SHAPE RECOGNITION BASED ON THE MEDIAL AXIS APPROACH

Nualsawat Hiransakolwong  Khanh Vu  Kien A. Hua  Sheau-Dong Lang

School of Computer Science
University of Central Florida
Orlando, FL 32816-2362, U.S.A.

Abstract
We propose a novel, shape-matching algorithm using skeletal graphs. The topology of skeletal graphs is captured and compared at the node level. Such graph representation allows preservation of the skeletal graph’s coherence without sacrificing the flexibility of matching similar portions of graphs across different levels. By using appropriate sampling resolution, we are able to achieve a high recognition rate, and at the same time, significantly reduce space and time complexity of matching. We tested our approach against the Directed Acyclic Graph (DAG) method on noisy graphs and occluded or cluttered scenes. The results show that our approach is an effective and efficient technique for shape recognition.

Keywords: Skeletal graph, graph matching, shape recognition, shock graph

1. Introduction
Although there has been extensive work in the area of shape representation and matching, shape recognition is still an open research problem. Many shape-matching approaches have emerged, but high space, time complexity and moderate recognition rate are still the limiting factors for their acceptance.

Some shape-recognition techniques are known as curve outline-based matching methods [3]. They often suffer from one or more of the following drawbacks: asymmetric treatment of the two curves, sensitivity to sampling, lack of rotation, scaling invariants and sensitivity to articulations and deformations of parts. In [4], the point sets are matched, which gives the advantage of not requiring ordered boundary points, but the match does not necessarily preserve the coherence of shapes in that the relationship among portions of shape in the process of matching may not be preserved.

Zhu and Yuille [5] have proposed a framework (FORMS) for matching animate shapes by comparing their skeletal graphs, the medial axis of shapes. However, the applicability to inanimate objects is limited due to the choice of primitives used in modeling. A variant of the medial axis is the shock structure, which is obtained by viewing the medial axis as the locus of singularities (shocks). Many approaches to shock graph matching have emerged. However, these approaches have not been extensively tested on noisy graphs, occluded scenes or cluttered scenes.

In this paper, we propose a technique that significantly reduces space and time requirements while improving recognition rate. Our approach is based on many-to-many graph matching of skeletal graphs constructed from shapes. The matching algorithm is shown robust to scaling, rotation and translation. It is less sensitive to noise and occlusion. We show, by experimental results, that our method is significantly better than a recently proposed alternative shock graph represented in the Directed Acyclic Graph (DAG) method [1]. We present our technique in the next section.

2. Skeletal-Graphs Matching
A skeleton is an undirected tree captured from a silhouette of a binary image using the medial-axis method [6], see Figure 1. A skeleton tree consists of nodes and edges. There is a length associated with each edge. Each node has a maximum of eight connected neighbors. The angle between edges is a multiple of 45 degrees. The level of a node is defined as how close it is to the center of the tree, where a center is a vertex such that the maximum value of the distances between $u$ and all other vertices is a minimum. It is a well-known fact a tree has either a single center or two centers connected via an edge [9]. In our studies that follow, a skeletal tree will be represented from the leaf nodes toward the center(s). The “root” of this representation corresponds to a center node. In the case of two centers, selecting either as root node will determine the level of all nodes in the skeleton.

In order to represent a skeletal graph, we capture the topology of its individual nodes, i.e., the nodes’ connected neighbors, the relationship with each neighbor and the length of the edges. We construct a signature table to keep information regarding the topology of nodes in the graph. There is only one table to store the nodes’ information for the entire dataset.

The signature table consists of three attributes: signature, ID and “node detail”. Thus, a row in the table is a 3-tuple. ID is a positive integer assigned to each signature. It serves as an identification of a node’s topology. A signature consists of eight fields, corresponding to eight possible connected neighbors. Each field identifies the existence of a connected
neighbor, the topology via an ID and the relationship with the node. A field containing 0 signifies there is no connected neighbor at that angle. A field of 3, for example, indicates there is a connected neighbor whose topological ID is 3, while a field of “P” implies the connected neighbor is its parent node. The fields are filled starting with P (the parent) or the highest ID for root nodes and going in the clockwise direction.

Node detail captures the fact that there could be many nodes with the same signature and ID. It is a linked list of nodes, each in the form node/length/parent_node/image_number (list of children nodes or parent). length is the length of the edge connecting the node with its parent_node.

