Pseudocode

邱欣欣

May 27, 2015

Algorithm 1 Space-based Depth-First Search (SDFS)

Input: The feature points P, their number n and connectivity matrix M in the binary skeleton image.
Output: The cycles with their feature points and connectivities in the binary skeleton image.

1: **for** each point $a_i \in P$ **do**
2:   **for** each point $b_j \in P$ **and** connects to a_i **do**
3:     clear $CyclePath$ for initialization
4:     add points a_i and b_j to $CyclePath$ in turn
5:     **while** $n--$ **do**
6:       $p_0 \leftarrow CyclePath(end - 1)$
7:       $p_1 \leftarrow CyclePath(end)$
8:       calculate vector $V_{p_1 p_0} = V_{p_0} - V_{p_1}$
9:       **for** each point $p_{2_k} \in P$ **and** connects to p_1 **do**
10:         **if** $V_{p_1 p_0}.x \times V_{p_1 p_{2_k}}.y - V_{p_1 p_0}.y \times V_{p_1 p_{2_k}}.x > 0$ **then**
11:           add p_{2_k} to $RightHandPoints$
12:         **else if** $V_{p_1 p_0}.x \times V_{p_1 p_{2_k}}.y - V_{p_1 p_0}.y \times V_{p_1 p_{2_k}}.x < 0$ **then**
13:           add p_{2_k} to $LeftHandPoints$
14:         **else**
15:           add p_{2_k} to $CyclePath$
16:         **end if**
17:       **end for**
18:       **if** $RightHandPoints$ is not empty **then**
19:         **for** each point p_{2_m} in $RightHandPoints$ **do**
20:           calculate $\theta_{p_0 p_1 p_{2_m}} = \arccos \frac{V_{p_1 p_0} \cdot V_{p_1 p_{2_m}}}{|V_{p_1 p_0}||V_{p_1 p_{2_m}}|}$
21:         **end for**
22:         add p_{2_m} determined by $\min\left(\theta_{p_0 p_1 p_{2_m}}\right)$ to $CyclePath$
23:       **else if** $LeftHandPoints$ is not empty **then**
24:         **for** each point p_{2_n} in $LeftHandPoints$ **do**
25:           calculate $\theta_{p_0 p_1 p_{2_n}} = \arccos \frac{V_{p_1 p_0} \cdot V_{p_1 p_{2_n}}}{|V_{p_1 p_0}||V_{p_1 p_{2_n}}|}$
26:         **end for**
27:       add p_{2_n} determined by $\min\left(\theta_{p_0 p_1 p_{2_n}}\right)$ to $CyclePath$
28:       **end if**
29:       **if** $CyclePath$'s point number is greater than 2 **and** last point is the same as first point **then**
30:         **if** $checkCyclePath(p_0, p_1, p_2, ...)$ **then**
31:           output $CyclePath(p_0, p_1, p_2, ...) \Rightarrow Cycles$
32:           break
33:         **end if**
34:       **end if**
35:     **end while**
36:   **end for**
37: **end for**
38: remove the duplicated $CyclePath$s from $Cycles$