International Journal of Digital Information and Wireless Communications (IJDIWC) 4(3): 372-386
The Society of Digital Information and Wireless Communications, 2014 (ISSN: 2225-658X)

Applications of the *Rotating Calipers* to Geometric Problems in Two and Three Dimensions

Godfried T. Toussaint
New York University Abu Dhabi
United Arab Emirates
gt42@nyu.edu

ABSTRACT

A paper published in 1983 established that the *rotating calipers* paradigm provides an elegant, simple, and yet powerful computational tool for solving several two-dimensional geometric problems. Since then the rotating calipers have been extended to three dimensions, and have been applied to many new problems. In the present paper the history of this tool is reviewed, and stock is taken of the rich variety of computational problems and applications that have been tackled with it during the past thirty years.

KEYWORDS

Rotating calipers, algorithms, computational geometry, geometric complexity, computer graphics, computer vision, robotics, combinatorial optimization, line fitting, statistics, graphs, thrackles, rotating planes

1 INTRODUCTION

The *rotating calipers* paradigm constitutes a powerful, simple, elegant, and computationally efficient tool that can solve a wide variety of geometric problems in practice. The basic idea first appeared in the 1978 Ph.D. thesis of Michael Shamos, where it was applied to the computation of the maximum distance between the elements of a convex set: the diameter of a convex polygon in the plane [1]. Later I coined the name "Rotating Calipers" for this procedure, and generalized it in a number of ways to solve several other two-dimensional geometric problems. In 1983 I presented some of these results at an IEEE conference in Athens, Greece [2]. Since then the rotating calipers paradigm has been generalized further to solve other problems in two as well as three dimensions. In the present paper the thirty-year history of this tool is reviewed.

2 THE ROTATING CALIPERS IN 2-D

The elegant and simple algorithm in Shamos' thesis [1], showing that the diameter of a convex n-sided polygon may be computed in O(n) time in the worst case, resembles rotating a pair of calipers through 360º, once around the polygon. This concept is illustrated in Fig. 1, which depicts a convex polygon, and its two horizontal lines of support (lower and upper) at vertices p_i and p_j, respectively. Note that the lines of support are *directed*, as indicated by their arrows. This permits the specification of the direction of rotation, so that if the lines are rotated in a clockwise direction while being pivoted about vertices p_i and p_j, the respective angles θ_i and θ_j that the lines make with vertices p_{i+1} and p_{j+1}, will decrease.

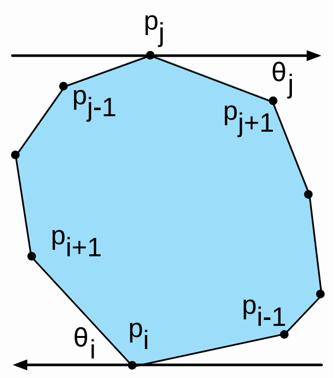

Fig. 1. The rotating calipers.

2.1 The Diameter of a Convex Polygon

The *diameter* of a polygon P is the maximum distance between a pair of points in P. Since P contains an infinite number of points, searching all of them is out of the question. To obtain an efficient algorithm we need a characterization of the diameter in terms of a finite subset of the points in P. A simple contradiction argument shows that the diameter of P is determined by a pair of vertices of P. If two points of P determine the diameter and are not vertices of P then the line segment joining these two points may be increased in length by moving either point so that it coincides with a vertex of P. Therefore the diameter may be calculated by examining only the distances between all the pairs of vertices of P, and selecting the maximum distance encountered. However, this naïve (brute force) search algorithm requires a number of operations that grows as the square of the number of vertices in the polygon. A more fruitful characterization narrows the set of candidates to be searched down to linear size. Several approaches have been tried in the past in order to speed up diameter-finding algorithms [3], and some characterizations have proved to be incorrect [4]. Several hill-climbing algorithms published are not guaranteed to yield the diameter because they assume that the distance between a vertex of a convex polygon and the remaining vertices scanned in order, is a unimodal function, when in fact such a distance function may have $\Omega(n)$ local maxima [4]. Indeed, a convex polygon may have $\Omega(n^2)$ pairs of vertices that are local maxima of their distance functions. However, a valid characterization of the diameter was obtained by Shamos [1] via the pairs of vertices, such as p_i and p_j in Fig. 1. These vertices are *antipodal*, meaning that they admit parallel lines of support. Shamos showed that the diameter of a polygon is determined by two of its antipodal vertices. Furthermore, a polygon with n vertices has $O(n)$ antipodal pairs, assuming that all the vertices of P that have an angle of 180º have been removed, which is a straightforward matter. The rotating calipers provide a simple $O(n)$ time procedure for searching all the antipodal pairs to find the maximum. The idea is to first place a pair of parallel lines of support in any orientation, say horizontal, as in Fig. 1, and then "rotate" the lines, while keeping them as support lines of the polygon, until they are horizontal again. Such a procedure will visit all pairs of antipodal vertices. The crucial observation that makes this seeming infinite continuous process discrete and finite is that the rotation process can hop from vertex to vertex. Observe in Fig. 1, that as the two lines of support rotate, the vertices p_i and p_j maintain their antipodality property until one of the lines lies flush with an edge of P. In Fig. 1, θ_j is smaller than θ_i, and thus the line advances from p_j to p_{j+1} identifying the next antipodal pair p_i and p_{j+1}. At each step all that is needed is a comparison of two angles to determine which of the two is smaller, which can be done in constant time.

2.2 The Width of a Convex Polygon

The *width* of a polygon P is the minimum distance between a pair of parallel lines of support of P. As with the diameter definition, to obtain an algorithm, the width must be characterized by a finite subset of the lines of support that can be searched efficiently. The following property yields such a characterization. Let p_i and p_j be two vertices of the convex polygon that admit parallel lines of support, and have internal angles less than 180º, as in the example of Fig. 1. If no edge of P lies on the support lines, there exists a preferred direction of rotation for the support lines such that their separation distance decreases. This implies that the width of the polygon is characterized by a vertex and an edge (a vertex-edge pair) that are antipodal, *i.e.*, such that one of the lines of support lies flush with an edge, as shown in Fig. 2. This characterization of the width of a convex polygon found application to the segmentation of plane curves in the work of Ichida and Kiyono [5] (see also [6]). The characterization has also led to several algorithms for computing the width of a polygon. A useful property in this regard is the fact that for a line that contains any edge of a convex polygon P, the perpendicular distance between the line and the vertices of P, as they are traversed in order, defines a *unimodal* function [7]. Kurozumi and Davis [8], and Imai and Iri [9], independently proposed algorithms for computing the width of a convex polygon by visiting each edge of the polygon, and for each edge searching for the vertex furthest from it (in a perpendicular sense). Since this distance function is unimodal the algorithms in [8] and [9] apply binary search to locate these vertices, for each edge of P. This approach results in algorithms with $O(n \log n)$ worst-case time complexities.

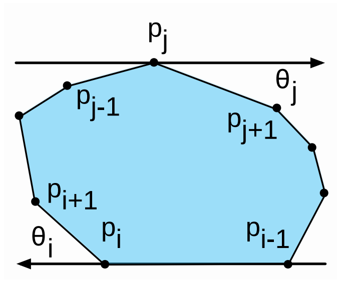

Fig. 2. The rotating calipers and the *width* of *P*.

Houle and Toussaint [10] show that an O(n) time algorithm is achievable by avoiding binary search altogether, and using the rotating calipers instead. To initialize the algorithm, a line of support is constructed through any edge, such as p_{i-1} and p_i in Fig. 2, and the vertex furthest from this line is found (p_j in Fig. 2). At each step during the rotation of the lines, the succeeding edge selected is the one that makes the smaller angle with its line of support. This process yields the vertex opposite the edge in only constant time.