Table 1 shows the signature table filled with node information of the two example skeleton trees, Figures 1 (a) and (b). The first row encodes the information for all leaf nodes. They have signature (P,0,0,0,0,0,0,0), which is assigned ID = 1. Node detail is a linked list of all leaves in both trees in the order of processing. The next type of node, whose ID = 2 has signature (1,1,1,0,1,0,0,0). There is only one node, node 2 of Figure 1 (a), of this type. Therefore, node detail contains 2/0/0/1a (3,4,5,1). The signature and the node detail indicate that node 2 of Figure 1 (a) is the root node (length=0, and no parent), whose children nodes are nodes 3, 4, 5 and 1, counting clockwise.

The last row of Table 1 describes node 3 of Figure 1 (b). This node is the root node and has three children nodes: 5, 2 and 4. Its signature indicates that its children node 5 is a node of ID=4, node 2 of ID=3 and node 4 of ID=1, see Figure 1 (b).

Table 1: The signature table for Figures 1 (a) and (b).

<table>
<thead>
<tr>
<th>Signature</th>
<th>ID</th>
<th>Node detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>(P,0,0,0,0,0,0,0)</td>
<td>1</td>
<td>1/74/2/1a (2), 3/68/2/1a (2), 4/37/2/1a (2), 5/23/2/1a (2), 1/28/2/1b (2), 4/75/3/1b (3), 6/60/5/1b (5), 7/62/5/1b (5), 8/60/2/1b (2)</td>
</tr>
<tr>
<td>(1,1,1,0,1,0,0,0)</td>
<td>2</td>
<td>2/0/0/1a (3,4,5,1)</td>
</tr>
<tr>
<td>(P,0,1,0,0,0,1,0)</td>
<td>3</td>
<td>2/14/3/1b (3,8,1)</td>
</tr>
<tr>
<td>(P,0,1,0,1,0,0,0)</td>
<td>4</td>
<td>5/22/3/1b (3,6,7)</td>
</tr>
<tr>
<td>(4,0,3,0,0,1,0)</td>
<td>5</td>
<td>3/0/0/1b (5,2,4)</td>
</tr>
</tbody>
</table>

Table 2: The signature table for Figure 1(c) with node 3 as root.

<table>
<thead>
<tr>
<th>Signature</th>
<th>ID</th>
<th>Node detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>(P,0,0,0,0,0,0,0)</td>
<td>1</td>
<td>1/28/2/1c (2), 4/75/3/1c (3), 5/22/3/1c (5), 6/60/2/1c (2)</td>
</tr>
<tr>
<td>(P,0,1,0,0,0,1,0)</td>
<td>3</td>
<td>2/14/3/1c (3,6,1)</td>
</tr>
<tr>
<td>(3,0,0,1,0,1,0)</td>
<td>6</td>
<td>3/0/0/1c (2,4,5)</td>
</tr>
</tbody>
</table>

2.2 Matching Shapes

The idea behind our shape recognition is as follows. When a skeleton is a 2-root type, there are two possible configurations (Figure 1 (c)). To ensure matching, e.g., between a 2-root query and a 2-root skeleton, both configurations of either need to be compared. We chose to code one configuration for each database skeleton, either 1-root or 2-root. 2-root queries will be coded in both configurations. We use signatures to identify two types of matches: an exact match for two nodes of the same topology and at the same level, and a close match for two nodes of similar topologies at any level. Exact matches can be declared when the two signatures are the same.

Close matches are ones when query node’s children match some children of the DB node. For example, query signature (P,0,0,0,1,0,1,0) is a close match with the DB signature (P,0,1,0,1,0,1,0). When the query node is a root node, we compare the DB signature with all circular-shiftings of the query signature. For example, query’s signature (3,0,0,0,1,0,1,0) is shifted to (1,0,3,0,0,1,0), where a close match with DB signature (4,0,3,0,0,0,1,0) is declared. For a close match, we update the order of the children in “node detail” to reflect the change in its signature. The matching algorithm is summarized as follows:

1. Clear all marks on the DB signatures.
2. While there are query signatures, do
   i. Select one query signature,
ii. Identify all DB exact and close matched signatures,
iii. Mark all DB matches accordingly.
iv. Remove the query signature.

