Project: “Recognition of digital circle”

Introduction

The objective of this project is to implement an 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 explicitely 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.*

Let us now consider algorithm 1 (which uses routine 2). It is a randomized and recursive algorithm that checks whether two point sets, denoted by S^- and S^+, are separable by a circle in expected linear-time. This algorithm is based on the following idea: for each point $s_i \in S^- \cup S^+$, if it belongs to S^- (resp. S^+), but is located ouside (resp. inside) the current separating circle, then either it is on another separating circle or the two input sets are not circularly separable at all. In the aim of deciding between these two alternatives, the set of possibly separating circles is restricted to circles passing by s_i and the same algorithm is recursively called. A each call, the set of possibly 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^- and S^+ or not.

Question 4 *When its last parameter is 0, what does algorithm 2 ? Compare to the previous question. When its last parameter is $k \in \{1,2,3\}$, what does algorithm 2 ?*

Question 5 *Implement algorithms 1 and 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
1 Construct the set of $S = S^- \cup S^+$ ;
2 Randomly permute the points of $S$ ;
  // points of $S$ are numbered from 0 to $|S| - 1$, $|S|$ is the size of the set
3 Initialize $p_1, p_2, p_3$ with three points of $S$ ;          // we assume here that $|S| > 3$
  // recursive step
4 return areCircularlySeparable($S^-$, $S^+$, $S$, $p_1$, $p_2$, $p_3$, 3) ;
```

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

```
   Input: $S^-, S^+ \subset \mathbb{Z}^2$, the inner and outer point sets, $S = S^- \cup S^+$
   $p_1, p_2, p_3 \in \mathbb{Z}^2$, three points characterizing a circle
   $n$, degree of freedom: only the first $p_k$ for $k$ from 1 to $n$ are variable
   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”
 1 if $n \geq 0$ then
 2     Initialize $p_k$ for $k$ from 1 to $n$ with $n$ points of $S$ ;
 3     $i \leftarrow 0$ ;
 4     areSeparable $\leftarrow$ TRUE ;
 5     while areSeparable and $i < |S|$ do
 6         if ($s_i \in S^-$ and $s_i$ is outside the circle passing by $p_1, p_2, p_3$)
 7         or ($s_i \in S^+$ and $s_i$ is inside the circle passing by $p_1, p_2, p_3$) then
 8             $p_n \leftarrow s_i$ ;
 9             areSeparable $\leftarrow$ areCircularlySeparable($S^-$, $S^+$, $S$, $p_1$, $p_2$, $p_3$, $n + 1$) ;
10         return areSeparable ;
11 else
12     return FALSE ;
```

2 Digital circle

Let us consider now the contour C of a digital set Z. If Z is connected and does not have any hole, its contour consists of a circular sequence of 1-cells. Each 1-cell of the contour is the common edge of a 2-cell whose center is a point of Z and a 2-cell whose center is a point of $\bar{Z}$. Let us denote by C^- (resp. C^+) the set of digital points of Z (resp. $\bar{Z}$) that are the center of a 2-cell incident to a 1-cell of C.

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

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

Question 6 *Implement a function that checks whether C is a digital circle or not.*

Question 7 *Perform a running time analysis of your recognition function.*

- *Implement a function that constructs circles 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 ?*

3 Extra works

Question 8 *Modify your recognition procedure in order to have an on-line algorithm, which takes input points two by two (one belonging to C^- and one belonging to C^+) and updates the current separating circle on the fly. What is the time complexity of your algorithm ?*

Question 9 *Use this algorithm to segment a contour into digital circular arcs or to compute the whole set of maximal digital circular arcs.*