2.3 The Minimum-Area Enclosing Rectangle

The algorithm described in the previous sub-section, for computing the width of a polygon with the rotating calipers, represents an application of the original calipers that use two parallel lines of support to compute the diameter, to solve a different problem. However, the rotating calipers tool itself has also been generalized in several ways. One generalization introduced in [1] uses more than two rotating lines of support. One nice example uses four lines of support to tackle a problem that arises in the areas of image processing and computer vision [11], [12], [13]-[15] optimal packing and layout problems in manufacturing [16], and automatic tariffing in goods traffic [17]. The problem is that of computing the minimum-area enclosing rectangle of a convex n-sided polygon. The usefulness of this rectangle in packing problems is obvious, but it also has applications to shape analysis. For example, the rectangularity of a shape may be measured by the difference in the areas of the shape and its smallest enclosing rectangle [12]. Freeman and Shapira [16] showed that the smallest rectangle must have one of its four edges flush (collinear) with an edge of the polygon, as illustrated in Fig. 3. The algorithm they propose involves visiting every edge of P, such as edge [p_{i-1} p_i] in Fig. 3, and locating the three associated extreme vertices that complete the enclosing rectangle, such as p_t, p_j and p_s in Fig. 3. Their algorithm inspects all the vertices of P to find these three extreme vertices, leading to a total computational complexity of O(n^2). If on the other hand each of these extreme vertices is located in O(log n) steps using binary search, instead of a linear scan (which is valid due to the unimodality property of the distance functions involved [45]), then the solution may be found in O(n log n) time. However, the rotating calipers with four lines of support solves this problem elegantly in O(n) time, as follows. To initialize the procedure any edge of the polygon is selected as the base of a candidate for the smallest enclosing rectangle, and the three extreme vertices are found by a linear scan of all the vertices, as in [16]. The rest of the algorithm proceeds in a manner similar to that used in the algorithm for computing the width of the polygon, except that here the four lines of support are rotated by the smallest of the *four* angles that the lines make with their succeeding clockwise edges, as shown in Fig. 3. In this way every edge of the polygon generates its corresponding candidate rectangle as the support lines make a full revolution around the polygon. Furthermore each candidate rectangle is generated in constant time, resulting in a total time complexity of O(n). An alternate approach to solve this problem in O(n) time, that uses a data structure known as a *star*, is described in [18] (see also [19] for the use of the star in a different context).

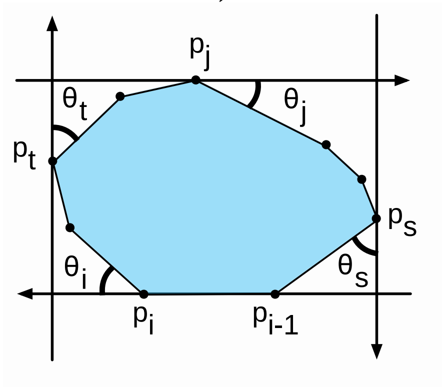

Fig. 3. The minimum-area enclosing rectangle.

In closing this sub-section it is worth noting that the rotating calipers have also be used to find minimum-area enclosing triangles [20], [21], squares [22], minimum-perimeter enclosures [23], and the densest double-lattice packing of a convex polygon [24]. The case of the minimum-area triangles enclosing a convex polygon is most similar to the rectangle problem considered in this subsection. Klee and Laskowski [20] provided the key characterization of the optimal solution. They proved that if *T* is a local minimum among the triangles that contain the polygon, then at least one side of *T* must lie flush with an edge of the polygon. This property led the authors to propose an O($n \log^2 n$) time algorithm for finding the optimal solution. O'Rourke *et al.*, used the rotating calipers to solve this problem in linear time [21]. The rotating calipers used with four rotating lines can also be used to find the maximum-area quadrilateral enclosing a convex polygon. Hosono, Meijer, and Rappaport [25] use it to compute the visibility graph of a set of non-intersecting translates of the same compact convex object in the plane.

2.4 The Maximum Distance Between Two Convex Polygons

The maximum distance between two convex polygons, *P* and *Q*, arises in several applications including pattern recognition, cluster analysis, and unsupervised learning [26]. It is defined as the largest distance determined by one point in *P* and another point in *Q*. As with the diameter problem, the search for the maximum distance may be restricted to the vertices of *P* and *Q*, denoted by $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots, q_n$, respectively. This maximum distance between *P* and *Q* is given by:

$$d_{\max}(P,Q) = \max\{d(p_i, q_j)\},\ i, j=1, 2, \ldots, n, \quad (1)$$

where maximization is done over all *i* and *j*, and $d(p_i, p_j)$ is the Euclidean distance between p_i and p_j. Bhattacharya and Toussaint [27] showed that two sets of points may be partitioned into eighteen subsets such that the maximum distance between the two sets is equal to the largest of the eighteen diameters of each of these subsets. Furthermore, the diameter of each subset may be computed with the rotating calipers algorithm applied to their convex hulls. If the two sets are convex polygons, their algorithm runs in O(*n*) time. However, a simpler and direct O(*n*) time algorithm for the case of convex polygons was later discovered by Toussaint and McAlear [28]. Their algorithm follows from the generalization of the notion of an antipodal pair of points for a single polygon, as illustrated in Fig. 4.

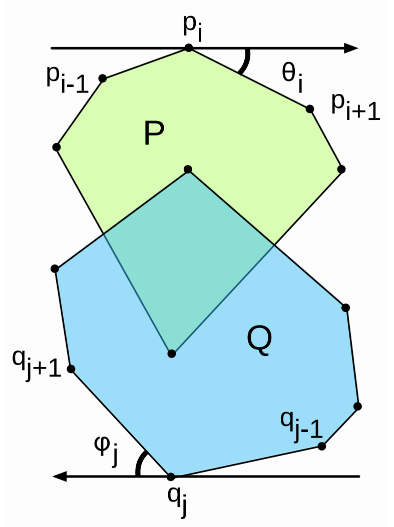

Fig. 4. The maximum distance between two convex polygons is determined by an *antipodal pair between* them.

An *antipodal pair between two sets P* and *Q*, is defined as a pair of vertices $p_i \in P$ and $q_j \in Q$, such that they admit parallel lines of support of *P* and *Q*, at p_i and q_j, respectively, with the added restrictions that the support lines are oriented in opposite directions, and each polygon lies to the right of its support line. Note that both polygons need not be contained in the parallel strip defined by the two support lines, as is the case for the configuration illustrated in Fig. 4. If one polygon is smaller than the other, or if it lies inside the other, it may protrude outside of this strip. It is shown in [28] that the maximum distance between the two sets is determined by an antipodal pair between the sets. There are only a linear number of such pairs, and they can be searched in O(*n*) time by rotating the calipers in the same manner as was done in the diameter algorithm. In Fig. 4, the pair p_i and q_j are one candidate pair, and the next candidate pair is obtained in O(1) time by rotating the calipers in a clockwise manner by the smaller of the angles θ_i and φ_j.

