The Translation-Scale-Rotation Diagram for Point-Containing Placements of a Convex Polygon*

Gill Barequet† Yuval Scharf†

Abstract

We present a diagram that captures containment information for scalable rotated and translated versions of a convex polygon. For a given polygon P and a contact point q in a point set S, the diagram parameterizes possible translations, rotations, and scales of the polygon in order to represent containment regions for each additional point v in S. We present geometric and combinatorial properties for this diagram, and describe how it can be computed and used in the solution of several geometric problems.

1 Introduction

Given a set of points in the plane and a convex polygon, we consider problems in which we try to cover the points (or a maximum number of them) with the given polygon. Depending on the specific problem, we allow the polygon to be translated, rotated, scaled, or any combination of the above.

One set of problems that has received considerable attention in the literature of computational-geometry is the placement of a polygon so that it contains a given point set (or a subset of it) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Problem variants include placement of a polygon by translation only, placement by translation and rotation, or placement allowing some geometric transformation (such as scaling, offsetting, or perspective transformations). Optimization variants of the problem include maximization of the number of contained points as well as minimization of the size of the polygon (or polygonal annulus) by either scaling or offsetting.

One such problem, that of finding a translation and rotation of a convex polygon P that maximizes the number of contained points from a given input set S, was studied by Dickerson and Scharstein [6]. As part of their solution they presented the so-called *rotation diagram.* The rotation diagram $R_{P,q}$ represents all possible placements of a convex polygon P in contact with a particular point $q \in S$. This 2D diagram parameterizes translations along one axis and rotations along the other axis. For every other point $v \in S$, the diagram has a region of all placements of P containing v. The cited work describes the combinatorial and geometric properties of rotation diagrams.

Barequet and Dickerson [3] followed that work and created the so-called *translation-scale* diagrams. These diagrams are also 2D, in which one axis represents scaling of the polygon and the other axis represents translation of the polygon. It is shown in [3] how a few containment problems can be solved using those diagrams.

In [3, 6], and also here, the diagrams emphasize placements of a polygon that are *in contact* with some point of the set. This is because any not-in-contact placement of the polygon that optimizes some point-containment problem can be modified to an in-contact placement without altering the set contained in the polygon.

In this paper we first explore polygon placements that allow scaling, rotating, and translating the polygon. In particular, we present a 3D containment diagram similar to that of [3, 6], but representing all of translation, rotation, and scale. We show some properties of the diagram and solve some problems by using the diagram.

2 The Diagram

Our goal is to combine the two diagrams presented in [3, 6]. Given a convex m-gon P and a set S of n points, we want to build a 3D diagram that describes all the possibilities to cover S with P when we allow translation, scale, and rotation of P.

In [3] TS diagrams are built as follows. A diagram is built for every point $q \in S$. The diagram represents all the different ways to put a copy of P such that q is in contact with the boundary of P. For every other point $v \in S$ we draw a region in $D_{P,q}$, the diagram of q. The x-axis parameterizes the translation. Every point on the x-axis represents a point on the boundary of P that will cover q. The y-axis parameterizes the scale. Every point on the y-axis represents the inverse of the scale of the polygon P needed to cover the point v after the translation, if possible.

We extend the TS diagram of [3] to the third (z) dimension (the height), where the additional dimension represents rotations of P. Consider a TS diagram, and

*Work on this paper by both authors has been supported in part by the European FP6 Network of Excellence Grant 506766 (AIM@SHAPE). Work by the first author has also been supported in part by Grant 01-06-00032 under the Korean-Israeli Research Cooperation.

†The Technion—Israel Institute of Technology, Haifa 32000, Israel, [barequet,scharf]@cs.technion.ac.il

slightly rotate P clockwise. That is, advance slightly along the z direction in the diagram. How will the new TS diagram look like?

For ease of exposition we temporarily restrict our attention to only one region in the diagram. So there are only two points in the set, q, v, and we draw the region created for v in the diagram of q. Since P is rotated clockwise, $\overrightarrow{qv}$ hits the vertices of the upper chain earlier relative to the x-coordinate of the hitting points before the rotation. Therefore, the vertices of the upper chain will move to the left in the diagram. As we continue to rotate P, vertices of the upper chain will pass above vertices of the lower chain. After rotating P enough, the leftmost vertex of the upper chain becomes the leftmost vertex of the lower chain, and the rightmost vertex of the lower chain becomes the rightmost vertex of the upper chain. As we continue to rotate P, the TS diagram will scroll counter-clockwise. Note that as the vertices move to the left, their heights are changed as well.

