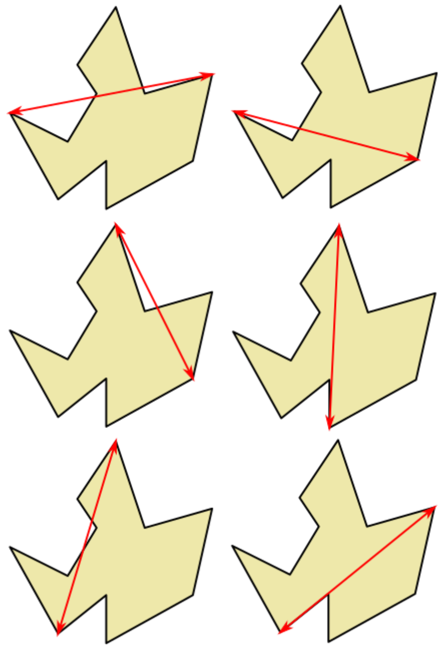
WIKIPEDIA

Rotating calipers

In computational geometry, the method of **rotating calipers** is an algorithm design technique that can be used to solve optimization problems including finding the width or diameter of a set of points.

The method is so named because the idea is analogous to rotating a spring-loaded vernier caliper around the outside of a convex polygon.[1] Every time one blade of the caliper lies flat against an edge of the polygon, it forms an antipodal pair with the point or edge touching the opposite blade. The complete "rotation" of the caliper around the polygon detects all antipodal pairs; the set of all pairs, viewed as a graph, forms a thrackle. The method of rotating calipers can be interpreted as the projective dual of a sweep line algorithm in which the sweep is across slopes of lines rather than across x- or y-coordinates of points.



Sequence of probes around the convex hull of a polygon to determine its diameter using Rotating Caliper method.

Contents

History

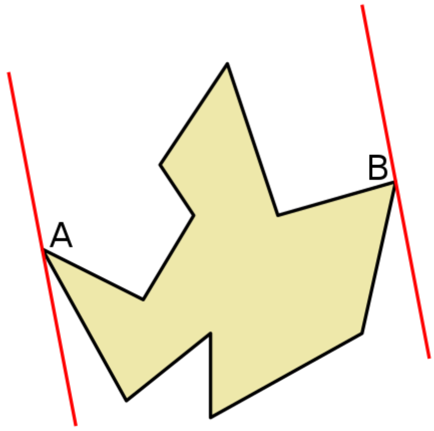
The rotating calipers method was first used in the dissertation of Michael Shamos in 1978.[2] Shamos uses this method to generate all antipodal pairs of points on a convex polygon and to compute the diameter of a convex polygon in $O(n)$ time. Godfried Toussaint coined the phrase "rotating calipers" and also demonstrated that the method was applicable in solving many other computational geometry problems.[3]

Shamos's algorithm

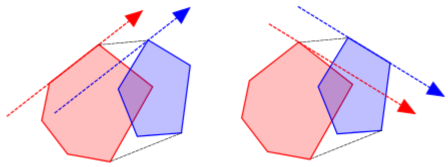
Shamos gave following algorithm in his dissertation (pp 77–82) for the rotating calipers method that generated all antipodal pairs of vertices on convex polygon:[2]

```
/* p[] is in standard form, ie, counter clockwise order,
     distinct vertices, no collinear vertices.
   ANGLE(m,n) is a procedure that returns the clockwise angle
```

```
     swept out by a ray as it rotates from a position parallel
     to the directed segment Pm,Pm+1 to a position parallel to
Pn,Pn+1
   We assume all indices are reduced to mod N (so that N+1 = 1).
*/
GetAllAntiPodalPairs(p[1..n])
  //Find first anti-podal pair by locating vertex opposite P1
  i = 1
  j = 2
  while angle(i, j) < pi
    j++
  yield i,j

  /* Now proceed around the polygon taking account of
     possibly parallel edges. Line L passes through
     Pi, Pi+1 and M passes through Pj, Pj+1
  */

  //Loop on j until all of P has been scanned
  current = i
  while j <> n
    if angle(current, i+1) <= angle(current, j+1)
      j++
      current = j
    else
      i++
      current = i
    yield i,j

    //Now take care of parallel edges
    if angle(current, i+1) = angle(current, j+1)
      yield i+1, j
      yield i, j+1
      yield i+1, j+1
      if current = i
        j++
      else
        i++
```



An *antipodal pair of vertex* and their *supporting parallel lines*.



