Computational Geometry

Marco Pellegrini

Lecture 3: Turn test and segment intersection test

1

Summary of lecture 3

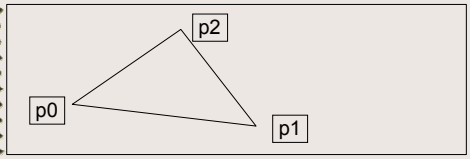
- Review of Lecture 2
  Area of a triangle,
  Area of a simple polygon,
  Robustness.
- Summary of Lecture 3.
  Left-Turn/Right turn test,
  Segment Intersection test.

2

Visiting a triangle in two ways.

(clockwise and anticlockwise)

- Take a triangle T=(p0,p1,p2) in anticlockwise way.
- We have 2Area(T) = f(p0,p1) + f(p1,p2) + f(p2,p0).
- If we go clockwise we have
  f(p0,p2) + f(p2,p1) + f(p1,p0) = -2Area(T).



3

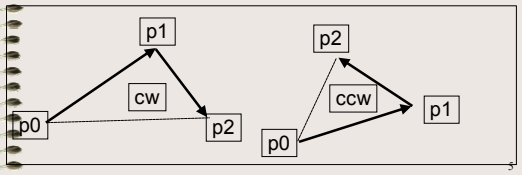
Sign of the area

- The formula f(p0,p1)+f(p1,p2)+f(p2,p0) gives us a number :K.
- The absolute value of K: |K| is twice the area.
- If K>0 then p0,p1,p2 are in anti-clockwise order.
- If K<0 then p0,p1,p2 are in clockwise order.
- If K=0 then the 3 points p0, p2, p2 are collinear.

4

Turns and sign of the area.

- If p0,p1,p2 make a left-turn then the area of T=(p0,p1,p2) is positive.
- If p0,p1,p2 make a right turn then the area of T=(p0,p1,p2) is negative.

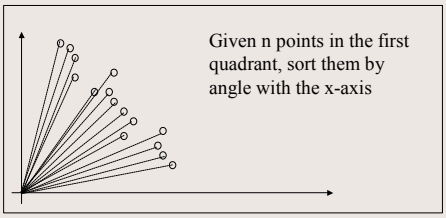


5

Applications of left-turn/right-turn test.

We saw the convexity test for a simple polygon in lecture 1.

Another application is to sort by angle without computing any angle.



Given n points in the first quadrant, sort them by angle with the x-axis

6

Sorting by angle.

- We can use any general sorting algorithm (e.g. merge-sort, heap-sort, quicksort).
- When we compare two points pi and pj we have just to check. If (pi,O,pj) is a left turn then the angle of pi is greater than the angle of pj.
- We can sort by angle around any point p, not only the origin. In this case we test (pi,p,pj) for left-handedness.

7

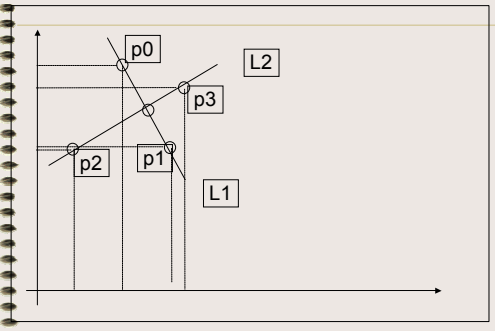
Intersection of Segments

(high school method)

- We are given segments (p0,p1) and (p2,p3) on the plane and we have to determine whether they intersect or not.
- The high-school method would be:
  (1) determine the line through p0 and p1,
  (2) determine the line through p2 and p3,
  (3) find the intersection point of the lines,
  (4) check that the intersection is in the segments.

8

Intersection of segments

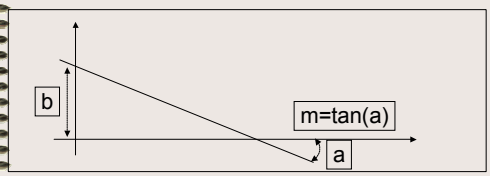


9

Equation of the lines.

(slope/intercept form)

- The slope/intercept equation of the line is $y = mx + b$, for non-vertical lines.
- m is the slope (i.e. the tangent of the angle of the line with the positive x-axis.
  b is the intercept on the y-axis.



10

Equation of a Line through 2 points

- The line through 2 points p0=(x0,y0) and p1=(x1,y1) is:
- (y-y1)/(y0-y1) = (x-x1)/(x0-x1).
- If we put such line in slope/intercept form we get:
  m = (y0-y1)/(x0-x1),
  b = (1-m)x1
- So, m is defined only if $x0 \neq x1$. The case x0=x1 must be treated separately.
- Using the above formula we find the equations of lines L1 through (p0,p1) and L2 through (p2,p3).

