Sketch Creation utilizing Shape Matching Techniques

Ioannis Andreou, Nikitas M. Sgouros

Dept. of Technology Education and Digital Systems
University of Piraeus, Karaoli & Dimitriou 80, Piraeus, Greece
{gandreou, sgouros}@unipi.gr

ABSTRACT

This paper describes a sketch creation method that can act as a front-end to a visual image retrieval system by allowing the user to draw the types of images s/he is interested in. In addition, the method supports sketch generation processes, in general, by allowing the designer to automatically replace sketch components in the current sketch with similar components from a shape database. The method consists of a shape matching step in which the contents of the user sketch are compared with the contents of a shape database and of a placement step in which similar shapes from the database are properly aligned and replace automatically the contents of the user sketch. Shape matching uses a mixture of local and global shape features such as the turning function difference and the degree of openness of a curve.

1. INTRODUCTION

Recently there has been a growing interest in visual information retrieval technologies, i.e., systems that allow querying, storage and retrieval of visual media based on its visual content. These systems need to allow the user to sketch the major features of an image and submit the resulting drawing as a query for similar images in an image database [5]. Furthermore, in many cases such as in the creation of drafts for engineering drawings [8] or in the composition of business presentations [7] the user should be able to compose a high-quality image fast based on rough sketches of its contents. Finally, the generation of artistic works can benefit from the analysis of drawing sketches made by different artists [3].

This paper describes a sketch creation method that can act both as a front-end visual query module to visual information retrieval systems and as an aid tool for fast image composition. The method allows the user to draw shapes on the computer screen using the mouse cursor. At any time during this process the user can query an image database for similar shapes and then select the ones s/he thinks are more relevant. The system can then automatically align and replace user-drawn objects with the chosen database shapes in the user sketch.

There are two major advantages associated with the use of such a system: (a) the user does not have to be an expert designer, as s/he can always replace his/her own drawings with existing better ones, and, (b) the creation of a sketch or image that refers to a particular theme can be speeded up and made more robust by using a shape database that was specially created for this theme.

The rest of the paper is organized as follows. Section 2 presents related work, while section 3 describes in detail the sketch creation method. Section 4 describes the sketch drawing implementation. Finally, section 5 draws overall conclusions and describes future work.

2. RELATED WORK

Shape matching and retrieval approaches can be classified as either global or local. Global approaches are based on shape properties that depend on the entire shape. Local approaches depend on local structural features of each shape. Examples of global approaches include the QBIC system [5] that uses a set of algebraic moment invariants as well as perimeter and major axis orientation to describe each shape, along with Mallat [4] that uses wavelet transformations as the basis for shape description. Examples of approaches that use local features include the elastic matching of 2-D templates [1] and the curvature scale space approach [2]. Approaches using global features cannot handle occluded objects and often lack the ability to capture perceptually significant local shape information. Local feature approaches, on the other hand, can be computationally inefficient and noise sensitive. The shape matching method described in this paper attempts to overcome these problems by using a mixture of global and local features (e.g. circularity, convexity, eccentricity and turning function difference).

Sketching for querying approaches were used as part of visual image retrieval systems [5, 9-10]. Most of these approaches assume that the sketch matches quite closely the desired image and that all the elements in the drawing are valid.
3. THE SKETCH CREATION METHOD
The sketch creation method consists of the following sequence of steps:
1. Match the shapes drawn by the user to those in a shape database in order to find similar components.
2. Insert the shapes found in the database into the user sketch so that they are properly aligned with the ones defined by the user.

The following subsections describe each one of these steps in detail.

3.1. Shape matching
The shape matching method accepts as input two polygonal curves with N and M number of vertices, respectively. It then looks for similar vertex sequences in these curves. The basic comparison measure used is the absolute difference in the turning angles occurring at two consecutive edges in a polygon traversal. In order to eliminate noise, each curve is first sampled uniformly at a predefined resolution and then smoothed using median filtering or linear interpolation. In the rest of this paper, we refer to the resolution of the sampling process (or the polygon) as N.

Shape matching proceeds in three steps. The first step evaluates the overall similarity of two polygonal curves using a set of global shape features that depend on whether these curves can be considered ‘open’ or ‘closed’. More specifically, the degree of openness of a curve is computed as the ratio of the distance between the first and the last vertex of the curve to the perimeter of the curve. A curve is considered ‘closed’ only if this ratio is below a threshold. The default value for this threshold was experimentally set to 0.1 in our implementation.

