**Using Shape Contexts For Shape Matching**

(Visual Modeling Course Project)

|  |  |  |
| --- | --- | --- |
| Amogh Param  Department of Computer Science  University of California,  Los Angeles  aparam@cs.ucla.edu | Deepthi N Rao  Department of Computer Science  University of California,  Los Angeles  deepthinrao@cs.ucla.edu | Saransh Gupta  Department of Computer Science  University of California,  Los Angeles  saranshgupta@cs.ucla.edu |

**Abstract - Images of the same shape may vary significantly in their vector representations. However, we can employ the notion of shape similarity for object recognition. We compute the shape context, which is a shape descriptor that captures the relative positions of other points on the shape contours. This gives a globally discriminative characterization of the shape and not just a localized descriptor. For matching, we find the correspondence between these points by posing it as a weighted bipartite graph-matching problem (optimal assignment). Having found the correspondence, we then use regularized thin-plate splines for finding the aligning transform that best matches the shapes. Finally, we can train a k-NN classifier that uses a shape distance metric defined in the paper for digit recognition on the MNIST dataset and compare the performance with other approaches.**

|  |  |  |
| --- | --- | --- |
| **1** | **Introduction** |  |
| **2** | **Methodology**  1. Edge Detection 2. Sampling 3. Shape Descriptors 4. Cost Matrix 5. Alignment Transform 6. Shape Distance |  |
| **3** | **Results** |  |
| **4** | **Future Work** |  |
| **5** | **Conclusions** |  |
| **6** | **References** |  |

1. **Introduction**

With more and more images being generated in digital form, there is a growing interest in finding images from large databases of images. Retrieval of images can be done based on some descriptor such as the ‘shape’ - which is an important visual feature. Matching of shapes may be a trivial task for humans, but visually similar shapes may differ significantly in their vector representations. Consider the two digits in Figure 1. When the image intensity vectors of these two digits are compared using L2 norms, they would seem very different. But, these are two different ways of writing the number 2, i.e. they belong to the same category.



**Figure 1**. Examples of two handwritten representation of the same digit which are signficantly different pixel-to-pixel

In this project, we implement shape matching using shape contexts method, which does the following tasks:

1. Extract a rich shape descriptor from the images
2. Solve the correspondence problem between two shapes,
3. Use the correspondences to find an aligning transform
4. Compute the shape distance metric that quantifies the dis-similarity between the shapes

Given the shape distance of an input image to a database of images or a set of selected prototypes that best represent each possible class of shapes, we can classify the image into the correct category by using simple algorithms like the K-nearest neighbor classifier.

1. **Methodology**

In this project, we used MATLAB for the implementation of the algorithm. As mentioned in the previous section, there are three major tasks. In this section, we lay down the finer details and all the steps involved.

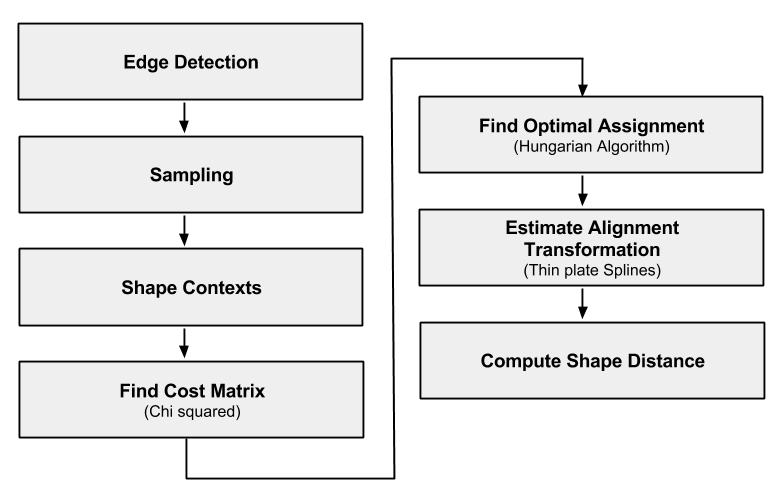
The first step is to extract the inner and outer boundary edges of the object. These boundary pixels are then sampled. Only a small number of points are used for purposes of computational tractability. The sample points are then used for the computation of the shape descriptor – the shape context. Since it is a matching problem, the shape contexts are computed for both the input image and the target prototype. We then estimate the best correspondence between points from the input image to the target image based on the chi-squared distance between the shape contexts of the two images. The thin plate spline is then used to estimate the aligning transform that best matches the sampled points on the first image to the corresponding points on the target image. Lastly, the ‘shape distance’ is calculated using the weighted sum of three different measures of dissimilarity, as detailed in the sections below.

* 1. **Edge Detection**

In image processing, an edge of an object in an image is defined as the region where a sharp change in the image brightness value is present. These pixels point out the difference between neighboring pixels. They may or may not have loops. Any edge detector algorithm may be used to obtain edges points from the image. Contour points are obtained from edges. These pixels are closed loop of points that define the boundary or the shape of objects which can be used for matching and recognition purposes. Objects with holes have multiple sets of these shape describing points. Contours can be obtained at different levels, with the height measured from x-y plane.

