Optimal Algorithm for Convexity Measure Calculation

Alexander Kolesnikov, Pasi Fränti
Speech & Image Processing Unit
Department of Computer Science
University of Joensuu
Joensuu, Finland
{koles, franti}@cs.joensuu.fi

Abstract—Recently a new convexity measure has been proposed based on approximation of input contour with a convex polygon. In this paper, an optimal algorithm is proposed for the construction of the convex polygon. The introduced algorithm provides exact value of the convexity measure and can therefore be used for evaluation of faster heuristic algorithms.

Keywords—convexity; polygonal approximation; dynamic programming

I. INTRODUCTION

Convexity is an important characteristics of contours used in shape analysis, pattern recognition and image data retrieving [1-5]. For example, it has been used as a border irregularity measure in medical image analysis [6].

A new convexity measure of digital contours has been recently proposed in [7]. The main idea of the measure is to construct a convex polygon \( P \) so that it best fits a given input contour \( Q \) in the sense of maximizing overlap of the polygons \( Q \) and \( P \), or equivalently, minimizing the area of the region bounded between the polygons \( P \) and \( Q \), see Fig. 1 for an example. Experiments indicated that the convexity measure is more robust and symmetric in comparison to traditional convexity measures based on convex hull of the input contour \( P \).

![Figure 1. Input contour \( P \) and its convex approximation \( Q \) (left), mismatch region \( X=(P \ XOR \ Q) \) is shaded as gray color (right).](image)

Rosin and Mumford [7] recognized that the problem of finding the approximating convex polygon \( Q \) is not trivial. Furthermore, they proposed to use a genetic algorithm to find a heuristic solution. This approach provides reasonably good approximations when the objects are highly convex but they can be far away from the optimal when the shape of the object is more complex and less convex.

In this paper, we propose an optimal algorithm for constructing the approximating polygon \( Q \) in order to provide exact measurement of the proposed convexity function. The proposed algorithm is useful when the exactness of the convexity measure is needed either for the sake of accuracy, or as a point of comparison for evaluating the usefulness of faster heuristic algorithms.

II. CONVEX POLYGONAL APPROXIMATION

A. Problem Formulation

A closed \( N \)-vertex polygonal curve \( P \) in two-dimensional space is defined as the ordered set of vertices \( P = \{p_1, p_2, \ldots, p_N\}, \text{ } p_{N+1} = p_1 = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{N+1}, y_{N+1}) = (x_1, y_1)\} \). Let us construct an approximating convex polygon \( Q = \{q_1, q_2, \ldots, q_M\} \) so that it best fits the input contour \( P \) in terms of maximizing overlap of the polygons \( Q \) and \( P \), or equivalently, minimizing the area \( S_X \) of the mismatch region \( X \) of polygons \( P \) and \( Q \). The mismatch region \( X \) is bounded between the two polygons \( P \) and \( Q \):

\[
X = P \ XOR \ Q \ P \ AND \ Q,
\]

and approximation error is defined as the area of \( X \):

\[
S_X = \text{Area}(X) - \text{Area}(P \ XOR \ Q).
\]

The convexity measure \( C_Q \) is then defined as a function of the area \( S_Q \) of the approximating convex polygon \( Q \), and the area \( S_X \) of the mismatch region \( X \):

\[
C_Q = \left(1 + \frac{S_X}{S_Q}\right)^{-1}.
\]

The problem of exact of calculation of \( C_Q \) is then reduced to finding the optimal approximation \( Q \) for the given input shape \( P \) so that the \( S_X \) is minimized:

\[
S_X^{(\text{opt})} = \min_Q \left\{ \text{Area}(P \ XOR \ Q) \right\}.
\]

The problem was attacked by genetic algorithm in [7]. However, it is not feasible to develop complicated
heuristics for the problem as they do not guarantee the optimally, and therefore, the exact calculation of the convexity measure. Instead, optimal solution can be developed based on dynamic programming (DP).

B. Dynamic Programming Approach

At first, we introduce an optimal algorithm for convex polygonal approximation using a fixed starting point. We then extend this approach to the problem under consideration.

The exact solution of the optimization problem (4) with the present constraint can be found using dynamic programming for solving shortest path in a weighted graph. A full graph \( G(P) \) is first constructed on the vertices of \( P \) so that the nodes \( V = \{v_1, v_2, \ldots, v_N\} \) of the graph correspond to the vertices of the curve \( P \). Weight \( w(i,j) \) of the edge \((v_i, v_j)\) \( G(P) \) is defined as the area-based approximation error \( S_x(p_i, p_j) \) of the curve segment \( P_{ij} = (p_i, p_j) \) by a linear segment \( L_{ij} = (p_i, p_j) \):

\[
S_x(p_i, p_j) = \sum_{u=0}^{N} |S_u(r_u, r_{u+1})| \quad (5)
\]

where \( \{r_u\} \) are the cross-points of the curve segment \( P_{ij} \) with line segment \( L_{ij} \), see Fig. 2. Both the cross-points and the area \( S_x(p_i, p_j) \) of the polygon can be calculated in \( O(N) \) time. Thus, the time complexity of the error calculation for one segment is \( O(N) \).