If we fix the scale and consider the two-dimensional diagram spanned by the two other axes, we will get the translation-rotation diagram [6] for that scale. As the scale (of the plane) grows, there is less freedom in translating and rotating the polygon P for covering the point v (while keeping q on the boundary of P), and hence the translation-rotation diagram shrinks. The highest point in the three-dimensional diagram represents the maximum scale of the plane (that is, the minimum scale of P) that allows the coverage of v by P. The two other coordinates specify the translation and rotation. Clearly, in this configuration the points v and q lie at the two endpoints of the diameter of the rotated version of P.

3 Properties

Define the *structure* of a TS diagram to be the order of the vertices of the diagram according to their x coordinates (along the translation axis of the diagram).

Theorem 1 *A translation-scale-rotation diagram is made of $O(m^2)$ slices with different structures.*

Proof. A change in the structure of the diagram is caused by rotating P such that if the vector $\overrightarrow{qv}$ originates from a vertex of P, it points in the direction of another vertex of P. Clearly, there are $O(m^2)$ critical rotation angles of this type. As we rotate the polygon a full cycle in a monotone fashion, we reach every angle exactly once. Therefore, no structure can appear more than once, and the claim follows. □

Consider a vertex u of P. We will now describe its effect on the diagram. Let α be the angle between the two edges of P that share u. Note that instead of rotating the polygon by θ we will rotate $\overrightarrow{qv}$ by $-\theta$. In the course of this process u draws a z-monotone curve.

For a given vertex $u \in P$, let us divide the z axis (the rotation) into four parts. In the first part u is on the lower chain. This part is α-long and the height of the corresponding vertex in the diagram is the width of P in the direction $\overrightarrow{qv}$ rotated by $-\theta$ and originating from u. In this part u draws a curve which is the concatenation of m *simple curves*. Each simple curve represents the widths of P as above, in a range of rotations: from the rotation in which $\overrightarrow{qv}$ points to a vertex $w \in P$ to the rotation in which it points to a neighbor of w.

In the second and forth parts, each one of length $\pi-\theta$, u is the rightmost and leftmost vertices of the region, respectively. In the third part u is on the upper chain. This part is α-long and the height of the corresponding vertex in the diagram is the width of P in the direction $\overrightarrow{qv}$ rotated by $\pi - \theta$ and originating from a point on the boundary of P such that it points at u.

We locate the origin of the vector $\overrightarrow{qv}$ at u and start to rotate it. When it points to the inside of P, u is on the lower chain. When it points outside of P, u is on the upper chain. In the transitions between those two parts, u is the rightmost or leftmost vertex of the region. The height of the vertex is calculated as in the TS diagram. Note that both the first and the third parts have the same heights. The difference is that in the first part the x coordinate is fixed and in the third part the vertex moves from right to left as described above.

We now consider again a set of more than two points, and focus on how different regions in the diagram interact. Recall that the rotation axis "wraps around."

Theorem 2 *If points $v, w \in S$ are equidistant from q, then their regions are identical, except that one of them is a shifted version of the other by $\angle vqw$ along the rotation axis.*

Proof. The theorem holds as a direct consequence of how the diagram is built. Its starting direction is arbitrary, and then the polygon is rotated by 2π. The only difference between v and w is the relative orientation relative to q. □

Clearly, if we move v further from q in the direction $\overrightarrow{qv}$, then the region of v will be stretched up by $d(v', q)/d(v, q)$, where v' is the new location of v. The next theorem summarizes the above discussion stating the complexity of a single region.

Theorem 3 *The boundary of every region contains m curves, and each such curve is the concatenation of m simple curves. The total number of intersections among these curves is $O(m^2)$, which is the total complexity of the boundary.*

Proof. The first part of the theorem is a result of the discussion following Theorem 1. Each polygon vertex draws one curve. The second part of the theorem is the result of Theorem 1. □

Barequet and Dickerson [3] showed that in a TS diagram every two regions intersect at most once. What happens to an intersection point in the three-dimensional diagram?

We focus on two segments s_1 and s_2 of the boundaries of two different regions. Those segments intersect in a point t. As we rotate the polygon s_1 and s_2 move, and so does t while drawing a three-dimensional curve. We can compute the movement of s_1 and s_2, so we can compute the curve drawn by t as well. After rotating enough the polygon, s_1 and s_2 will cease intersecting. Either s_1 will intersect a neighbor of s_2, s_2 will intersect a neighbor of s_1, or a neighbor of s_1 will intersect a neighbor of s_2. A new curve will be drawn starting from the endpoint of the last curve. We upper bound the number of such events.