* 1. **Sampling**

A shape is described by a discrete set of points sampled from the internal and external contours on the object [1]. The obtained points are then sampled to get the optimal number of points that best describe the contours of the shape. This makes the computation reliable, economical and much faster. In

 **Figure 1**: Processing Steps

our experiment, we choose hundred uniformly sampled points to describe shape of the object. Assuming that the contours are piecewise smooth, in this way we can obtain a good approximation to the original continuous shapes.

* 1. **Shape Descriptors**

Given a set of sample points on an image, we wish to find a rich descriptor that encodes both the local and global information about these points. Before we describe the shape context descriptor, we briefly review one broad categorization of descriptors:

1. Brightness-based- they make more direct use of the brightness of pixels.
2. Feature-based- they involve the use of spatial arrangements of extracted features such as edge elements or junctions.

**2.3.1 Brightness based methods**

An alternative approach to feature based methods is to make direct use of the gray values within the visible potion of the object, rather than concentrating on the shape of the contour or other extracted features. One category of approaches computes the correspondences or the alignment using the gray-scale values. Other categories of brightness based methods build classifiers without explicitly finding the alignment or correspondences. Such methods rely on having a large dataset of training examples with which a learning algorithm can be trained to match shapes.

Some of the popular methods based on this approach include Principal Component Analysis (PCA), Support Vector Machines (SVM) and the LeNet classifier.

**2.3.2 Feature based methods**

Feature vectors that contain descriptors such as area or moments can be computed and then these vectors between the shapes can be compared. But such techniques often discard detailed shape information in the process. There has been a lot of research in the domain of shape similarity using silhouette images. Since silhouette images have the property of not having holes or other internal markings, the resulting boundaries are represented by a single-closed curve, which can be parameterized by arc length. Since this curve is one dimensional, a convenient solution to matching would be to use dynamic programming approaches.

Since these silhouettes ignore internal contours and are difficult to extract from real images, they are limited in their application as shape descriptors for general objects. Methods that treat the shapes as sets of points in the two dimensional image are much rather preferred as they are easier to be extracted. We do not use this method in our project because it doesn’t return correspondences. Other methods to find shape similarity include casting sample points in images into finite element spring mass models, and eventually finding correspondences by comparing modes of vibration. Yet other methods use spatial configurations of a small number of key ‘landmark’ points. These methods sacrifice the shape information available in smooth portions of the object contours.

* + 1. **Shape Context**

The shape context is a feature-based descriptor that is used for correspondence based matching of shapes. For each boundary point, we calculate the shape context, thereby reducing the problem of finding the correspondences between the boundary points as a problem of optimally assigning the shape contexts of these boundary points of the shapes to be matched.

Consider the set of vectors originating from one boundary point to all the other boundary points. This acts like a good descriptor of the shape relative to that point since it captures a lot of global information about the shape. However, as the number of sample boundary points increases, using all these vectors becomes unfeasible. In order to reduce the descriptor length, these distance vectors originating from point pi are coarsely binned in order to give a histogram hi, which represents the distribution of relative distance between the points.

The histograms are taken uniformly in the log-polar space. This makes the descriptor more sensitive to nearby points than farther away points. This makes it more robust to noise and outliers.

* 1. **Cost Matrix**

Once we compute the shape contexts, we consider the problem as an optimal assignment problem. As the shape-contexts are distributions represented as histograms, we use the chi-squared test statistic to compute the cost matrix where  
Cij = C(pi,qj) represents the cost of matching a point pi on the first shape and a point qj on the second shape.

Where, hi(k) and hj(k) denote the K-bin normalized histogram at pi and qj respectively.

* + 1. **Optimal Assignment**

Using the cost matrix we try to minimize the total cost of matching, given the set of costs, Cij, between all pairs of points pi on the first shape and qj on the second shape.

We match based on the constraint that the matching between the points is one-to-one. This optimal assignment or bipartite graph matching problem is solved in O(N3) time using the Hungarian algorithm which provides a dynamic programming solution, where the input to the algorithm is the square cost matrix and the output is a permutation, π(i), such that the above equation is minimized.

To handle cases where we have outliers, we can add “dummy” nodes to each point set with a constant matching cost of εd, which can essentially be considered as the threshold for outlier detection. A point is matched to one of these nodes when there is no real match available at a cost lower than εd. This can also be applied to the case when the cost matrices of the two shapes are not equal. We can add dummy nodes to the smaller point set and hence get two square cost matrices and proceed with the matching.

* 1. **Alignment Transform**

Given the correspondences between the boundary points based on the chi-squared distance between the shape context descriptors, we need to find a plane transformation T: R2 → R2 that can be used to map the points on the input image to the corresponding points on the target image.

One broad category of transformation models includes linear transformations, which include rotation, scaling, translation, and other affine transforms. Linear transformations are global in nature, thus, they cannot model local geometric differences between images.

