Project: “Recognition of digital circle”

Introduction

The objective of this project is to implement a linear programming based algorithm for the recognition of digital circles.

We expect from you:

- A short report with answers to the “formal” questions and a description of your implementation choices and results.
- A C++ project (`CMakeLists.txt` plus several **commented** cpp program files).

1 Digital disk

Let $I \subset \mathbb{Z}^2$ be a rectangular digital image. Let $Z \subset I$ be a digital set and $\bar{Z} = I \setminus Z$ be its complement set.

A digital set Z is a *digital disk* if and only if there exists a circle of center $\omega \in \mathbb{R}^2$ and of radius $\rho \in \mathbb{R}$ such that:

$$\begin{cases} \forall z \in Z, \ \|z - \omega\|^2 \leq \rho^2 \\ \forall z \in \bar{Z}, \ \|z - \omega\|^2 \geq \rho^2 \end{cases} \tag{1}$$

Question 1 *Let us consider the Gauss digitization of a convex shape $X \subset \mathbb{R}^2$ at gridstep h:*

$$Dig(X, h) = X \cap (h \cdot \mathbb{Z}^2).$$

Show that if X is an Euclidean disk, then $Dig(X, h)$ is a digital disk. Show however that the converse is not true.

Question 2 *There exists a unique circle passing by three digital points. Show that we can test whether another digital point lies INSIDE, ON or OUTSIDE such a circle with integer-only computations and without explicitly computing its center and radius. You may have a look at “in-circle test”, broadly used in computational geometry, e.g. to compute the Delaunay triangulation of a point set.*

Question 3 *Implement a function that checks whether two given digital sets are separated by a given circle passing by three digital points or not.*

2 Linear programming

Let us now consider algorithm 1 (which uses routine 2). It is a randomized and recursive algorithm that checks whether two point sets are separable by a circle in expected linear-time (see [Sei91] or [dBCvKO00][chapter4]).

The union of the inner point set, denoted by S^-, and the outer point set, denoted by S^+, is merely denoted by S. All the points of S are numbered from 1 to $|S|$, the size of S. The idea consists in maintaining a separating circle while iterating over the points $s_i \in S$ for i from 1 to $|S|$. For each point s_i, three cases may occur:

- if it belongs to S^- (resp. S^+) and it is located INSIDE (resp. OUTSIDE) or ON the current separating circle, there is nothing to do.

- Otherwise (lines 6-9 of algorithm 2):
    1. Either the two input sets are not circularly separable at all,
    2. or there exists a separating circle passing by s_i.

In the aim of deciding between these last two alternatives, the set of possible separating circles is restricted to circles passing by s_i and the same algorithm is recursively called from 1 to i (line 9 of algorithm 2). At each recursive call, the set of possible separating circles is restricted so that the base case involves a unique circle passing by three given points and consists in checking whether it separates S^- from S^+ or not (lines 11-17 of algorithm 2).

Algorithm 1: areCircularlySeparable(S^-, S^+, p_1, p_2, p_3)

Input: $S^-, S^+ \subset \mathbb{Z}^2$, the inner and outer point sets
$p_1, p_2, p_3 \in \mathbb{Z}^2$, three points characterizing a circle
Result: "true" if S^- and S^+ are circularly separable, "false" otherwise
Output: p_1, p_2, p_3, three points caracterizing a separating circle if "true"
`// initialisation step`
Construct the set of $S = S^- \cup S^+$;
Randomly permute the points of S ;
`// points of` S `are numbered from 1 to` $|S|$`,` $|S|$ `is the size of the set`
Initialize p_1, p_2, p_3 with three points of S ; `// we assume here that` $|S| > 3$
`// recursive step`
return areCircularlySeparable(S^-, S^+, S, $|S|$, p_1, p_2, p_3, 3) ;

Question 4 *Implement algorithms 1 and 2. Provide test files.*

3 Experiments

Question 5 *In order to check whether two connected digital sets Z and $\bar{Z}$ are circularly separable, it is enough to consider only the digital boundaries of Z and $\bar{Z}$. Implement a function that takes as input the common contour of Z and $\bar{Z}$ and that checks whether Z and $\bar{Z}$ are circularly separable or not. You may use* DGtal *and more precisely the class* `GridCurve` *that can return the set of boundary digital points (see e.g.* `IncidentPointsRange`*).*

Question 6 *Perform a running time analysis of your recognition function.*

- *Implement a function that constructs the contour of a disk of a given radius.*
- *Output the running time of your recognition function for disks of increasing radius.*
- *Plot the graph of the running times against the size of the contour.*
- *Do you observe the expected linear-time complexity ?*

Algorithm 2: areCircularlySeparable(S^-, S^+, S, n, p_1, p_2, p_3, k)

Input: $S^-, S^+ \subset \mathbb{Z}^2$, the inner and outer point sets, $S = S^- \cup S^+$
n, number of points of S to process ($1 \leq n \leq |S|$)
$p_1, p_2, p_3 \in \mathbb{Z}^2$, three points characterizing a circle
k, number of variable points among $\{p_1, p_2, p_3\}$ ($0 \leq k \leq 3$)
Result: "true" if S^- and S^+ are circularly separable, "false" otherwise
Output: p_1, p_2, p_3, three points caracterizing a separating circle if "true"

```
areSeparable ← TRUE ;
if k > 0 then
    for l from 1 to k, initialize p_l with a point of S ;
    i ← 1 ;
    while areSeparable and i < n do
        if (s_i ∈ S^- and s_i is strictly OUTSIDE the circle passing by p_1, p_2, p_3)
        or (s_i ∈ S^+ and s_i is strictly INSIDE the circle passing by p_1, p_2, p_3) then
            p_k ← s_i ;
            areSeparable ← areCircularlySeparable(S^-, S^+, i, p_1, p_2, p_3, k − 1) ;
        i ← i + 1 ;
else
    i ← 1 ;
    while areSeparable and i < n do
        if (s_i ∈ S^- and s_i is strictly OUTSIDE the circle passing by p_1, p_2, p_3)
        or (s_i ∈ S^+ and s_i is strictly INSIDE the circle passing by p_1, p_2, p_3) then
            areSeparable ← FALSE ;
        i ← i + 1 ;
return areSeparable ;
```

4 Extra works

Question 7 *Modify your recognition procedure in order to have an on-line algorithm, which takes input points two by two (one belonging to the boundary of Z and one belonging to the boundary of $\bar{Z}$) and updates the current separating circle on the fly. What is the time complexity of your procedure ?*

Question 8 *Use your on-line procedure to partition a contour into digital circular arcs and to compute the whole set of maximal digital circular arcs.*

References

[dBCvKO00] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: algorithms and applications.* Springer, 2000.

[Sei91] Raimund Seidel. Small-dimensional linear programming and convex hulls made easy. *Discrete & Computational Geometry*, 6(1):423–434, 1991.