11

Finding the intersection point

- To find the intersection point we solve the following system of equations:
  (L1) y = m1x +b1
  (L2) y = m2x +b2
- The solution is:
  yi=(b1m2 -b2m1)/(m2-m1)
  xi=(b1-b2)/(m2-m1)
- Finally we have to check that:
  min(x0,x1) <= xi <= max(x0,x1) AND
  min(x2,x3) <= xi <= max(x2,x3).
- I do not check anything on yi. Why?
- The case m1=m2 needs a special treatment.

12

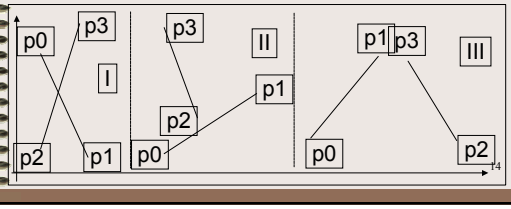
Detecting is easier than Finding.

- We have found the intersection (xi,yi) of the two segments (p0,p1), and (p2,p3).
- We have to handle several special cases and we have used division in several places.
- Division generates truncation errors that can produce the wrong final result.
- But, we do not have to FIND the intersection point in order to DETECT its existence.
- Moreover we can DETECT the existence of an intersection using only left-turn tests.

13

Proper Intersections.

- First we deal with the case of proper intersections. We suppose that no 3 points among p0,p1,p2,p3 are collinear.
- Exercise: test left turn for: (p0,p1,p2), (p0,p1,p3), (p2,p3,p0), (p2,p3,p1) in 3 cases:



14

Observations

- The 3 points to test have been chosen in the following way: take two end-points of one segment and one of end-point of the other segment. The two points from the same segment are chosen in both cases in the same order.
- In cases II and III at least one pair of turns based on the same segment are equal.
- In case I the two turns based on each segment are different.
- Hypothesis: (p0,p1) meet (p2,p3) if and only if: (left-turn(p0,p1,p2) xor left-turn(p0,p1,p3) AND (left-turn(p2,p3,p0) xor left-turn(p2,p3,p1).

15

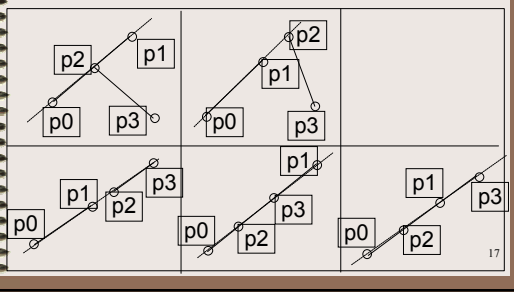
A formal proof.

- Consider the segments (p0,p1) and (p2,p3) and the lines L1 and L2.
- If (p0,p1) meets (p2,p3) then also L1 meets (p2,p3) and L2 meets (p0,p1).
- But the converse is also true: if L1 meets (p2,p3) and L2 meets (p0,p1), then they meet in the same point, and this point belongs to the two segments.
- The segment (p0,p1) meets L2 if p0 and p1 are on opposite sides of L2, which is the same as to require that (p2,p3,p0) and (p2,p3,p1) make different turns (one left and one right).
- A similar argument holds for L1 and (p2,p3).

16

Improper intersections.

- Here we have that 3 points among p0,p1,p2,p3 are collinear, so at least one of the tests cannot return a true/false value.



Improper intersections, 2.

- We define a procedure Between that given 3 collinear points (a,b,c) determines wether c is between a and b.
- BETWEEN(a,b,c: points)
  IF NOT COLLINEAR(a,b,c)
  THEN RETURN false
  IF x(a)≠ x(b)
  THEN RETURN (min(x(a),x(b)) ≤ x(c) AND
  max(x(a),x(b)) ≥ x(c))
  ELSE RETURN (min(y(a),y(b)) ≤ y(c) AND
  max(y(a),y(b)) ≥ y(c))
  END IF

Segment intersection Test

- Intersect((a,b),(c,d): Segments)
  IF [Collinear(a,b,c) OR
  Collinear(a,b,d) OR
  Collinear(c,d,a) OR
  Collinear(c,d,b)]
  THEN
  RETURN [Between(a,b,c) OR
  Between(a,b,d) OR
  Between(c,d,a) OR
  Between(c,d,b)]
  ELSE
  RETURN [[left(a,b,c) xor left(a,b,d)] AND
  [left(c,d,a) xor left(c,d,b)]]
  END IF

Exercise

- Implement in C the procedure to test intersection of segments.
- Test the procedure with segments that represent all the cases shown in class.

Conclusions

- The formula to compute the signed area of a triangle is used to determine left-turns/right turns.
- left-turn tests are used to test convexity, to sort by angle.
- Finding an intersection requires divisions, BUT detecting an intersection requires only left-turn and collinearity tests.
- We have now the whole method to test simplicity of polygons.
- The INTERSECT procedure is robust against truncation errors.
- More uses of these tests are to come.

21