Rotating calipers, finding a bridge between two convex polygons

Another version of this algorithm appeared in the text by Preparata and Shamos in 1985 that avoided calculation of angles:[4]

```
GetAllAntiPodalPairs(p[1..n])
  i0 = n
  i = 1
  j = i+1
  while (Area(i,i+1,j+1) > Area(i,i+1,j))
    j = j+1
    j0 = j
    while (j <> i0)
      i = i+1
      yield i,j
      while (Area(i,i+1,j+1) > Area(i,i+1,j)
        j=j+1
        if ((i,j) <> (j0,i0))
              yield i,j
        else
              return
      if (Area(j,i+1,j+1) = Area(i,i+1,j))
        if ((i,j) <> (j0,i0))
          yield i,j+1
        else
          yield i+1,j
```

Using monotone chain algorithm

This method has several advantages including that it avoids calculation of area or angles as well as sorting by polar angles. The method is based on finding convex hull using Monotone chain method (https://en.wikibooks.org/wiki/Algorithm_Implementation/Geometry/Convex_hull/Monotone_chain) devised by A.M. Andrew[5] which returns upper and lower portions of hull separately that then can be used naturally for rotating calipers analogy.[6]

```
/* All indices starts from 1.
   dir(p,q,r) returns +ve number if p-q-r segments are clockwise,
```

```
    -ve number if they are anti clockwise and 0 if collinear.
    it can be defined as (q.y-p.y)(r.x-p.x) - (q.x-p.x)(r.y-p.y)
*/
GetAllAntiPodalPairs(p[1..n])
  //Obtain upper and lower parts of polygon
  p’ = Sort p lexicographically (i.e. first by x then by y)
  U, L = create new stacks of points
  for k = 1 to n
    while U.size > 1 and dir(U[k-1], U[k], p’[k]) <= 0
      U.pop()
    while L.size > 1 and dir(L[k-1], L[k], p’[k]) >= 0
      L.pop()
    U.append(p’[k])
    L.append(p’[k])

  //Now we have U and L, apply rotating calipers
  i = 1
  j = L.size
  while i < U.size or j > 1
    yield U[i], L[j]

    //if i or j made it all the way through
    //advance other size
    if i = U.size
      j = j - 1
    else if j = 1
      i = i + 1
    else if (U[i+1].y - U[i].y) * (L[j].x - L[j-1].x)
         > (U[i+1].x - U[i].x) * (L[j].y - L[j-1].y)
      i = i + 1
    else
      j = j - 1
```

Applications

Toussaint[7] and Pirzadeh[8] describes various applications of rotating calipers method.

Distances

- Diameter (maximum width) of a convex polygon[9][10]
- Width (minimum width) of a convex polygon[11]
- Maximum distance between two convex polygons[12][13]
- Minimum distance between two convex polygons[14][15]
- Widest empty (or separating) strip between two convex polygons (a simplified low-dimensional variant of a problem arising in support vector machine based machine learning)
- Grenander distance between two convex polygons[16]
- Optimal strip separation (used in medical imaging and solid modeling)[17]

Bounding boxes

- Minimum area oriented bounding box
- Minimum perimeter oriented bounding box

Triangulations

- Onion triangulations
- Spiral triangulations
- Quadrangulation
- Nice triangulation
- Art gallery problem
- Wedge placement optimization problem[18]

Multi-Polygon operations

- Union of two convex polygons
- Common tangents to two convex polygons
- Intersection of two convex polygons[19]
- Critical support lines of two convex polygons
- Vector sums (or Minkowski sum) of two convex polygons[20]
- Convex hull of two convex polygons

