contours of which were traced semi-automatically with the aid of an "edge-seeking" selection tool. Notice the deformation of the fins in the query image as the fish moves on water. As before, results show that the method is adept at retrieving very similar objects.

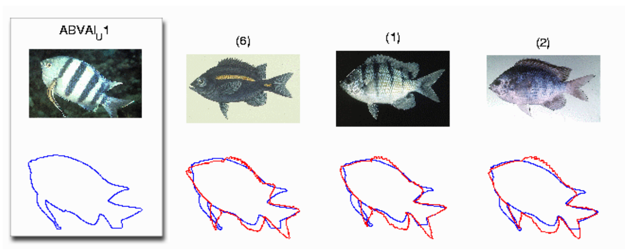

Figure 3. Results from querying the Fishbase database.

Note that even though contours were used to represent these objects, were used to select the feature points, and shape similarity metrics were used to verify the match, it was not necessary to do so. The feature points could very well have been selected from the raw image itself, and a different image similarity metric used, such as one that uses color/texture information. The decision to rely on contours in this case was based on available data, and its suitability to this domain.

4. CONCLUSION

We have presented a novel method to index and efficiently retrieve very similar images belonging to the same class, while at the same time disregarding visual distortions caused by a change in viewpoint and nonessential shape deformation. Experimental results were provided to show the validity and strength of the proposed scheme.

5. REFERENCES

[1] Califano, A. and R. Mohan, *Multidimensional Indexing for Recognizing Visual Shapes*, IEEE Trans. on Patt. Anal. and Mach. Intell., April 1994, Vol. 16, No. 4, pp. 373-392.

[2] Crane, G. Editor. 2001. *The Perseus Digital Library*. Tufts Univ. http://www.perseus.tufts.edu. February 2001.

[3] Froese, R. and D. Pauly. Editors. *FishBase 2000*. http://www.fishbase.org, February 2001.

[4] Lamdan, Y., J.T. Schwartz and H.J. Wolfson, *Affine Invariant Model-Based Object Recognition,* IEEE Trans. on Robotics and Automation. Vol 6. No. 5. October, 1990.

[5] Mokhtarian, F. Abbasi S. and Kittler J. *Robust and Efficient Shape Indexing through Curvature Scale Space*. Proc. 6th Brit. Mach. Vis. Conf. Sept 1996, pp 53-62.

[6] Mundy, J. and Zisserman, A. (eds.) *Geometric Invariance in Computer Vision.* MIT Press, Cambridge, MA, 1992.

[7] Reiss, T. *Recognizing Planar Objects Using Invariant Image Features*. Springer-Verlag, Berlin, 1993.