In the case of two ‘closed’ curves the method computes the circularity, convexity, eccentricity and openness of each one using the closed polygon resulting from the connection of its first and last vertex. The overall similarity of such curves is estimated by comparing their circularity, convexity, eccentricity and openness. The estimation of the overall similarity of two curves where at least one is classified as ‘open’ is based on only the comparison of their openness. In either case the matching process terminates unsuccessfully if the curves differ significantly in their overall similarity.

The second step produces a number of matching vertex sequences for the two input polygons based on a comparison of their turning function difference.

The third step selects the final set of non-overlapping vertex sequences between the two shapes that will comprise the final match. The selection process favors vertex sequences that are placed within reasonable distances between one another (distance is measured as the minimum number of edges separating two regions), have significant length and their relative orientations are not very different in the two shapes. The matching score of the final set is computed as the proportion of the cardinal number of the vertices in this set to the minimum resolution of the two curves.

The shape-matching process is scale invariant for ‘closed’ curves, as the only measure that describes the size of the curve is the sampling resolution, N. However, in the case of sketching we have to consider the fact that very often the user draws an ‘open’ curve. In this case, in order to calculate the resolution of such a curve the system needs to compensate for its lack of knowledge about the actual length of the missing part of the curve. To this end, two parameter values are used: (i) Nr, a positive integer value that represents the number of the different sampling resolutions that should be used for the drawn shape, before each of the resulting polygons (from the sampling procedure) is submitted to the shape matching process, and, (ii) Vr, a positive real value, representing the possible variation of the length of the missing curve segment L, from the distance between the first and the last vertex of the drawn curve. The user can configure either of these parameters or use their default values (Nr = 3, Vr = 3.0). If N is the resolution that would be used for the shape, if it was closed, and N_i is the resolution that should be used for the i-th sampled ‘open’ curve, then N_i is calculated as:

$$N_i = \frac{i \cdot Vr \cdot Ds}{Nr}$$ (1)

where Ds is the number of vertices that the missing part of the polygon would correspond to, if the curve was ‘closed’ (by connecting the first and the last vertices of the curve) and it was sampled with resolution N. N_i is rounded into an integer value and should always be positive.

For each of the sampled versions of the query shape the retrieval engine is triggered and the final result is the merged sorted set of all the retrieved results. Increasing Nr increases the overall complexity of the search process, but also improves recall. All of the above parameters, along with the independent feature thresholds can be configured by the user. However, the search engine can also be configured automatically using information existing in the curve database file or generate appropriate configuration parameters, using only the resolution N of the database polygons.

3.2. Shape placement
For each polygonal curve drawn by the user the results of the shape matching process is a list of possible similar shapes from the database ordered according to their matching score. When the user selects one of these shapes, it is rotated and scaled to best match the
corresponding shape in the sketch. It is then drawn with
different color on top of the user-drawn curve in order for
the designer to decide whether to incorporate it in the
sketch or not.

Let’s consider the case where shape S1 selected from
the database must replace shape S2 drawn by the user
in the sketch. Shape placement starts by properly aligning
the centers of gravity for S1 and S2. To this end, their
centers of gravity (COG) are made coincident through
translation to the center of the coordinate system (0, 0).
Furthermore, the scale parameters for S1 are calculated by
dividing the dimensions of the bounding rectangles of the
two shapes.

In order to calculate the correct rotation angle for the
alignment of the two shapes we must initially calculate
their circularities (if both are classified as ‘closed’). If both
shapes are ‘closed’ and their circularities are below a
threshold (so that they do not resemble a circle) we follow
procedure 1, otherwise procedure 2.

3.2.1 Procedure 1: Closed and non-cyclical shapes
For each one of these shapes we extract the parameters of
a line (described by equation Y=a+rX), for which the
average difference of every point inside the shape from it
is minimum, using the first and second order normalized
moments N11, N20, N02. This line corresponds to the major
axis of the best matching ellipse for each shape. We
perform all these operations with each shape centered at
(0, 0), so that the line can be described by the equation
Y=rX, and r=tan(R), where R is the rotation angle. This also
enables us to optimize the moment calculation functions.
The mathematical definitions used for geometric moments
(Mpq) and normalized moments (Npq), of order p+q for the
points of an area A and the angle of rotation R that
corresponds to the Y=rX line that is the major axis of the
best matching ellipse for area A follow in Eqs. (2), (3) and
(4).