2.5 Minkowski Sum of Two Convex Polygons

Consider two points r and s in the plane, denoted by $r(x_r,y_r)$ and $s(x_s,y_s)$, specified by their x and y coordinates. The Minkowski sum (also called the vector sum) of r and s is the new point $t(x_t,y_t)$, where $x_t = x_r+x_s$ and $y_t = y_r+y_s$. The Minkowski sum of two convex polygons P and Q, denoted by $P \oplus Q$, is the set of points obtained by the Minkowski addition of each and every point in P with each and every point in Q. The Minkowski sums of polygons in the plane, and polyhedra in space, find application in spatial planning problems in the field of robotics [29], [30]. The Minkowski sum of two convex polygons, P and Q may be characterized in terms of their vertices, thus making it computable. In particular, $P \oplus Q$ is a convex polygon, and has at most $2n$ vertices, which are Minkowski sums of the vertices of P with those of Q. This characterization implies the following algorithm: first compute all $\Theta(n^2)$ pairwise Minkowski additions of the vertices of P and Q, and then compute the convex hull of the resulting set. Using an efficient $O(n \log n)$ time convex hull algorithm, such as Graham's algorithm [31], yields an $O(n^2 \log n)$ time algorithm for computing the Minkowski sum. However, a much faster $O(n)$ time algorithm may be obtained by exploiting a characterization of the Minkowski sum in terms of a modification of the notion of an antipodal pair of vertices. Two vertices $p_i \in P$ and $q_j \in Q$ are defined as being *co-podal* if, and only if, they admit parallel directed lines of support of P and Q, at $p_i \in P$ and $q_j \in Q$, respectively, such that the support lines are oriented in the same direction, and each polygon lies to the right of its support line, as illustrated in Fig. 5. The following characterization of the Minkowski sum of P and Q may now be obtained: the vertices of $P \oplus Q$ are the Minkowski sums of *co-podal* pairs of vertices of P and Q. This characterization permits the computation of $P \oplus Q$ by rotating the calipers in the manner as shown in Fig. 5, where the pair p_i and q_j is a candidate pair of vertices to be summed. The subsequent candidate pair is obtained in O(1) time by rotating the calipers in a clockwise manner by the smaller of the angles θ_i and φ_j. Let $z_k = p_i \oplus q_j$ denote a vertex of $P \oplus Q$ that has been computed. Then the succeeding vertex $z_{k+1} = p_i \oplus q_{j+1}$ if $\varphi_j < \theta_i$, $z_{k+1} = p_{i+1} \oplus q_j$ if $\theta_i < \varphi_j$, and $z_{k+1} = p_{i+1} \oplus q_{j+1}$ if $\theta_i = \varphi_j$. Since there are no more than $2n$ vertices in $P \oplus Q$ it follows that $O(n)$ time suffices to compute it.

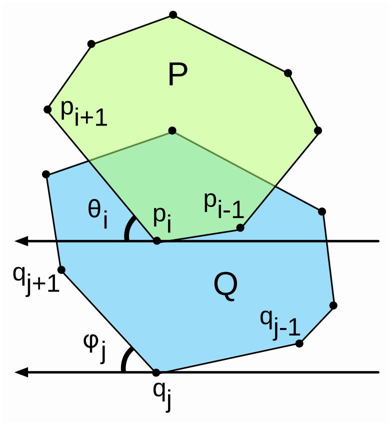

Fig. 5. The Minkowski sum of two convex polygons.

2.6 The Convex Hull of Two Convex Polygons

There exist applications where it is required to compute the convex hull of two convex polygons. For example, the divide and conquer approach to computing the convex hull of a set of n points in the plane repeatedly (recursively) merges the convex hulls of two smaller subsets of the points [32]. A linear time merge step is sufficient to yield an algorithm for the convex hull of the set that runs in $O(n \log n)$ time. The convex hull of two convex polygons, P and Q, consists of three types of edges: edges of P, edges of Q, and edges that connect vertices of P with those of Q. The latter edges are called *bridges* (also *common tangents*). In Fig. 6 (a) the dashed line L_B that supports polygons P and Q at vertices p_i and q_j, respectively, determines a bridge of the convex hull of P and Q. The convex hull of two convex polygons may therefore be computed by rotating clockwise two directed lines of support (one on each polygon) oriented in the same direction, such that each polygon is to the right of its line of support, as was done for the Minkowski sum problem. Whenever the two lines of support are not collinear (overlapping) one of them lies to the left of the other. For instance, in Fig. 6 (a), line L_Q lies to the left of L_P. Later in the process line L_P will lie to the left of L_Q. This implies that the lines must, at some time during the rotation, completely overlap, and whenever they do so they identify a bridge. Thus, the convex hull of the two polygons may be obtained by rotating the calipers a full revolution, and at each step outputting the vertex of either *P* or *Q*, that lies on the leftmost supporting line. Since each vertex of *P* and *Q* is visited only once, and there are O(*n*) bridges, O(*n*) time suffices for the entire computation.

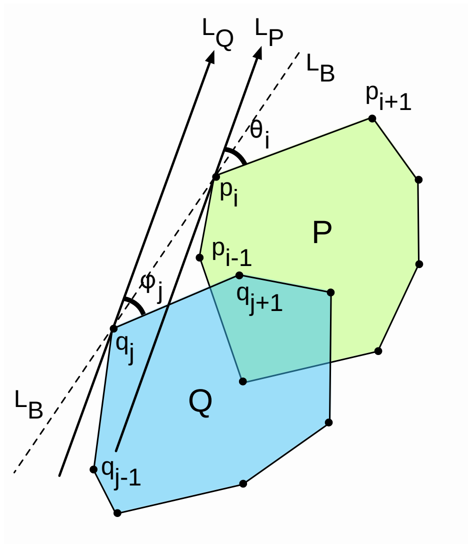

Fig. 6 (a). The convex hull of two convex polygons.

An alternate O(*n*) time algorithm for finding the common tangents of two convex polygons, that uses only one rotating line, was described by Bernard Chazelle in the context of a divide-and-conquer algorithm for finding the convex hull of *n* points which are first sorted by *x*-coordinate [33]. Assume the polygon *P* is left of *Q*, and refer to Fig. 6 (b). First the leftmost vertex $p_i \in P$ is found. Then the vertex $q_j \in Q$ is determined such that the line defined by p_i and q_j is tangent to *Q*, and *Q* lies beneath it. This line is the starting position for its clockwise rotation while remaining a support of *Q*. At the next step of this rotation, the line is advanced either to vertex p_{i+1} or to vertex q_{j+1}, depending on which angle of rotation is the smaller of the two. Throughout the rotation, each time a vertex of *P* and another of *Q* are both on the rotating line, the vertices are tested in O(1) time to determine if they constitute a bridge. A bridge has the property that the four angles determined by p_i and q_j and its adjacent vertices are all *right* turns. For example, in Fig. 6 (b) the angles formed by (p_{r-1}, p_r, q_s), (p_{r+1}, p_r, q_s), (p_r, q_s, q_{s-1},), (p_r, q_s, q_{s+1},) are all right turns, and therefore p_r and q_s determine a bridge.

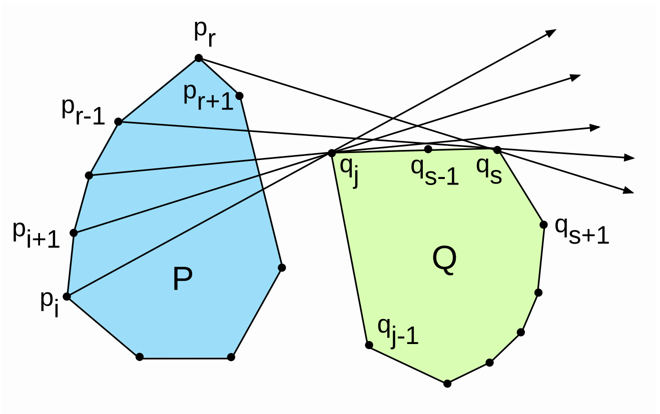

Fig. 6 (b). Finding the bridge of two convex polygons with one rotating caliper line.

Computing bridges between two convex polygons have also found application in solving special cases of the travelling salesperson problem (TSP), in which there are nested convex obstacles [34].