The second category of transformations allows 'elastic' or 'nonrigid' transformations. These transformations are capable of locally warping the target image to align with the reference image.

Non-rigid transformations include the thin plate spline transformation (TPS) model. The thin plate spline is the 2D generalization of the cubic spline. In its regularized form, which is discussed below, the TPS model includes the affine model as a special case.

Given the displacement vectors of the points on the input image (xi,yi) to their corresponding points on the target image (xi’,yi’) with values vi , we employ two thin plates for modeling the transformations along the x and the y directions. That is, we interpolate a planar deformation field from the sparse displacement vectors.

The 2D vector field/ transformation is given by:

The Thin plate deformation energy for interpolation of the data along one direction (x or y):

This TPS model is a special case of poly-harmonic spline and has an implicit representation in the form of the radial basis function. The interpolant f(x,y) is given by :

In the above equation, the first three terms are the affine terms that correspond to translation, scaling along x direction and scaling along the y direction respectively.

Where U(r) = r2 log r, f(xi,yi) = vi and the following conditions hold true:

Thus we get a linear system of equations that can be solved for the TPS coefficients:

Where, , the ith row of P is (1, xi, yi), w and v are column vectors formed from wi and vi, respectively, and a is the column vector with elements a1, ax, ay. We will denote the (n + 3) x (n + 3) matrix of this system by L. L is nonsingular and we can find the solution by inverting L. If we denote the upper left n x n block of L-1 by A, then we have

The regularized TPS energy that is minimized for estimating f is given by:

* 1. **Shape Distance**

The shape distance metric proposed by the authors is a weighted sum of three terms:

1. Shape context distance
2. Image appearance distance
3. Bending energy

The shape context distance is the chi-squared distance between the target image and the input image after the best alignment transformation is performed.

The image appearance distance (Dac) is used since the shape context does not at all account for the gray-scale and texture information around the sampled boundary points. Dac(P,Q) is sum of the squared differences of the pixel intensity values in Gaussian windows around the points given by:

The bending energy used is that after 3 iterations of shape context matching and TPS transformation re-evaluation.

For the images of digits the weights used are 1.6, 1 and 0.3 respectively for the three terms mentioned above.

1. **Results**

We have established some results against MNIST data set of handwritten digits. MNIST dataset of handwritten digits consists of 60,000 training and 10,000 test digits.

In our experiment, we obtain the edges of the digits by applying Canny Edge Detection, and sample 100 points from these edges to represent each digit. The cost computation for bipartite matching also accounts for dissimilarity of local tangent angles. We defined the cost as:

Where is the shape context cost and , computed as gives the tangent angle dissimilarity.

Figure 2 shows a diagram of log-polar histogram bins used for computing shape contexts. We have used 5 bins for log r and 12 bins for.

The correspondences are calculated iteratively and the following correspondences are obtained. For one of the many test samples, the correspondences are obtained as shown in figure 3.

The warping of the digits is as shown in figure 4. Progressively, the images are warped in different iterations. This makes use of the thin plate spline functions for transforming coordinates.

Table 1 shows values of shape context cost and If for different iterations. In each iteration, we can see that the values are decreasing.

Figure 2. Log-polar histogram bins for computing shape context.

|  |  |
| --- | --- |
| Iteration 1 | Iteration 2 |
| Iteration 3 | Iteration 4 |
| Iteration 5 |  |

Figure 3. Correspondences of two similar digits

|  |  |
| --- | --- |
| Iteration 1 | Iteration 2 |
| Iteration 3 | Iteration 4 |
| Iteration 5 |  |

Figure 4: Different iterations of image warping

|  |  |  |
| --- | --- | --- |
| **Iter #** | **If** | **S.C cost** |
| **1** | 1000 | 0.15979 |
| **2** | 0.061765 | 0.10792 |
| **3** | 0.013989 | 0.054519 |
| **4** | 0.0066203 | 0.043877 |
| **5** | 0.0044934 | 0.039482 |
| **6** | 0.0025944 | 0.036152 |

Table 1: If and SC cost values for different iterations.

1. **Future Work**

This experiment can be extended to recognize 3D objects in the images, trademark retrieval, silhouettes, etc. In order to recognize 3D images, several images capturing an object from different views are obtained. The training data consists of equally spaced views of the object and the test set consists of remaining views. Silhouette matching uses the same approach as digit recognition. The performance of silhouette matching can be measured using “bullseye test” in which each image is used as a query and the number of correct images in the top 40 matches are counted.

1. **Conclusion**

We have demonstrated software for image segmentation and video contour tracking that uses intelligent scissors and snakes to supplement each other’s limitations, and experi- mented with combining them so that the scissors provide a nice initialization for the snake. We designed a practical GUI so that users could conveniently use the software. It was shown that as long as user finds an appropriate configuration, our software can deliver satisfactory results on biology images and videos, and it shows promise in medical applications. Finally, we have discussed an immediate po- tential application of this work to the astrophysics problem of automatically tracing the overlapping and closely-stacked coronal loops in images of the sun.