Theorem 4 *Every intersection point of two regions draws a curve which is made of $\Theta(m^2)$ simple curves in the worst case.*

Proof. Omitted in this short version of the paper. □

There is another type of event, in which four points are on the boundary of a copy of the polygon. This event occurs when three regular intersection points coincide. Calculating those events is relatively easy. Every intersection point has four other intersection points as immediate neighbors along the boundary regions. A trivial upper bound on the number of these events is $O(n^3m^4)$: we need to choose three points (the forth is the one used to build the diagram) and three polygon segments for those points. This is the best upper bound we know. We show now an example in which there are $\Omega(n^3m)$ such events. We will use a regular polygon with $m = 4k$ ($k \in \mathbb{N}$) segments. By rotating it we get k positions in which four segments are parallel to the x, y axes. Those segments will cover the four points. This will give us the factor $\Omega(m)$. We put $n = 3j + 1$ ($j \in \mathbb{N}$) points as follows. We put j points on the positive part of the y-axis. The points are infinitesimally close to each other at distance about 1 from the origin. We put the second set of j close points around the location $(0, -1)$. We put the third set around $(-1, 0)$. The single point that is used to build the diagram is put at distance one from the origin rotated counter-clockwise by the angle β from the positive x-axis. By choosing a point from each set we get four points that will be on the boundary of the polygon. The vertical distance between the upper and lower points is about 2. The horizontal distance between the right and left points is $1+\varepsilon+\cos\beta$, where ε is infinitesimally close to zero. Instead of rotating the polygon we will rotate the points by the angle α. We choose α that makes the horizontal distance and the vertical distance equal, so as to enable us to cover the points. After rotating the points the vertical distance is $2 \cdot \cos\alpha$. The horizontal distance is $(1+\varepsilon)\cos\alpha+\cos(\alpha+\beta)$. We get $(1-\varepsilon)\cos\alpha = \cos(\alpha+\beta)$, and after simplification, we get $\tan\alpha = \cot\beta + (\varepsilon-1)/\sin\beta$. If β is small, then α is small too, so even when m is big and every polygon segment is short we can choose β such that the segments are long enough to cover the points.

Now we are ready to state the complexity of the TSR diagram of a polygon P with m vertices with respect to a set S of n points.

The TSR diagram is made of n three-dimensional regions, the boundary of each of which contains $O(m^2)$ simple curves. The total number of intersections among these curves is $O(m^2)$ (Theorem 3). The regions intersect each other $O(n^2)$ times. The intersection of two regions is bounded by the xz-surface and by the boundaries of the two regions. The intersection of the boundaries is the concatenation of $\Theta(m^2)$ simple curves in the worst case (Theorem 4). Altogether we have $O(n^2m^2)$ simple curves, and $O(n^2m^2)$ vertices. In the full version of the paper we show that this bound is attainable in the worst case. That is, we have $\Theta(n^2m^2)$ simple curves in the worse case (see also [11, Theorem 2.6]). The number of four-point intersection points is $O(n^3m^4)$. We conclude that the complexity of the diagram is $O(n^3m^4)$ in the worst case.

4 Computation

We build a TS diagram at $z = 0$ (the original orientation). Then we use a sweep procedure to build the rest of the diagram. For each region vertex and intersection point we compute the next simple curve and the next event. Whenever there is a change in an intersection point, e.g, a simple curve ends and another curve starts, we check if it coincides with its neighbors. At any point in time there are $O(n(n+m))$ events in the event queue: one for each TS vertex and two for each TS intersection (the intersection point and the closest four-point intersection point involving that intersection point). There are $O(nm^2)$ region-vertex events, $O(nm^2)$ structure-change events, $O(n^2m^2)$ region-intersection events, and $O(n^3m^4)$ complex intersection events, for a total of $O(n^3m^4)$ events. Each event is handled in $O(\log(nm))$ time because it involves performing local computations on one or two simple curves and standard operations on the event queue. Every event causes the insertion of another event to the event queue. This takes $O(\log(nm))$ time for each event. The time complexity of the sweep is therefore $O(n^3m^4\log(nm))$.

Although we can build the entire diagram, this may be redundant for some applications. According to Theorem 2, we can build a region for a point that is one unit to the right of q. Then, all the real regions can be computed on-demand from that region. Even that region doesn't have to be computed entirely. Our description of the diagram will include one TS diagram with only one region for a point one unit to the right of q, and the sorted set of $O(m^2)$ critical rotation angles (as defined in Theorem 3).

Building this data structure takes $O(m^2 + n)$ time. This is also the space that this algorithm requires.