2.7 Intersecting Two Convex Polygons

Computing the intersection of two convex polygons is a fundamental operation that occurs in many applications. For example, the divide and conquer approach to computing the intersection of a set of *n* half-planes, repeatedly merges the intersections of two smaller subsets of the half-planes, which are (perhaps unbounded) convex polygonal sets [35]. A linear-time algorithm for intersecting two convex polygons, that uses the *slab method*, was described by Michael Shamos in his thesis [1], which leads to an O(*n* log *n*) time algorithm for the half-plane intersection problem. An alternate O(*n*) time algorithm was proposed by O'Rourke [36]. A transparently clear O(*n*) algorithm along with an easy proof of correctness, that uses the rotating calipers was presented in [37]. The algorithm exploits the fact that if two convex polygons intersect, then there exists an intersection point corresponding to each bridge in the convex hull of the two polygons, as illustrated in Fig. 7, where the dashed line L_B identifies a bridge determined by vertices $q_j \in Q$ and $p_i \in P$, and *x* denotes the intersection point corresponding to this bridge. The algorithm in [37] first determines if the two polygons intersect. If they do then the rotating calipers are used to find the convex hull as described in Sub-section 2.6, after which for each bridge the corresponding intersection point is found by a simple step-down procedure along the convex chains searching for the two intersecting edges.

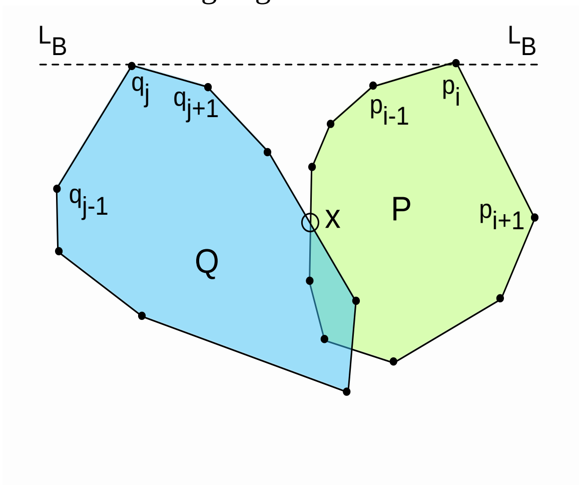

Fig. 7. A bridge and its corresponding intersection point *x* of two intersecting convex polygons.

2.8 The Critical lines of Support of Two Convex Polygons

Given two disjoint convex polygons *P* and *Q*, the critical support lines are the two lines that separate *P* and *Q*, such that they are both support lines for *P* and *Q*. In Fig. 8 the two critical support lines, L_P and L_Q, are the dashed lines. One line contains vertices p_{i+1} and q_j, and the other contains $p_{i\text{-}1}$ and $q_{j\text{-}1}$. Intuitively, the two critical support lines may be obtained by rotating any line that separates the two polygons, in clockwise and counterclockwise directions as much as possible, while maintaining the separability of the two polygons. Critical support lines find application to a variety of problems, some of which are described in the following subsections. For two convex polygons they can be computed in linear time using the rotating calipers in a manner similar to that of computing the maximum distance, by initially placing the two parallel lines oriented in opposite directions such that each polygon is to the right of its support line, as in Fig. 4. During the rotation phase of the calipers the critical support lines are detected each time that both support lines are collinear (overlap each other), as in Fig. 8.

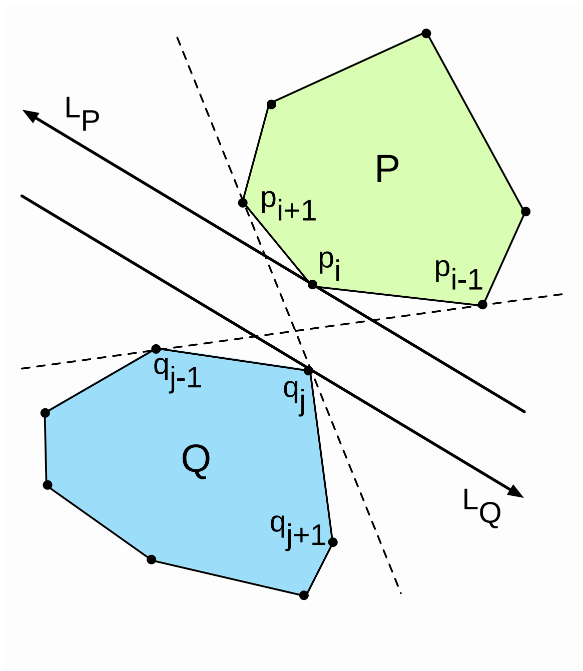

Fig. 8. The two critical support lines (dashed) determined by two disjoint convex polygons.

2.9 The Widest Separating Strip Between Two Convex Polygons, and Machine Learning

In the context of pattern classification and machine learning, two polygons *P* and *Q* may be thought of as regions in a feature space that enclose points of the training data for a two-class discrimination problem. Any separating line then is a linear classification rule that can be used to classify future patterns (points) depending on whether they lie on one side or the other of this line. In order for the classifier to make more confident decisions it is desired to pick the separating line that is furthest from the two polygons. Such a line may be chosen as the centerline of the widest empty strip that separates the polygons. Such classifiers are referred to as large-margin classifiers [38], (also wide separation of sets [39]) and make up the geometric backbone of support vector machines [40]. For the case of two planar convex polygons the widest empty strip is determined by either one vertex from each polygon, or by a vertex of one polygon and an edge of the other. Furthermore, this strip may be detected when the support lines are oriented in between the directions of the two critical support lines, and may be found in O(n) time with the rotating calipers.

The widest empty strip problem is closely related to the problems of fitting lines to data [41], finding transversals of sets [42], and linear approximation of objects [43], [44], all of which have been solved efficiently using the rotating calipers. Similar problems occur when data is not linearly separable. In this setting Aronov *et al*., [45] use the rotating calipers to find linear "separators" that minimize a variety of different measures of error for the points misclassified by the separator.

2.10 The Grenander Distance Between Two Convex Polygons

Ulf Grenander [46] proposed that the distance between two disjoint convex polygons be measured by comparing the lengths of the connecting segments of their critical support lines, to the lengths of the polygonal chains spanned by these segments. To be more precise let the critical support lines be supported at p_i and p_r in P, and q_j and q_s in Q, and refer to Fig. 9.

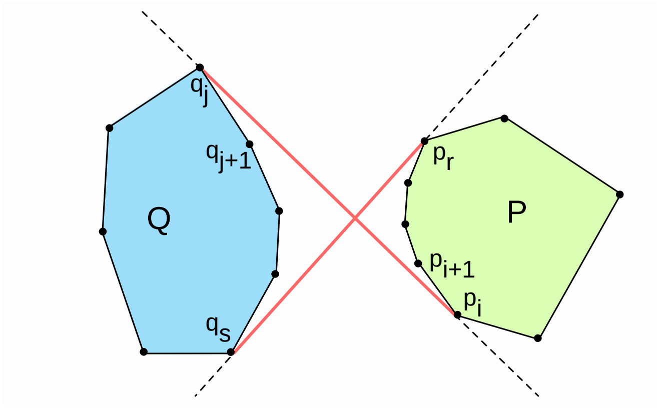

Fig. 9. The Grenander distance between two disjoint convex polygons.

The support vertices of the critical support lines partition the convex polygons into two polygonal chains: the *inner chains* visible from the intersection point of the support lines, and the complementary *outer chains* invisible from the intersection point. The *supporting chords* of P and Q are the line segments that connect q_j to p_i and p_r to q_s, shown as bold red line segments in Fig. 9. The Grenander distance is defined as the sum of the lengths of the two supporting chords, less the sum of the lengths of all the edges of the polygons that belong to both inner chains. Clearly, once the critical support lines have been computed with the rotating calipers, an additional O(n) time suffices for the computation of the lengths of the chords and chains, and therefore the Grenander distance may be computed in O(n) time overall.