Traversals

- Shortest transversals[21][22]
- Thinnest-strip transversals[23]

Others

- Non parametric decision rules for machine learned classification[24]
- Aperture angle optimizations for visibility problems in computer vision[25]
- Finding longest cells in millions of biological cells[26]
- Comparing precision of two people at firing range
- Classify sections of brain from scan images

Minimum width of a convex polygon

```
ARRAY points := {P1, P2, ..., PN};

points.delete(middle vertices of any collinear sequence of three points);

REAL p_a := index of vertex with minimum y-coordinate;
REAL p_b := index of vertex with maximum y-coordinate;

REAL rotated_angle := 0;
REAL min_width := INFINITY;

VECTOR caliper_a(1,0);    // Caliper A points along the positive x-axis
VECTOR caliper_b(-1,0);   // Caliper B points along the negative x-axis

WHILE rotated_angle < PI

  // Determine the angle between each caliper and the next adjacent edge in the polygon
  VECTOR edge_a(points[p_a + 1].x - points[p_a].x, points[p_a + 1].y - points[p_a].y);
  VECTOR edge_b(points[p_b + 1].x - points[p_b].x, points[p_b + 1].y - points[p_b].y);
  REAL angle_a := angle(edge_a, caliper_a);
  REAL angle_b := angle(edge_b, caliper_b);
  REAL width := 0;

  // Rotate the calipers by the smaller of these angles
  caliper_a.rotate(min(angle_a, angle_b));
  caliper_b.rotate(min(angle_a, angle_b));

  IF angle_a < angle_b
    p_a++;  // This index should wrap around to the beginning of the array once it hits the end
    width = caliper_a.distance(points[p_b]);
  ELSE
    p_b++;  // This index should wrap around to the beginning of the array once it hits the end
    width = caliper_b.distance(points[p_a]);
  END IF

  rotated_angle = rotated_angle + min(angle_a, angle_b);

  IF (width < min_width)
    min_width = width;

  END IF
END WHILE
```

```
RETURN min_width;
```

