Comp Sci 418 HW2

Philip King

1-30-24

Question 2.5

a. $Twin(Twin(\rightarrow e)) = \rightarrow e$
. true because the twins are in pairs. The twin of the twin is simply a loop.

b. $Next(Prev(\rightarrow e)) = \rightarrow e$
. true as even if there is an vertex with 3 or more edges as its destination, the $Next(Prev())$ will stay on the same incident face

c. $Twin(Prev(Twin(\rightarrow e))) = Next(\rightarrow e)$
. not true all the time, as there is a possibility of switching the incident face when swapping to the twin of an edge. In the case of vertex away from a main shape, (a triangle with an extra vertex), this is a shape that will show this not working by picking the inside edge.

d. $IncidentFace(\rightarrow e) = IncidentFace(Next(\rightarrow e))$
. true as $Next(\rightarrow e)$ will always preseve the incident face.

Question 2.9

Summary: This algorithm will take the vertex with the highest Y value
·This vertex will be on the outside bounds of the shape
·Then it will go around to every edge on the outside using $Next(v)$ as next will preserve the incident face to the left
·It will then add its face to a list (if not already in the list) then check the twin edge and add its incident face as well
·The algorithm will loop all the way through the outside bounds until returning back to the starting vertex.

Algorithm 1 All Outside Faces

Let $F \leftarrow$ an empty list of faces, used to store the result
let startVertex $\leftarrow$ vertex in graph with highest Y value
let currentEdge $\leftarrow Next(IncidentEdge(startVertex))$
//so that we can start the while loop
add $IncidentFace(IncidentEdge(startVertex))$
//add face next ot the startVertex just in case.

while $currentEdge \neq incidentEdge(startVertex)$ **do**
  if $incidentFace(currentEdge) \notin F$ **then**
    add $incidentFace(currentEdge) \rightarrow F$
  end if
  if $incidentFace(Twin(currentEdge)) \notin F$ **then**
    add $incidentFace(Twin(currentEdge)) \rightarrow F$
  end if
end while

return F //this should be every face around the outside of the DCEL

Question 2.12

sort by x, sweep for triangle blah blah

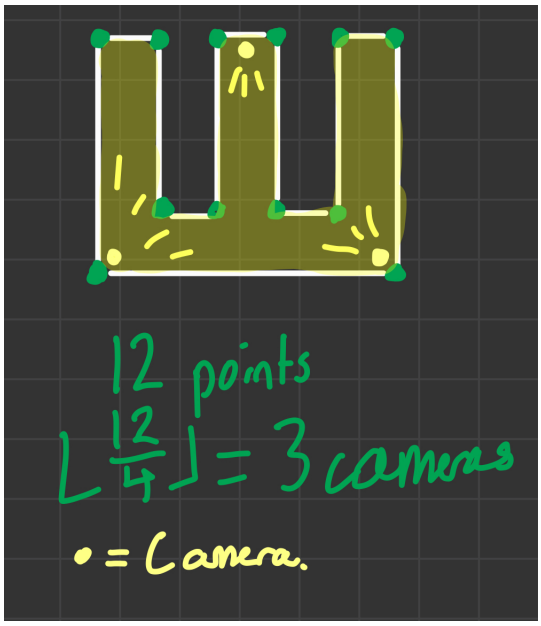
Question 2.14

This is a question of occlusion which many game engines have built it.

Method: To do this we will use a balanced binary search tree to store the line segments.
First we will get the point P and sort all over points by their polar angle and keep a record of this polar coordinate.
mext

Question 3.2

The most simple answer would be just a rectangle, below is a more involved answer.



If you were to add another "spike" to the polygon below it would add 4 more vertex and 1 more camera.