Algorithm for the shortest path in a graph was used for polygonal approximation of curves with minimum number of segments for a given maximum deviation \([8, 9]\). We follow the approach and introduce the cost function \( C(n) \) as the area-based minimum approximation error of curve segment \( P_n = \{p_1, p_2, \ldots, p_n\} \) with convex curve \( Q_n = \{q_1, q_2, \ldots, q_n; q_0 = p_1; q_n = p_n\} \):

\[
C(n) = \min_{i=1}^{n} \sum_{i=1}^{n} S_x(q_i, q_{i+1}) \quad (6)
\]

Recursive equation for calculating the function can be derived as follows:

\[
C(n) = \min_{i=1}^{n} \left[ C(j) + W(j, n) \right]; \quad n = 1, 2, \ldots, N.
\]

\[
B(n) = \arg \min_{i=1}^{n} \left[ C(j) + W(j, n) \right];
\]

The algorithm solving the recursion by dynamic programming in bottom-up manner is sketched in Fig. 3. Parent nodes are stored in the array \( B(n) \) to backtrack the solution. In order to satisfy the constraint of convexity, we eliminate non-convex partial solutions from consideration: for all sub-tasks the partial solutions are backtracked to the beginning of the curve \( P \) to check its convexity.

The complexity of the backtracking and the convexity testing is \( O(M) \), where \( M \) is the number of approximating nodes in the polygonal curve \( Q \). In the worst case, when almost all vertices of \( P \) belongs to the convex hull of \( P \), the time complexity of the testing procedure is \( O(N) \).

The complexity of the core algorithm for the shortest path calculation is \( O(N^3) \), the complexity for calculating the approximation error is \( O(N) \), and therefore, the total time complexity of the proposed algorithm with fixed starting point is \( O(N^3) \).

```
ShortestPath(s(i,j);double):H:integer
FOR j = 2 TO N DO
    C(j)= 1;
    C(1)=0;
    FOR j = 1 TO N DO
        FOR i = j-1 DOWNTO 1 DO
            C(j)=\min_{i=1}^{j} \left[ C(i) + W(i, j) \right];
        ENDFOR
    ENDFOR
    IF(C(i)+w < C(n)) THEN
        H = BacktrackSubpath();
        w = s(i,j);
        ELSE
            w = ;
        ENDIF
    IF(C(n)+w < C(n)) THEN
        C(n) = C(n) + w;
        B(n) = j;
        ENDIF
ENDFOR
ENDFOR
// Backtrack solution
H(1) =N
REPEAT
    H(m+1) = B(H(m));
    m=m+1;
UNTIL(H(m) > 1)
```

Figure 3. Pseudo code of the algorithm for solving the shortest path problem in a weighted graph with the constraint of convexity.
Figure 4. Results of the optimal (above) and genetic algorithm (below) approximation with convex polygon for test set #1 with the corresponding values of the convexity measure function $C_Q$. Approximation nodes of the polygon $Q$ are labeled with circles.

Figure 5. Results of the optimal (above) and genetic algorithm (below) approximation with convex polygon for test set #2 ("greebles") with the corresponding values of the convexity measure function $C_Q$. Approximation nodes of the polygon $Q$ are labeled with circles.

TABLE 1. The number of vertices $N$ and area $S_P$ of input polygon $P$; the number of vertices $M$ of approximating polygon $Q$ and processing time for the optimal algorithm. Area $S_Q$ of approximating polygon $Q$, approximation error $S_X$, fidelity of approximation $F$ and convexity measure $C_Q$ for optimal algorithm (Opt.A) and heuristic genetic algorithm (GA) [7], respectively.