2.11 Optimal Strip Separation in Medical Imaging and Solid Modeling

In several contexts such as medical imaging it is required to construct a solid model by stitching parallel polygonal slices together. A problem arises during the interpolation when the solid object is bounded by a single contour in one slice, and two contours in the adjoining slice. The computational geometric problem that results is the following [47]. Given two linearly separable polygons P and Q, and a third convex polygon R, it is required to compute the separating strip between P and Q, that covers the largest area of R. Barequet and Wolfers [47] present a linear-time algorithm for computing this optimum strip using the rotating calipers. They also consider the case when the polygon R is not convex, but in this case the running time of their algorithm is quadratic in the size of the input.

2.12 Aperture Angle Optimization for Visibility Problems in Graphics and Computer Vision

In several disciplines such as computer graphics, computer vision, robotics, operations research, visual inspection, and accessibility analysis in the manufacturing industry, the notion of visibility is fundamental. Often models assume that a camera can see in all directions, in effect idealizing the camera's aperture angle to 360º. In a more realistic model the aperture angle is much smaller than 360º. Furthermore, if the camera is movable, it is desirable to compute the maximum and minimum aperture angles that the camera may need as it travels in a constrained space. Let P and Q be two disjoint convex polygons in the plane. For a given a point x in P, the aperture angle at x with respect to Q is defined as the angle of the cone with apex at x, that contains Q, and has its two rays that emanate from x tangent to Q. The critical lines of support play singular roles in computing the extreme aperture angles, and may be efficiently computed with the rotating calipers [48].

2.13 Wedge Placement Optimization Problems

Wedge placement optimization problems arise in several contexts such as visibility with bounded aperture angles, and layout design of parts in stock cutting for manufacturing. A wedge W may be thought of simply as an unbounded cone with a fixed angle θ at its apex. Given an n-vertex polygonal region, such as R in Fig. 10, we are interested in computing the entire region where a camera (the apex of the cone) with aperture angle θ, may be positioned so that it is as close as possible to R. Such a region is bounded by a concatenation of arcs (called the *wedge cloud*) determined by the apex of the cone as it travels around R maintaining contact with R. Fig. 10 shows three points on the cloud, x_i, x_j and x_k which are camera locations with a fixed aperture angle θ. The original rotating calipers can be generalized so that the lines of support form any fixed angle to each other.

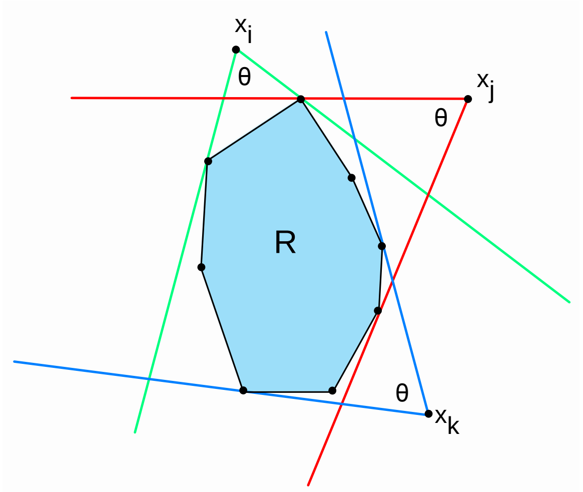

Fig. 10. Wedge fitting and the *wedge cloud*.

Teichmann [49] used the above generalization of the calipers to compute the wedge cloud of a convex polygon in O(n) worst-case time. Note that to maintain contact with the polygon R, a wedge must both rotate and translate.

2.14 Nonparametric Decision Rules

In the nonparametric discrimination problem we are given training data that belong to different classes, and it is desired to classify new incoming data into their respective classes. Jean-Paul Rasson and Grandville [50] proposed a geometric approach to the design of such a decision rule. Consider the two-dimensional, two-class problem, in which the classes are linearly separable, and refer to Fig. 11. In the training phase of the classifier the convex hulls P and Q of each class are computed and stored.

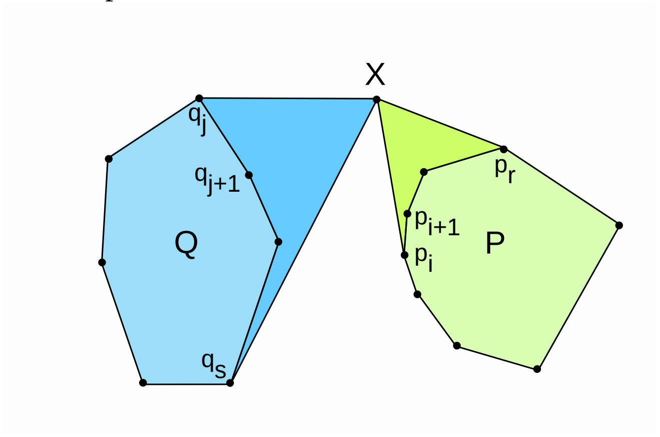

Fig. 11. A geometric non-parametric decision rule.

The decision rule for a new incoming pattern X is as follows. If X lies in P (or Q) it is classified as belonging to class P (or Q). If X lies outside both polygons it is classified to the class associated with the nearest polygon, where nearest is defined in terms of the area distance between X and each of the polygons. More precisely, the distance between X and Q is the absolute value of the difference between the area of polygon Q and the area of the convex hull of $Q \cup X$. Similarly, the distance between X and P is the absolute value of the difference between the area of polygon P and the area of the convex hull of $P \cup X$. These areas are colored dark blue and dark green, respectively. In this example the dark green area is smaller than the dark blue area, and therefore X would classified as belonging to class P. The four lines that connect X to the polygons are critical support lines between X and the polygons, where X may be considered as a degenerate single-vertex polygon, and may be computed in linear time with the rotating calipers.

2.15 Nice Triangulations and Quadrangulations of Planar Sets of Points

A convex polygonal *annulus* is the region in between two properly nested convex polygons, such as the blue shaded region consisting of the polygon *Q* less the interior of the green shaded polygon *P* (Fig. 12). This region admits a very simple triangulation by means of the rotating calipers [51]. A triangulation of a polygonal region is a partition of the region's interior into as many triangles as possible, obtained by inserting diagonals between pairs of vertices, without allowing any edges to properly cross each other. Initially two parallel lines of support, oriented in the same direction, are constructed through a pair of extreme vertices such as the vertices of *P* and *Q* with lowest *y*-coordinates, such that the polygons lie to the right of the lines (Fig. 12). As the calipers are rotated clockwise, the pairs of vertices that come into simultaneous contact with the lines of support are connected with an edge. The first few edges connected by this algorithm are shown in red in Fig. 12. In general a region admits many triangulations, some of which may have long edges, acute triangles, or other properties deemed undesirable for some applications such as mesh generation. One attractive property of the triangulation of the annulus obtained with the rotating calipers algorithm is that the triangulation tends to be nice, in the sense that the resulting triangles tend to be nearly regular. Furthermore, the method can be applied to more general problems such as obtaining nice triangulations of sets of points. One way of doing this is to first compute all the convex layers of the set [52], which yields a nested collection of annuli, each of which can be triangulated with the rotating calipers [53]. Another method involves first computing a spiral polygonal chain spanning the points [54], and then triangulating this spiral with the rotating calipers [53]. This latter triangulation also has the added nice property that it is *serpentine*, *i.e.*, its dual graph is a chain. In a similar approach the rotating calipers have been used to count and enumerate pointed pseudo-triangulations using the greedy diagonal flipping algorithm, by rotating the calipers along the interiors of two pseudo-triangles [55].