To filter out unlikely (close) matches, we use (DB node length)/(query node length) ratios and compute their median. Matches whose ratio is outside the range of median ratio – threshold, median ratio + threshold are eliminated. For each of the remaining matching pairs, exact or close match, we compute the matching score:

$$\text{Score} = w \frac{CN(T_1, T_2)}{\max(|T_1|, |T_2|)} + (1-w) \frac{CP(T_1, T_2) + 1}{\max(|T_1|, |T_2|)}$$

where CN is the number of matched nodes, CP the number of matched nodes whose ratios in the range of median ratio – threshold, median ratio + threshold, |T_j| is the size of query skeleton tree T_j, and |T_2| is of DB skeleton tree T_2. w is the weight in [0,1], which is set to adjust the relative significance of the two terms. In our experiments, we set w = 0.5 and threshold = 0.25.

2.3 Properties of the Algorithm

To prove the properties of our algorithm, we define our distance measure equivalent to the scoring function:

$$d(T_1, T_2) = 1 - \text{score} = 1 - \frac{CN(T_1, T_2)}{\max(|T_1|, |T_2|)} - (1-w) \frac{CP(T_1, T_2) + 1}{\max(|T_1|, |T_2|)}$$

**Theorem 1:** For any skeleton trees T_1, T_2 and T_3 the following properties hold true. (See proof at [8].)

i. 0 ≤ d(T_1, T_3) ≤ 1
ii. d(T_1, T_2) = 0 ⇔ T_1 is isomorphic to T_2 and vice versa
iii. d(T_1, T_2) = d(T_2, T_3)
iv. d(T_1, T_3) ≤ d(T_1, T_2) + d(T_2, T_3)

Theorem 1 indicates that our measure is a distance metric. It is also clear that our matching procedure is O(n^2) for larger block sizes. 4x4 helps remove noise and smooth out the skeletal graphs. Recognition rates, however, deteriorate rapidly for larger block sizes. This is attributed to the fact that the skeletal graphs now become too coarse to capture shapes in sufficient details. Since block size 4x4 yields the highest recognition rate at reasonable processing time, we chose to use this block size for our technique in subsequent experiments.

3.3 Comparative Study

In this experiment, we evaluated the performance of our technique and the DAG on a set of unoccluded queries. We used the same dataset as the DAG’s, consisting of up to 1,408 views of 11 objects and each object having 128 views. The results are summarized below (see Figure 3 and [8]):

- Our recognition rate is consistently better than the DAG’s as the database size increases.
- For unoccluded queries our method achieves higher recognition rate faster than the DAG does, as the number of views per object increases.
- For occluded queries, the recognition rates comparably reduce as the percentage of occlusion grows. Both methods yield recognition of greater than 70, even for 50% occluded objects.

Figures 4 to 7 show the results of some example queries against our database of about 4,000 images. For each query, top-ranked images are displayed along
with their scores. The higher the score is, the closer the match is. Table 3 summarizes the results of these queries.

Observe that the ranks of relevant images are very high, and the images are views at various angles and scaling of the queried object. In the last query, see Figure 7, we wanted to show that for heavily occluded queried objects (dog’s legs are occluded), our approach is still effective to achieve high recall (compared with the recall in the previous query: “unoccluded dog”, see Table 3).

<table>
<thead>
<tr>
<th>Figure</th>
<th>Query</th>
<th>Relevant</th>
<th>Retrieved</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Human</td>
<td>63</td>
<td>58</td>
<td>92%</td>
</tr>
<tr>
<td>5</td>
<td>Dolphin</td>
<td>52</td>
<td>46</td>
<td>88%</td>
</tr>
<tr>
<td>6</td>
<td>Dog</td>
<td>28</td>
<td>26</td>
<td>92.8%</td>
</tr>
<tr>
<td>7</td>
<td>Occluded dog</td>
<td>28</td>
<td>26</td>
<td>92.8%</td>
</tr>
</tbody>
</table>

Table 3 Summary of the results of these queries

4. Conclusions

Shape matching is an important yet open problem. We proposed a matching algorithm for skeletal graphs, which are used to represent shapes. The topology of skeletal graphs is captured and compared at the node level. The present technique achieves a high recognition rate, and at the same time, significantly reduces space and time complexity of matching. The results show that our approach is an effective and efficient technique for shape recognition.

5. Reference