<table>
<thead>
<tr>
<th>#</th>
<th>$N$</th>
<th>$M$</th>
<th>$T$, (s)</th>
<th>$S_Q$</th>
<th>Approx. error $S_X$</th>
<th>Fidelity $F$, %</th>
<th>Convexity $C_Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>390</td>
<td>9</td>
<td>19.6</td>
<td>2071</td>
<td>1409</td>
<td>100%</td>
<td>99.3%</td>
</tr>
<tr>
<td>2</td>
<td>598</td>
<td>10</td>
<td>70.3</td>
<td>5637</td>
<td>1460</td>
<td>100%</td>
<td>98.8%</td>
</tr>
<tr>
<td>3</td>
<td>604</td>
<td>7</td>
<td>67.5</td>
<td>4317</td>
<td>1943</td>
<td>100%</td>
<td>99.9%</td>
</tr>
<tr>
<td>4</td>
<td>1147</td>
<td>19</td>
<td>479.2</td>
<td>6326</td>
<td>8409</td>
<td>100%</td>
<td>28.3%</td>
</tr>
<tr>
<td>5</td>
<td>1190</td>
<td>10</td>
<td>530.9</td>
<td>686</td>
<td>3324</td>
<td>100%</td>
<td>73.8%</td>
</tr>
<tr>
<td>6</td>
<td>1234</td>
<td>5</td>
<td>551.4</td>
<td>7226</td>
<td>6368</td>
<td>100%</td>
<td>88.2%</td>
</tr>
<tr>
<td>7</td>
<td>339</td>
<td>11</td>
<td>13.7</td>
<td>5398</td>
<td>900</td>
<td>100%</td>
<td>97.6%</td>
</tr>
<tr>
<td>8</td>
<td>341</td>
<td>19</td>
<td>13.4</td>
<td>5816</td>
<td>592</td>
<td>100%</td>
<td>98.0%</td>
</tr>
<tr>
<td>9</td>
<td>332</td>
<td>20</td>
<td>12.4</td>
<td>4884</td>
<td>554</td>
<td>100%</td>
<td>96.7%</td>
</tr>
<tr>
<td>10</td>
<td>299</td>
<td>15</td>
<td>8.7</td>
<td>4579</td>
<td>504</td>
<td>100%</td>
<td>98.3%</td>
</tr>
<tr>
<td>11</td>
<td>313</td>
<td>19</td>
<td>10.1</td>
<td>4792</td>
<td>485</td>
<td>100%</td>
<td>96.2%</td>
</tr>
<tr>
<td>12</td>
<td>344</td>
<td>20</td>
<td>13.0</td>
<td>4661</td>
<td>545</td>
<td>100%</td>
<td>94.0%</td>
</tr>
</tbody>
</table>
C. Optimal Algorithm for Closed Curves

To find the optimal convex approximation for closed contour \( P \), we have to find the optimal location of the starting point as well. In principle, we have to test all vertices of \( P \) as a starting point for \( Q \) in order to find global minimum approximation error for \( S_X \). The complexity of this kind of full search would be \( O(N^3) \) because of \( N \) runs of the DP algorithm for fixed starting point of complexity \( O(N^2) \).

Fortunately, the time complexity of the algorithm with full search can be reduced by cost of \( O(N^3) \) space complexity. At the first run of the DP algorithm, the calculated weights of edges can be stored in a 2-D array of size \( N \times N \) to be used in the subsequent \((N-1)\) runs. Time complexity of the first run with error calculations is \( O(N^3) \).

In the subsequent \((N-1)\) runs, we do not need to calculate the weights again, but the convexity of solutions of sub-tasks has to be tested anyway. Complexity of the convexity testing algorithm is \( O(M) \), that gives \( O(N) \) in the worst case, if almost all vertices of polygon \( P \) lay on the convex hull of \( P \). But, on the other hand, in this case the global minimum of the cost function can be found faster, because the number of optimal starting points is \( O(N) \), too. The time complexity of one such run is between \( O(N^3) \) and \( O(N^3) \) (the worst case), or \( O(N^3) \) and \( O(N^3) \) for \((N-1)\) runs.

The complexity of the DP algorithm for optimal approximation of closed contour with convex polygon varies from \( O(N^3) \) to \( O(N^3) \) (the worst case).

III. RESULTS AND DISCUSSIONS

Experiments were performed with two sets of digitized shapes (see Fig. 4 and 5). Analysis of the algorithm for the test shapes show that the complexity of the optimal algorithm is \( O(N^3) \), because the number of approximating segments \( M \) is small \((M=5..20)\) in relative to the number of vertices \( N \) in the shapes \((N=300..1200)\).

The quality of the solution obtained by genetic algorithm [7] was evaluated by calculating fidelity \( F \), defined as [10]:

\[
F = \frac{S_X^{(opt)}}{S_X} \times 100\% ,
\]

where \( S_X^{(opt)} \) is the value of the approximation error value obtained by the optimal algorithm, and \( S_X \) by any heuristic non-optimal algorithm under consideration.

Experimental results in Table 1 show that the genetic algorithm can provide results for the approximation error \( S_X \) that are quite close to the optimal one. The processing time for genetic algorithm typically took a few seconds [7]. In some cases, however, the genetic algorithm was stuck in a local minimum. Usually it happened with shapes of low convexity measure \((C_Q<0.5)\) and with a large number of vertices.

The convexity measure depends on the area of the approximation polygon \( Q \) as well. In some cases, non-optimality of the approximation error \( S_X \) was compensated by the increase of the convex polygon area \( S_Q \) so the convexity measure \( C_Q \) is quite close to those obtained with the optimal algorithm. But in some cases, non-optimality of the approximating convex \( Q \) affects on value of the convexity parameter \( C_Q \), even the area \( S_Q \) is close to the minimal value.

IV. CONCLUSIONS

To calculate area-based convexity measure for contours, an approximating convex polygon with minimum mismatch area has to be constructed. Optimal algorithm for the approximating convex polygon construction is proposed. It provides the exact value of the convexity measure and can be used for evaluation of faster heuristic algorithms. The complexity of the introduced algorithm is \( O(N^4) \).

REFERENCES