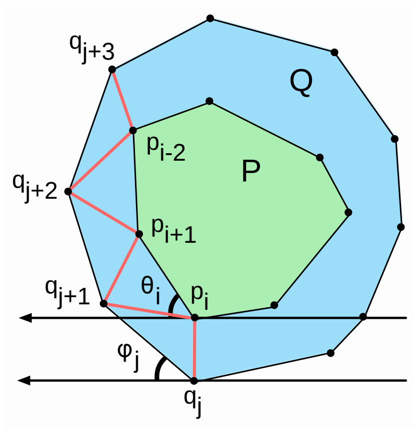

Fig. 12. Triangulating a polygonal convex annulus.

2.16 The Rotating Caliper Graph

The rotating caliper procedure, besides solving many problems directly, has also given rise to a new geometric data structure called the *rotating caliper graph* (RCG) that has found several applications to the design of efficient geometric algorithms [56]. Fig. 13 shows a convex polygon and its rotating caliper graph. The edges of the RCG consist of edges that connect the antipodal pairs of vertices of the polygon that are visited during a full rotation of the parallel lines of support in the diameter algorithm. The RCG may also be defined for arbitrary point sets, but in this case the caliper it only connects the convex hull vertices of the set. The RCG has several interesting and useful properties. One of these is of course that it contains a small linear number of edges. David Eppstein has exploited this property to obtain efficient algorithms for maintaining the RNG itself as well as the width and diameter of points sets under insertion and deletion operations of individual points [56]. Another interesting property of the RCG is that it is a *thrackle*. John Conway defined a thrackle as a planar graph embedding (drawing) with the property that every pair of edges intersects at a single point [57], [58]. In the case of the RCG each pair of edges contains an intersection point at either a common endpoint of the edges, or in their interiors. Since the edges are straight line segments, this type of thrackle is called a *linear thrackle*.

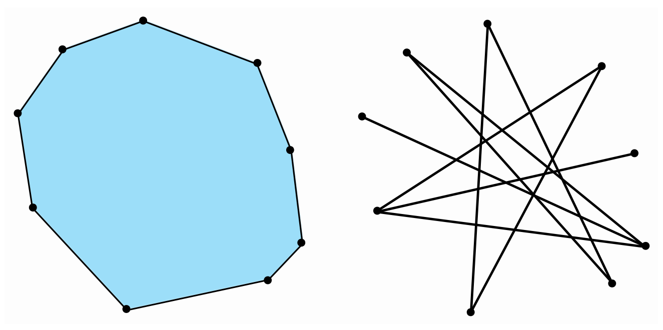

Fig. 13. Convex polygon and its rotating caliper graph.

3 THE ROTATING CALIPERS IN 3-D

3.1 The Convex Hull of a Set of Points

The natural generalization of the process of rotating a line constrained to remain tangent to a convex polygon in two dimensions, to 3-dimensional space, is rotating a plane that remains tangent to a convex polyhedron. To specify the direction of rotation uniquely the plane should be tangent to two points in space, which determine the axis of rotation of the plane. The simplest example of such an application is in the computation of the convex hull of *n* points in 3-D. A simple algorithm for this purpose first sorts the points by increasing *x*-coordinate, and then constructs the convex hull incrementally by inserting one point at a time in the sorted order in a growing convex hull. To initialize the process the four points with smallest *x*-coordinates may be connected to form a tetrahedron, which serves as the convex hull of the four points. At every subsequent step in the execution of this algorithm we have a convex polyhedron P_{i-1} (the convex hull of the first p_{i-1} points considered), and the *i-th* point p_i that lies to the right of P_{i-1}. The process of inserting p_i is illustrated in Fig. 14, where the *z*-axis (not shown) is orthogonal to the *xy*-plane, in the direction of the viewer. First a plane orthogonal to the *xy*-plane that contains p_i and is tangent to P_{i-1} at point *a*, is found. The projection of this plane onto the *xy*-plane is shown in Fig. 14 as the line through p_i and *a*. Using this line as the axis of rotation, the plane is rotated until it collides with another vertex of P_{i-1} (vertex *b* in the Figure), thus creating a triangle (p_i, *a*, *b*), shown shaded in transparent green. This triangle becomes a face of the new convex hull, and the line through p_i and *b* becomes the new axis of rotation for the rotating plane. This process continues until the rotating plane arrives at the edge (p_i, *a*) again. The new faces inserted in the new convex hull form a cone with apex at p_i. Finally, the faces of the previous convex hull that lie inside this cone and are visible from p_i are deleted, to complete the new convex hull. The addition of each point to the growing convex hull can be done in O(*n*) time leading to an algorithm with O(n^2) time overall [59]. If a divide-and-conquer approach is used to compute the convex hull, a similar approach may be used in which the rotating plane wraps around two convex polyhedral. Such a procedure, originally proposed by Preparata and Hong [60], leads to an algorithm that runs in O(*n* log *n*) time.

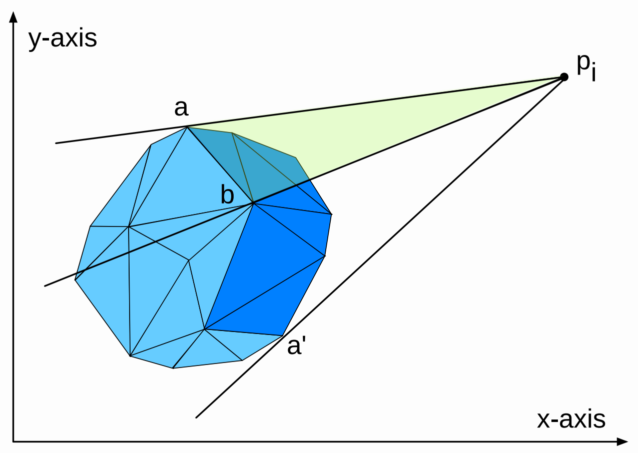

Fig. 14. Computing the convex hull in 3-D incrementally using a rotating plane.

The above algorithm for inserting a new vertex into the 3-dimensional convex hull of points in space can also be used to solve other problems. In the hidden-surface problem in computer graphics the point p_i represents a point light source, and the rotating calipers then yields the portion of a convex polyhedron illuminated by that point. In the problem of non-parametric classification the point p_i represents a new pattern to be classified. Here the volume of the region inside the cone determined by p_i and the portion of the convex polyhedron visible from p_i is a measure of the distance from p_i to the polyhedron, thus generalizing the 2-dimensional approach of Rasson and Grandville [50] to three dimensions.

3.2 Finding Minimal Enclosing Boxes

A natural generalization of the problem of finding the minimum-area rectangle that encloses a given convex polygon, is finding the minimum-volume box that contains a given convex polyhedron. However, unlike its two-dimensional rectangular counterpart, the minimum-volume box bounding a convex polyhedron need not have one of its faces flush with a face of the polyhedron. Fig. 15 illustrates the minimum-volume box enclosing a regular tetrahedron (shaded in blue) in which no faces of the box are flush with faces of the tetrahedron. Note however, that in this example every face of the box is flush with an edge of the tetrahedron. In fact, O'Rourke [61] showed that a box of minimum volume enclosing any convex polyhedron must have at least two adjacent faces flush with edges of the polyhedron. This characterization allows O'Rourke to design an algorithm that performs a type of three-dimensional caliper rotation starting with every pair of edges of the polyhedron, that runs on O(n^3) time. The caliper in effect consists of two planes orthogonal to each other that rotate in unison, reminiscent of the wedge-placement optimization problems in two dimensions.

