< Back | Python O(n), by checking the directions of every edge, learned from Robert Sedgewick's Algorithm course

xuhongnever ★ 21 Last Edit: October 15, 2018 3:42 PM · 1.4K VIEWS

17

The idea is to check whether the edges are always make turns in the same directions (clockwise or counterclockwise)
The method to check direction is from Robert Sedgewick's Algorithm course.

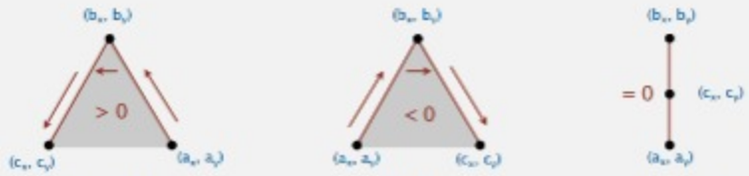
Implementing ccw

CCW. Given three points a, b, and c, is $a \rightarrow b \rightarrow c$ a counterclockwise turn?

- Determinant (or cross product) gives 2x signed area of planar triangle.

$$2 \times Area(a, b, c) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = \underbrace{(b_x - a_x)(c_y - a_y) - (b_y - a_y)(c_x - a_x)}_{(b - a) \times (c - a)}$$

- If signed area > 0, then $a \rightarrow b \rightarrow c$ is counterclockwise.
- If signed area < 0, then $a \rightarrow b \rightarrow c$ is clockwise.
- If signed area $= 0$, then $a \rightarrow b \rightarrow c$ are collinear.



counterclockwise clockwise collinear

76

```
class Solution(object):
    def isConvex(self, points):
        """
        :type points: List[List[int]]
        :rtype: bool
        """
        def direction(a,b,c):
            return (b[0]-a[0])*(c[1]-a[1]) - (b[1]-a[1])*(c[0]-a[0])
        d = None
        for i in range(1,len(points)):
            a = direction(points[i-2],points[i-1],points[i])
            if a == 0: continue
            if d == None: d = a
            else:
                if a*d < 0: return False
        if direction(points[-2],points[-1],points[0]) * d < 0:return False
        return True
```

Comments: 3 Best | Most Votes | Newest to Oldest | Oldest to Newest

Type comment here... (Markdown is supported)

Post

yuanzz ★ 30 July 23, 2018 7:04 AM

```
for i in range(1,len(points)):
```

I think you should start the iteration from index 0 instead of 1
then you dont need to add

```
if direction(points[-2],points[-1],points[0]) * d < 0:return False
```

1 Reply

CRunner ★ 89 September 18, 2018 6:43 PM

Here's a Java implementation:

```
public boolean isConvex(List<List<Integer>> points) {
    int n = points.size();
    Integer dir = null;
    for (int i = 2; i < n+3; i++) {
        int f = (i-2)%n;
        int s = (i-1)%n;
        int t = i%n;
        int d = getD(points.get(f), points.get(s), points.get(t));
        if (d == 0)
```

Read More

0 Reply

abilityfun ★ 32 January 21, 2017 7:05 AM

I improved on your solution by making it more pythonic and wrapping around completely.

```
class Solution(object):
    def isConvex(self, points):
        """
        :type points: List[List[int]]
        :rtype: bool
        """
        def direction(a,b,c):
            return (b[0]-a[0])*(c[1]-a[1]) - (b[1]-a[1])*(c[0]-a[0])
```

Read More

0 Reply