See also

- Convex polygon
- Convex hull
- Smallest enclosing box
- es:Rotating calipers

References

1. "Rotating Calipers" (http://www-cgrl.cs.mcgill.ca/~godfried/research/calipers.html) at Toussaint's home page
2. Shamos, Michael (1978). "Computational Geometry" (http://euro.ecom.cmu.edu/people/faculty/mshamos/1978ShamosThesis.pdf) (PDF). Yale University. pp. 76–81.
3. Toussaint, Godfried T. (1983). "Solving geometric problems with the rotating calipers" (http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.155.5671). Proc. MELECON '83, Athens.
4. Shamos, Franco P. Preparata, Michael Ian (1985). *Computational Geometry An Introduction*. New York, NY: Springer New York. ISBN 978-1-4612-7010-2.
5. Andrew, A. M. (1979). "Another efficient algorithm for convex hulls in two dimensions". *Information Processing Letters*. **9** (5): 216–219. doi:10.1016/0020-0190(79)90072-3 (https://doi.org/10.1016%2F0020-0190%2879%2990072-3).
6. Eppstein, David. "Convex hull and diameter of 2d point sets (Python recipe)" (http://code.activestate.com/recipes/117225/).
7. Toussaint, Godfried (2014). "The Rotating Calipers: An Efficient, Multipurpose, Computational Tool" (http://sdiwc.net/digital-library/web-admin/upload-pdf/00001036.pdf) (PDF). *Proceedings of the International conference on Computing Technology and Information Management*: 215–225.
8. Pirzadeh, Hormoz. "Computational geometry with the rotating calipers" (http://digitool.Library.McGill.CA:80/R/-?func=dbin-jump-full&object_id=21623&silo_library=GEN01). *McGill Library*.
9. Binay K. Bhattacharya and Godfried T. Toussaint, "Fast algorithms for computing the diameter of a finite planar set," *The Visual Computer*, Vol. 3, No. 6, May 1988, pp.379–388.
10. Binay K. Bhattacharya and Godfried T. Toussaint, "A counter example to a diameter algorithm for convex polygons," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. PAMI-4, No. 3, May 1982, pp. 306–309.
11. Michael E. Houle and Godfried T. Toussaint, "Computing the width of a set," *IEEE Transactions Pattern Analysis & Machine Intelligence*, Vol. 10, no. 5, September 1988, pp. 761–765.
12. Godfried T. Toussaint and Jim A. McAlear, "A simple O(*n* log *n*) algorithm for finding the maximum distance between two finite planar sets," *Pattern Recognition Letters*, Vol. 1, 1982, pp. 21–24.
13. Binay K. Bhattacharya and Godfried T. Toussaint, "Efficient algorithms for computing the maximum distance between two finite planar sets," *Journal of Algorithms*, vol. 14, 1983, pp. 121–136.
14. Godfried T. Toussaint and Binay K. Bhattacharya, "Optimal algorithms for computing the minimum distance between two finite planar sets," *Pattern Recognition Letters*, vol. 2, December, 1983, pp. 79–82.
15. "Rotating Calipers" (https://web.archive.org/web/20150330010154/http://cgm.cs.mcgill.ca/~orm/rotcal.html). 2015-03-30. Archived from the original on 2015-03-30. Retrieved 2017-03-22.
16. MARTINEZ, HUGO M. (January 1, 1978). "Review of: "PATTERN SYNTHESIS", by U. Grenander, Springer-Verlag, New York, 1976. 509 pp" (https://dx.doi.org/10.1080/03081077808960672). *International Journal of General Systems*. **4** (2): 126–127. doi:10.1080/03081077808960672 (https://doi.org/10.1080%2F03081077808960672). ISSN 0308-1079 (https://www.worldcat.org/issn/0308-1079).
17. Barequet and Wolfers (1998). "Optimizing a Strip Separating Two Polygons". *Graphical Models and Image Processing*. doi:10.1006/gmip.1998.0470 (https://doi.org/10.1006%2Fgmip.1998.0470).
18. Teichmann, Marek (1989). "Wedge placement optimization problems" (http://digitool.Library.McGill.CA:80/R/-?func=dbin-jump-full&object_id=55636&silo_library=GEN01).

19. Godfried T. Toussaint, "A simple linear algorithm for intersecting convex polygons, *The Visual Computer*, Vol. 1, 1985, pp. 118–123.
20. Tomas Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, Vol. 32, No. 2, 1983, pp. 108–120.
21. Binay K. Bhattacharya and Godfried T. Toussaint, "Computing shortest transversals," *Computing*, vol. 46, 1991, pp. 93–119.
22. Binay K. Bhattacharya, Jurek Czyzowicz, Peter Egyed, Ivan Stojmenovic, Godfried T. Toussaint, and Jorje Urrutia, "Computing shortest transversals of sets," *International Journal of Computational Geometry and Applications*, Vol. 2, No. 4, December 1992, pp. 417–436.
23. Jean-Marc Robert and Godfried T. Toussaint, "Linear approximation of simple objects," *Computational Geometry: Theory and Applications*, Vol. 4, 1994, pp. 27–52.
24. Rasson and Granville (1996). "Geometrical tools in classification". *Computational Statistics & Data Analysis*. **23** (1): 105–123. doi:10.1016/S0167-9473(96)00024-2 (https://doi.org/10.1016%2FS0167-9473%2896%2900024-2).
25. "Some Aperture-Angle Optimization Problems" (https://link.springer.com/article/10.1007/s00453-001-0112-9). *Algorithmica*. **33** (4): 411–435. 2002-08-01. doi:10.1007/s00453-001-0112-9 (https://doi.org/10.1007%2Fs00453-001-0112-9). ISSN 0178-4617 (https://www.worldcat.org/issn/0178-4617).
26. "Incorrect Diameter Algorithms for Convex Polygons" (http://cgm.cs.mcgill.ca/~athens/cs507/Projects/2000/MS/diameter/document.html).