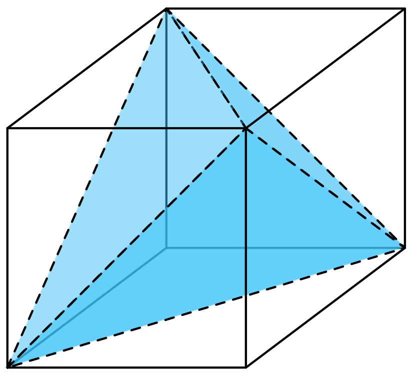

Fig. 15. The minimum-volume bounding box of a regular tetrahedron.

Motivated by the fact that O'Rourke's O(n^3) exact algorithm is complicated and slow in practice, Chang, Gorissen, and Melchior [62] proposed an approach formulated as an unconstrained optimization problem. They compare their method to previous methods, and conclude that their algorithm is faster and more reliable.

In practice one may be satisfied with an even simpler algorithm that yields an approximation to the smallest bounding box. Such a method proposed by O'Rourke constructs the minimum-volume enclosing box with the constraint that one of the faces of the convex polyhedron lie flush with a face of the box. In this version of the problem it suffices to project the convex polyhedron onto every plane determined by a face of the polyhedron, projecting the polyhedron onto that plane, and solving the minimum area enclosing rectangle problem with the planar rotating calipers. The resulting rectangle then forms the base of the box, the height of which is determined by the vertex of the polyhedron that is furthest from the base. This simple approximation algorithm runs in O(n^2) time. Approximation algorithms that run in almost linear time exist, but they are more complicated, and use for example, principal components rather than the rotating calipers [63], [64]. Additional optimization methods for finding bounding boxes and a discussion of their application to *brachytherapy* (internal radiotherapy) may be found in [65].

4 CONCLUSION

This paper has focused on extensions of the rotating calipers, and their applications to a variety of geometric problems in the plane and three dimensions. While the problem of computing the *minimum* distance between sets, resembles superficially the *maximum* distance problem, it is conspicuously absent here. This is because the problem appears difficult to crack with the rotating calipers [66]. The rotating calipers have also been generalized to work on two-dimensional manifolds other than planes, such as spheres and cones [67]. The reader is referred to the web page and thesis of Hormoz Pirzadeh [68], for details of some characterization proofs, and animation applets that help to visualize several of the 2-dimensional algorithms that have been described in this paper.
http://cgm.cs.mcgill.ca/~orm/welcome.html

4 ACKNOWLEDGEMENTS

This research was supported by a grant from the Provost's Office of New York University Abu Dhabi, through the Faculty of Science, in Abu Dhabi, The United Arab Emirates.

5 REFERENCES

1. M.I. Shamos, Computational geometry, Ph.D. thesis, Yale University, 1978.
2. G. T. Toussaint, “Solving geometric problems with the rotating calipers,” Proceedings of IEEE MELECON'83, Athens, Greece, May 1983, pp. A10. 02/1-4.
3. B. K. Bhattacharya, and G. T. Toussaint, “Fast algorithms for computing the diameter of a finite planar set,” The Visual Computer, vol. 3, no. 6, pp. 379-388, 1988.
4. D. Avis, G. T. Toussaint, and B. K. Bhattacharya, “On the multimodality of distances in convex polygons,” Computers and Mathematics with Applications, vol. 8, no. 2, pp. 153-156, 1982.
5. K. Ichida, and T. Kiyono, “Segmentation of plane curves,” Transactions of the Institute of Electronics and Communication Engineers of Japan, vol. 58-D, pp. 689-696, 1975.
6. G. T. Toussaint, “On the complexity of approximating polygonal curves in the plane,” Proceedings of IASTED International Symposium on Robotics and Automation, 1985, pp. 59-62.
7. G. T. Toussaint, “Complexity, convexity, and unimodality,” International Journal of Computer and Information Sciences, vol. 13, no. 3, pp. 197-217, 1984.
8. Y. Kurozumi, and W. A. Davis, “Polygonal approximation by the minimax method,” Computer Graphics and Image Processing, vol. 19, pp. 248-264, 1982.
9. H. Imai, and M. Iri, “Polygonal approximations of a curve: Formulations and solution algorithms,” in Computational Morphology, G. T. Toussaint, (Ed.), Amsterdam, The Netherlands: North-Holland, 1988, pp. 71-96.
10. M. E. Houle, and G. T. Toussaint, “Computing the width of a set,” IEEE Transactions Pattern Analysis & Machine Intelligence, vol. 10, no. 5, pp. 761-765, 1988.
11. S. Drazić, N. Ralević, and J. Zunić, “Shape elongation from optimal encasing rectangles,” Computers and Mathematics with Applications, vol. 60, pp. 2035-2042, 2010.
12. P. L. Rosin, “Measuring shape: ellipticity, rectangularity, and triangularity,” Machine Vision and Applications, vol. 14, pp. 172-184, 2003.
13. J. Žunić and P. L. Rosin, “A new convexity measure for polygons,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 7, pp. 923-934, 2004.
14. J. Žunić and P. L. Rosin, “Rectilinearity measurements for polygons,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 9, pp. 1193-1200, 2003.
15. J. Žunić and P. L. Rosin, and L. Kopanja, “On the orientability of shapes,” IEEE Transactions on Image Processing, vol. 15, no. 11, pp. 3478-3487, 2006.
16. H. Freeman and R. Shapira, “Determining the minimum-area encasing rectangle for an arbitrary closed curve,” Communications of the A.C.M., vol. 18, pp. 409-413, 1975.
17. F. C. A. Groen, P. W. Verbeek, N. de Jong, and J. W. Klumper, “The smallest box around a package,” Pattern Recognition, vol. 14, nos. 1-6, pp. 173-178, 1981.
18. G. T. Toussaint, “Pattern recognition and geometrical complexity”, Proceedings of the Fifth International Conference on Pattern Recognition, Miami Beach, December 1980, pp. 1324-1347.
19. T. Needham, “A visual explanation of Jensen’s inequality,” American Mathematical Monthly, vol. 100, no. 8, pp. 145-185, 1993.
20. V. Klee, and M. C. Laskowski, “Finding the smallest triangles containing a given convex polygon,” Journal of Algorithms, vol. 6, no. 3, pp. 359-375, 1985.
21. J. O’Rourke, A. Aggarwal, S. Maddila, and M. Baldwin, “An optimal algorithm for finding minimal enclosing triangles,” Journal of Algorithms, vol. 7, pp. 258-269, 1986.
22. S. Das, P. P. Goswami, and C. S. Nandy, “Smallest k-point enclosing rectangle and square of arbitrary orientation,” Information Processing Letters, vol. 94, pp. 259-266, 2005.
23. J. S. B. Mitchell, and V. Polishchuk, “Minimum-perimeter enclosures,” Information Processing Letters, vol. 107, pp. 120-124, 2008.
24. D. Mount, “The densest double-lattice packing of a convex polygon,” J. E. Goodman, R. Pollack, W. Steiger, (Eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 6, 1999, pp. 245-262.
25. K. Hosono, H. Meijer, and D. Rappaport, “On the visibility graph of convex translates,” Discrete Applied Mathematics, vol. 113, pp. 195-210, 2001.
26. R. O. Duda, and P. E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.
27. B. K. Bhattacharya, and G. T. Toussaint, “Efficient algorithms for computing the maximum distance between two finite planar sets,” Journal of Algorithms, vol. 14, pp. 121-136, 1983.
28. G. T. Toussaint, and J. A. McAlear, “A simple O(*n* log *n*) algorithm for finding the maximum distance between two finite planar sets,” Pattern Recognition Letters, vol. 1, pp. 21-24, 1982.
29. T. Lozano-Perez, “An algorithm for planning collision-free paths among polyhedral obstacles,” Communications of the ACM, vol. 22, pp. 560-570, 1979.

