



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Featured content](#)
[Current events](#)
[Random article](#)
[Donate to Wikipedia](#)
[Wikipedia store](#)

Interaction

[Help](#)
[About Wikipedia](#)
[Community portal](#)
[Recent changes](#)
[Contact page](#)

Tools

[What links here](#)
[Related changes](#)
[Upload file](#)
[Special pages](#)
[Permanent link](#)
[Page information](#)
[Wikidata item](#)
[Cite this page](#)

Print/export

[Create a book](#)
[Download as PDF](#)
[Printable version](#)

Languages


[Español](#)

 [Edit links](#)

[Create account](#) [Log in](#)

Article [Talk](#)

[Read](#) [Edit](#) [View history](#)



Rotating calipers

From Wikipedia, the free encyclopedia

In [computational geometry](#), **rotating calipers** is the method that has been found useful in solving number of problems.

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.

Contents [\[hide\]](#)

- History
- Shamos's algorithm
- Using monotone chain algorithm
- Applications
 - Distances
 - Bounding boxes
 - Triangulations
 - Multi-Polygon operations
 - Traversals
 - Others
- Minimum width of a convex polygon
- References
- See also

History [\[edit\]](#)

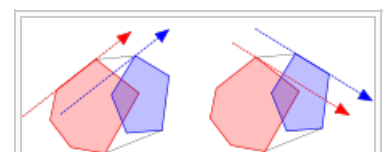
Rotating calipers method was first used in the dissertation of the [Michael Shamos](#) in 1978.^[2] The algorithm in that dissertation uses this method for generating all [antipodal](#) pairs of points on a [convex polygon](#) for computing the diameter of a convex polygon in $O(n)$ time. [Godfried Toussaint](#) coined the phrase "rotating calipers" and also demonstrated that the method was applicable in solving many computational geometry problems involving [wide range of areas](#).^[3]

Shamos's algorithm [\[edit\]](#)

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

```
/* p[] is in standard form, ie, counter clockwise
order,
distinct vertices, no collinear vertices.
ANGLE(M,nJ 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
```



Rotating calipers, finding a bridge between two convex polygons 

```

*/

//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++

```

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,i))
        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 [\[edit\]](#)

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](#) [↗](#) devised by A.M. Andrew^[5] which returns upper and lower portions of hull separately that then can be used naturally for rotating callipers 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 callipers
    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 [\[edit\]](#)

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

Distances [\[edit\]](#)

- 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]
- 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^[15]
- Optimal strip separation (used in medical imaging and solid modeling)^[16]

Bounding boxes [\[edit\]](#)

- Minimum area [oriented bounding box](#)
- Minimum perimeter [oriented bounding box](#)

Triangulations [\[edit\]](#)

- Onion [triangulations](#)
- Spiral [triangulations](#)
- [Quadrangulation](#)
- Nice triangulation
- Art gallery problem
- Wedge placement optimization problem^[17]

Multi-Polygon operations [\[edit\]](#)

- Union of two convex polygons
- Common tangents to two convex polygons
- Intersection of two convex polygons^[18]
- [Critical support lines](#) of two convex polygons
- Vector sums (or Minkowski sum) of two convex polygons^[19]
- Convex hull of two convex polygons

Traversals [\[edit\]](#)

- Shortest transversals^{[20][21]}
- Thinnest-strip transversals^[22]

Others [\[edit\]](#)

- Non parametric decision rules for machine learned classification^[23]

- Aperture angle optimizations for visibility problems in computer vision^[24]
- Finding longest cells in millions of biological cells^[25]
- Comparing precision of two people at firing range
- Classify sections of brain from scan images

Minimum width of a convex polygon [\[edit\]](#)

```

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;

```

References [\[edit\]](#)

1. [^] "Rotating Calipers" [↗](#) at Toussaint's home page
2. [^] ^a ^b Shamos, Michael (1978). "Computational Geometry"  (PDF). Yale University. pp. 76–81.
3. [^] Toussaint, Godfried T. (1983). "Solving geometric problems with the rotating calipers" [↗](#). 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.
6. [^] Eppstein, David. "Convex hull and diameter of 2d point sets (Python recipe)" [↗](#).
7. [^] Toussaint, Godfried (2014). "The Rotating Calipers: An Efficient, Multipurpose, Computational Tool"  (PDF).

- Proceedings of the International conference on Computing Technology and Information Management*: 215–225.
8. [▲] Pirzadeh, Hormoz. "Computational geometry with the rotating calipers" [↗]. *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. [▲] MARTINEZ, HUGO M. (January 1, 1978). "Review of: "PATTERN SYNTHESIS", by U. Grenander, Springer-Verlag, New York, 1976. 509 pp." [↗]. *International Journal of General Systems* 4 (2): 126–127. doi:10.1080/03081077808960672 [↗]. ISSN 0308-1079 [↗].
 16. [▲] Barequet and Wolfers (1998). "Optimizing a Strip Separating Two Polygons". *Graphical Models and Image Processing*. doi:10.1006/gmip.1998.0470 [↗].
 17. [▲] Teichmann, Marek (1989). "Wedge placement optimization problems" [↗].
 18. [▲] Godfried T. Toussaint, "A simple linear algorithm for intersecting convex polygons, *The Visual Computer*, Vol. 1, 1985, pp. 118–123.
 19. [▲] Tomas Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, Vol. 32, No. 2, 1983, pp. 108–120.
 20. [▲] Binay K. Bhattacharya and Godfried T. Toussaint, "Computing shortest transversals," *Computing*, vol. 46, 1991, pp. 93–119.
 21. [▲] Binay K. Bhattacharya, Jurek Czyzowicz, Peter Egedy, 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.
 22. [▲] Jean-Marc Robert and Godfried T. Toussaint, "Linear approximation of simple objects," *Computational Geometry: Theory and Applications*, Vol. 4, 1994, pp. 27–52.
 23. [▲] Rasson and Granville (1996). "Geometrical tools in classification". *Computational Statistics & Data Analysis* 23 (1): 105–123. doi:10.1016/S0167-9473(96)00024-2 [↗].
 24. [▲] "Some Aperture-Angle Optimization Problems" [↗]. *Algorithmica* 33 (4): 411–435. 2002-08-01. doi:10.1007/s00453-001-0112-9 [↗]. ISSN 0178-4617 [↗].
 25. [▲] "Incorrect Diameter Algorithms for Convex Polygons" [↗].

See also ^[edit]

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

Categories: Geometric algorithms | Convex geometry

This page was last modified on 1 July 2015, at 15:13.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Mobile view](#)