5 Applications

We now describe a few applications of the TSR diagram. We show how to solve some of the problems presented in [3] when applied to the three-dimensional case.

Theorem 5 *The smallest scale of P, if there exists a translated and rotated version of P in contact with q and containing the entire point set S, can be computed in $O(n^3 m^4 \log(nm))$ time.*

Proof. Define the *count* of a point t in the TSR diagram to be the number of regions which contain t, including regions that have t on their boundaries. Since t represents a TSR of P, the count of t is the number of points covered by P when it is translated, scaled, and rotated according to t.

The problem can be solved while building the diagram. When we build the first (lowest) TS diagram, we will compute the counts of each region vertex and intersection point. We will also compute the count of each region vertex and intersection point that we will reach in the course of computing the diagram. The result will be the the extreme point along the scale direction among those points with count $n - 1$. □

Theorem 6 *For a given scale α and a point q, the maximum number of points that can be contained in a copy of P in contact with q and scaled by α can be determined in $O(nk^2 m^2 \log(km))$ time, where k is the maximum of contained points.*

Proof. We scale the polygon by α and provide it as input to the algorithm described in [6, Theorem 2]. □

Note that if the full diagram is given, we can also solve this problem by traversing the diagram along the z-axis and keeping the status of the line at scale $= \alpha$ within the sweep-plane. Since all the simple curves are y-monotone, each simple curve intersects the plane scale $= \alpha$ (and hence, its intersection with the swept plane) at most once. Therefore, the time complexity of the algorithm is the time needed to traverse the diagram. The time needed to traverse the diagram is comparable to its complexity, so the time complexity of the algorithm is $O(n^3 m^4)$.

Theorem 7 *Given a precomputed TSR diagram and a point q, the smallest scale of P in contact with q and containing at least k points can be computed in $O(n^3 m^4)$ time.*

Proof. We apply the following algorithm (see also [11, §4]). Define the *count* of a point t in the TSR as in the proof of Theorem 5. We observe that given a vertex of the DCEL structure that describes the diagram and the depth of a cell near it, we can traverse the neighboring cells and know their depths. Therefore, all we need is a starting point and its count. We traverse the diagram starting from a point with a count of 1 which was computed while building the diagram. Whenever we reach a region vertex or an intersection point we compute its count. We consider all the points with count at least k and return the one that represents the smallest scale (the one with highest y-value). The time needed to traverse the diagram is comparable to its complexity, So the time complexity of this algorithm is $O(n^3 m^4)$. □

We are not aware of any algorithm that can solve the problem more efficiently and without building the diagram explicitly.

References

[1] Barequet, G., Bose, P., Dickerson, M.T., and Goodrich, M.T.: Optimizing a constrained convex polygonal annulus, J. of Discrete Algorithms, **3** (2005), 1–26.

[2] Barequet, G., Briggs, A.J., Dickerson, M.T., and Goodrich, M.T.: Offset-polygon annulus placement problems, CGTA, **11** (1998), 125–141.

[3] Barequet, G., and Dickerson, M.T.: The translation-scale diagram for point-containing placements of a convex polygon, Proc. 12th CCCG, Fredericton, New Brunswick, Canada, 7–12, 2000.

[4] Barequet, G., Dickerson, M.T., and Goodrich, M.T.: Voronoi diagrams for polygon-offset distance functions, DCG, **25** (2001), 271–291.

[5] Barequet, G., Dickerson, M., and Pau, P.: Translating a convex polygon to contain a maximum number of points, CGTA, **8** (1997), 167–179.

[6] Dickerson, M., and Scharstein, D.: Optimal placement of a convex polygon to maximize point containment, CGTA, **11** (1998), 1–16.

[7] Duncan, C.A., Goodrich, M.T., and Ramos, E.A.: Efficient approximation and optimization algorithms for computational metrology, Proc. 8th Ann. ACM-SIAM SODA, New Orleans, LA, 121–130, 1997.

[8] Efrat, A., Sharir, M., and Ziv, A.: Computing the smallest k-enclosing circle and related problems, CGTA, **4** (1994), 119–136.

[9] Eppstein, D., and Erickson, J.: Iterated nearest neighbors and finding minimal polytopes, DCG, **11** (1994), 321–350.

[10] Guibas, L., Motwani, R., and Raghavan, P.: The robot localization problem, in Algorithmic Foundations of Robotics, A.K. Peters, Ltd., 1995, 269–282.

[11] Scharf, Y.: Covering points with a polygon, M.Sc. Thesis, Dept. of Computer Science, The Technion–Israel Institute of Technology, Haifa, Israel, October 2004.