30. T. Lozano-Perez, "Spatial planning: A configuration space approach," IEEE Transactions on Computers, vol. 32, no. 2, pp. 108–120, 1983.
31. R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," Information Processing Letters, vol. 1, pp. 132-133, 1972.
32. F. P. Preparata, and S. Hong, "Convex hulls of finite sets of points in two and three dimensions," Communications of the ACM, vol. 20, pp. 87-93, 1977.
33. B. Chazelle, "Approximation and decomposition of shapes," in Advances in Robotics 1: Algorithmic and Geometric Aspects of Robotics, (J.T. Schwartz, C.K. Yap, eds), Lawrence Erlbaum Associates (1987), pp. 145-185.
34. J. Abrahamson, A. Shokoufandeh, and P. Winter, "Euclidean TSP between two nested convex obstacles," Information Processing Letters, vol. 95, pp. 370-375, 2005.
35. F. P. Preparata, and D. E. Muller, "Finding the intersection of *n* half-spaces in time O(*n* log *n*)," Theoretical Computer Science, vol. 8, no. 1, pp. 45-55, 1979.
36. J. O'Rourke, "A new linear algorithm for intersecting convex polygons," Computer Graphics and Image Proccessing, vol. 19, pp. 384-391, 1982.
37. G. T. Toussaint, "A simple linear algorithm for intersecting convex polygons," The Visual Computer, vol. 1, pp. 118-123, 1985.
38. Y. Freund, and R. E. Schapire, "Large margin classification using the perceptron algorithm," Machine Learning, vol. 37, no. 3, pp. 277-296, 1999.
39. M. E. Houle, "Algorithms for weak and wide separation of sets," Discrete Applied Mathematics, vol. 45, pp. 139-159, 1993.
40. M. E. Mavroforakis, and S. Theodoridis, "A geometric approach to support vector machine (SVM) classification," IEEE Transactions on Neural Networks, vol. 17, no. 3, pp. 671-682, 2006.
41. C. Rey, and R. Ward, "On determining the on-line minimax linear fit to a discrete point set in the plane," Information Processing Letters, vol. 24, pp. 97-101, 1987.
42. B. K. Bhattacharya, J. Czyzowicz, P. Egyed, I. Stojmenovic, G. T. Toussaint, and J. Urrutia, "Computing shortest transversals of sets," International Journal of Computational Geometry and Applications, vol. 2, no. 4, pp. 417-436, 1992.
43. J.-M. Robert, and G. T. Toussaint, "Linear approximation of simple objects," Proceedings 9th Annual Symposium on Theoretical Aspects of Computer Science (STACS'92), Cachann France, February 1992, A. Finkel and M. Jantzen (eds.), Lecture Notes in Computer Science #577, pp. 233-244.
44. J.-M. Robert, and G. T. Toussaint, "Computational geometry and facility location," Proceedings of the International Conference on Operations Research and Management Science, Manila, The Philippines, December 11-15, 1990, pp. B-1 to B-19.
45. B. Aronov, D. Garijo, Y. Núñez-Rodrígues, C. Seara, and J. Urrutia, "Measuring the error of linear separators on linearly inseparable data," Discrete Applied Mathematics, vol. 160, 2012.
46. U. Grenander, Pattern Synthesis, Springer-Verlag, New York, 1976.
47. G. Barequet, and B. Wolfers, "Optimizing a strip separating two polygons," Graphical Models and Image Processing, vol. 60, no. 3, pp. 214-221, 1998.
48. P. Bose, F. Hurtado-Diaz, E. Omaña-Pulido, J. Snoeyink, and G. T. Toussaint, "Some aperture-angle optimization problems," Algorithmica, vol. 33, pp. 411-435, 2002.
49. M. Teichmann, Wedge Placement Optimization Problems, M.Sc. Thesis, School of Computer Science, McGill University, October 1989.
50. J. P. Rasson, and V. Granville, "Geometrical tools in classification," Journal of Computational Statistics and Data Analysis, vol. 2, pp. 105-123, 1996.
51. G. T. Toussaint, "New results in computational geometry relevant to pattern recognition in practice," In Pattern Recognition in Practice II, E. S. Gelsema and L. N. Kanal, (Eds.), North-Holland, Amsterdam, The Netherlands, 1986, pp. 135-146.
52. B. Chazelle, "On the convex layers of a planar set," IEEE Transactions on Information Theory, vol. 31, pp. 509-517, 1985.
53. G. T. Toussaint, "Quadrangulations of planar sets," Proceedings of the 4th International Workshop on Algorithms and Data Structures (WADS'95), Kingston, Canada, August 16-18, 1995, pp. 218-227.
54. J. Iwerks, and J. S. B. Mitchell, "Spiral serpentine polygonization of a planar point set," In Encuentros de Geometria Computacional, (Hurtado Festschrift), A. Márquez, *et al*. (Eds.): LNCS 7579, 2012, pp. 146-154.
55. H. Brönnimann, L. Kettner, M. Pocchiola, and J. Snoeyink, "Counting and enumerating pointed pseudo-triangulations with the greedy flip algorithm," In: Algorithm Engineering and Experiments (ALENEX'05), Vancouver, BC, Canada, January 2005.
56. D. Eppstein, "Average case analysis of dynamic geometric optimization," Computational Geometry: Theory and Applications, vol. 6, pp. 45-68, 1996.
57. R. Fulek and J. Pach, "A computational approach to Conway's thrackle conjecture," Computational Geometry: Theory and Applications, vol. 44, pp. 345–355, 2011.
58. J. Pach and E. Sterling, "Conway's conjecture for monotone thrackles," American Mathematical Monthly, vol. 118, no. 6, pp. 544–548, 2011.
59. S. L. Devadoss, and J. O'Rourke, Discrete and Computational Geometry, Princeton University Press, Princeton, New Jersey, 2011.
60. F. P. Preparata, and S. J. Hong, "Convex hulls of finite sets of points in two and three dimensions," Communications of the ACM, vol. 20, no. 2, pp. 87–93, 1977.

61. J. O'Rourke, “Finding minimal enclosing boxes,” International Journal of Computer and Information Science, vol. 14, pp. 183-199, 1985.
62. C.-T. Chang, B. Gorissen, and S. Melchior, “Fast oriented bounding box optimization on the rotation group SO(3;R),” ACM Transactions on Graphics, vol. 30, no. 5, October 2011, Article 122.
63. G. Barequet, and S. Har-Peled, “Efficiently approximating the minimum volume bounding box of a point set in three dimensions,” Journal of Algorithms, vol. 38, pp. 91-109, 2012.
64. I. T. Jolliffe, Principal Component Analysis. Springer, New York, NY, USA, 2002.
65. M. Lahanas, T. Kemmerer, N. Milickovic, K. Karouzakis, D. Baltas, and N. Zamboglou, “Optimized bounding boxes for three-dimensional treatment planning in brachytherapy,” Medical Physics, vol. 27, no. 10, pp. 2333-2342, October 2000.
66. G. T. Toussaint, and B. K. Bhattacharya, “Optimal algorithms for computing the minimum distance between two finite planar sets,” Pattern Recognition Letters, vol. 2, pp. 79-82, 1983.
67. C. Grima, and A. Márquez, Computational Geometry on Surfaces, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.
68. H. Pirzadeh, Computational Geometry with the Rotating Calipers, M.Sc. Thesis, School of Computer Science, McGill University, Montreal, Canada, November 1999.