\[ M_{p,q} = \int\int_{A} x^p y^q dx dy \]  \hspace{1cm} (2)

\[ N_{p,q} = \left( \frac{M_{p,q}}{M_{0,0}} \right) \] \hspace{1cm} (3)

\[ R = \tan^{-1}\left( \frac{N_{1,1}}{N_{2,0} - N_{0,2}} \right) \] \hspace{1cm} (4)

For the type of shapes we are dealing with, Mpq is
also the central moment of the area of the shape, as we
have already translated its COG to (0, 0). Following this
computation we rotate shape S1 round its COG in order to
align the major axes for the best matching ellipse of shapes
S1 and S2. In order to find the appropriate rotation angle
we rotate S1 by angle (R2-R1), where R1, R2 are computed
using Eq. (4) for the shapes S1 and S2 respectively. We
then use information from the matching process to see if
this is the correct angle. For each vertex v1 of shape S1
matched to a vertex v2 of shape S2 we check to see if they
lie on the same sides of their (now) common major axis. If
this is the case for most of the vertex pairs the angle is
correct. Otherwise, the angle of rotation for S1 is \( \pi - (R2-
R1) \).

3.2.2 Procedure 2: All other shape types
For all other shape types, the alignment procedure is
simpler. Both shapes are again centered at (0, 0). For each
vertex v1 of S1 matched against vertex v2 of S2, we
calculate angle D, using the angles F1, F2 between x-axis
and v1 and v2, respectively, as defined below:

\[ D = \begin{cases} 
(2 \pi) + D1 & , \quad D1 < -\pi \\
(2 \pi) - D1 & , \quad D1 > \pi \\
D1 - \pi & , \quad -\pi \leq D1 \leq \pi 
\end{cases} \] \hspace{1cm} (5)

D, as defined above, provides us with the angle,
in radians, that we must rotate v1 around (0, 0) to align it

Figure 1: Screenshot of the sketch drawing application.
The left panel contains run-time configuration options.
The top panel contains possible matches returned for the
current item (center). The curve that was selected by the
user (the leftmost one in the top panel) was placed with a
dotted outline on top of the current sketch curve, using
the placement procedure.
with v2. The appropriate rotation angle for the whole shape, \( R' \) is calculated as follows (the sums are calculated using all the matching pairs returned by the shape matching process):

\[
S = \sum D \\
\sum F(D, S) = R' \quad (7)
\]

where:

\[
F(D, S) = \begin{cases} 
2*\pi - D, & S < 0, D > 0 \\
2*\pi + D, & S > 0, D < 0 \\
D, & \text{otherwise}
\end{cases} \quad (9)
\]

We must note that this procedure is faster but it is not always accurate. For example, the result of the sampling process for the shapes might not have resulted in vertex points correctly spaced or the vertex matches returned by the matching process might not have been enough in cardinality or coverage of the shapes' outline. Furthermore, the calculation of COGs is not always accurate if at least one of the two shapes is 'open'. In some cases, the user has to intervene and correct the rotation and scaling proposed in this step.

4. IMPLEMENTATION

The sketch creation application was developed using the C++ programming language and the Windows Programming Interface (‘WINAPI’), utilizing features of Silicon Graphics template library (SGI-STL). We tested this application on a 1GHz Pentium3 PC with 128 MB of memory, running Windows 2000 with a database containing approximately 100 shapes based on the Squid database for marine life, and with enabling an option that would query the database for matches and draw the results each time the current shape was created, selected or changed by the user. The latency of the search process was satisfactory (below 20ms, although most of this latency is caused by internal operating system delays (e.g., message passing)).

5. CONCLUSIONS & FUTURE WORK

This paper described an integrated system for sketch drawing and sketch retrieval. The user of the system may draw individual sketch elements and then compare them against a database of such elements. Thus, s/he may find the database items that are most relevant to his/her sketch elements (polygonal curves), and use them to enhance or correct the final result, or view the source sketch of this element. Such a system can improve the ease and correctness of creating designs that apply to specific themes such as engineering or scientific drawings and enable graphics creators to automate parts of their work.

An interesting and challenging extension of this work is the combination of the shape matching method with image segmentation and feature extraction methods. If an image could be described as a set of (‘open’ or ‘closed’) curves, colors and primitive textures, then the shape matching method could also be applied to images, rather than sketches. Thus, the user of a sketch-drawing application could create high-quality multimedia content, by adding image components to his/her drawings.

6. REFERENCES


